GaussDB

Reference(Distributed_V2.0-10.1.0)

Issue 01

Date 2026-10-21



HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2026. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Cloud Computing Technologies Co., Ltd.

Trademarks and Permissions


and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: https://www.huaweicloud.com/intl/en-us/

Contents

1 SQL Reference 1
1.1 GaussDB SQL 1
1.2 Keywords 2
1.3 Data Types 42
1.3.1 Numeric Types 42
1.3.2 Monetary Types 58
1.3.3 Boolean Types 59
1.3.4 Character Types 60
1.3.5 Binary Types 70
1.3.6 Date/Time Types 73
1.3.7 Geometric Types 99
1.3.8 Network Address Types 102
1.3.9 Bit String Types 104
1.3.10 UUID Type 105
1.3.11 Text Search Types 106
1.3.12 JSON/JSONB Types 109
1.3.13 HLL Type 113
1.3.14 Range Types 117
1.3.15 OID Types 122
1.3.16 Pseudo-Types 124
1.3.17 Data Types Used by the Ledger Database 125
1.3.18 XML Types 126
1.3.19 XMLType 128
1.3.20 ACLItem Type 130
1.3.21 Array Types 131
1.3.22 Data Types Supported by the Vectorized Executor 136
1.3.23 ROWID Type 141
1.3.24 Vector Data Types 149
1.4 Character Sets and Collations 150
1.4.1 Character Sets and Collations of the Client Connection 150
1.4.2 Database-level Character Sets and Collations 151
1.4.3 Schema-level Character Sets and Collations 152
1.4.4 Table-level Character Sets and Collations 153

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.4.5 Column-Level Character Sets and Collations 154
1.4.6 Character Sets and Collations of Expressions of the String Type 156
1.4.7 Rules for Combining Character Sets and Collations 156
1.5 Constant and macro 158
1.6 Functions and Operators 159
1.6.1 Logical Operators 162
1.6.2 Comparison Operators 163
1.6.3 Character Processing Functions and Operators 164
1.6.4 Binary String Functions and Operators 229
1.6.5 Bit String Functions and Operators 232
1.6.6 Pattern Matching Operators 234
1.6.7 Arithmetic Functions and Operators 243
1.6.8 Date and Time Processing Functions and Operators 264
1.6.9 Type Conversion Functions 341
1.6.10 Geometric Functions and Operators 376
1.6.11 Network Address Functions and Operators 387
1.6.12 Text Search Functions and Operators 392
1.6.13 JSON/JSONB Functions and Operators 399
1.6.14 HLL Functions and Operators 429
1.6.15 SEQUENCE Functions 442
1.6.16 Array Functions and Operators 446
1.6.17 Range Functions and Operators 455
1.6.18 Aggregate Functions 503
1.6.19 Window Functions 527
1.6.20 Security Functions 540
1.6.21 Ledger Database Functions 555
1.6.22 Encrypted Functions and Operators 557
1.6.23 Set Return Functions 568
1.6.24 Overloaded Query Functions 571
1.6.25 Conditional Expression Functions 575
1.6.26 System Information Functions 583
1.6.27 System Administration Functions 632
1.6.27.1 Configuration Setting Functions 632
1.6.27.2 Universal File Access Functions 633
1.6.27.3 Server Signal Functions 635
1.6.27.4 Backup and Restoration Control Functions 638
1.6.27.5 Dual-Cluster DR Control Functions 649
1.6.27.6 Dual-Cluster DR Query Functions 655
1.6.27.7 Snapshot Synchronization Functions 660
1.6.27.8 Database Object Functions 661
1.6.27.9 Advisory Lock Functions 669
1.6.27.10 Logical Replication Functions 672

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.6.27.11 Segment-Page Storage Functions 719
1.6.27.12 Bucket System Functions 743
1.6.27.13 Undo System Functions 753
1.6.27.14 HTAP System Functions 776
1.6.27.15 Other Functions 810
1.6.28 SPM Functions 884
1.6.29 Statistics Information Functions 888
1.6.30 Trigger Functions 1010
1.6.31 Hash Functions 1012
1.6.32 Prompt Message Function 1019
1.6.33 Fault Injection System Function 1020
1.6.34 Redistribution Functions 1020
1.6.35 Distribution Key Recommendation Functions 1021
1.6.36 Internal Functions 1026
1.6.37 Sensitive Data Discovery Function 1038
1.6.38 Dynamic Data Masking Functions 1038
1.6.39 Hierarchical Recursion Query Functions 1039
1.6.40 Hotkey Feature Functions 1040
1.6.41 Global SysCache Functions 1041
1.6.42 Data Damage Detection and Repair Functions 1043
1.6.43 Multixact Recycling Function 1069
1.6.44 Functions of the XML Type 1069
1.6.45 Functions of the XMLType Type 1088
1.6.46 Pivot Table Functions 1100
1.6.47 Global PL/SQL Cache Functions 1101
1.6.48 UUID Functions 1107
1.6.49 SQL Statement Concurrency Control Functions 1108
1.6.50 SQL Tool Functions 1117
1.6.51 Multi-Tenant Database Functions 1126
1.6.52 Global Temporary Table Functions 1126
1.6.53 Database Link Functions 1131
1.6.54 GSMonitor System Resource Statistics Functions 1135
1.6.55 Sorting Calculation Functions 1141
1.6.56 Autonomous Transaction Feature Functions 1147
1.6.57 DSL Rule Management Functions 1152
1.6.58 ROWID Functions 1155
1.6.59 Functions for Recompilation upon Invalidation 1156
1.6.60 Other System Functions 1158
1.6.61 Deprecated Functions 1235
1.6.62 Vector Data Functions and Operators 1238
1.6.62.1 Vector Distance Calculation APIs 1238
1.6.62.2 Vector Operation Functions 1239

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.6.62.3 Vector Functions and Operators....1243
1.6.62.4 Functions and Operators for Similar Document Retrieval and Ranking....1254
1.7 Expressions....1256
1.7.1 Simple Expressions....1256
1.7.2 Condition Expressions....1259
1.7.3 Subquery Expressions....1263
1.7.4 Array Expressions....1266
1.7.5 Row Expressions....1268
1.7.6 Time Interval Expressions....1269
1.8 Pseudocolumn....1271
1.9 Type Conversion....1275
1.9.1 Overview....1276
1.9.2 Operators....1277
1.9.3 Functions....1279
1.9.4 Value Storage....1281
1.9.5 UNION, CASE, and Related Constructs....1283
1.10 System Operation....1289
1.11 Controlling Transactions....1290
1.12 Full-Text Search....1290
1.12.1 Introduction....1291
1.12.1.1 Full-Text Search....1291
1.12.1.2 What Is a Document....1292
1.12.1.3 Basic Text Matching....1292
1.12.1.4 Configuration....1293
1.12.2 Tables and Indexes....1294
1.12.2.1 Searching a Table....1294
1.12.2.2 Creating an Index....1295
1.12.3 Controlling Text Search....1297
1.12.3.1 Parsing Documents....1297
1.12.3.2 Parsing Queries....1298
1.12.3.3 Ranking Search Results....1300
1.12.3.4 Highlighting Search Results....1302
1.12.4 Additional Features....1303
1.12.4.1 Manipulating tsvector....1303
1.12.4.2 Manipulating Queries....1304
1.12.4.3 Rewriting Queries....1305
1.12.4.4 Collecting Document Statistics....1306
1.12.5 Parsers....1307
1.12.6 Dictionaries....1310
1.12.6.1 Overview....1310
1.12.6.2 Stop Words....1311
1.12.6.3 Simple Dictionary....1311

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.12.6.4 Snowball Dictionary....1312
1.12.7 Configuration Examples....1313
1.12.8 Testing and Debugging Text Search....1313
1.12.8.1 Testing a Tokenizer....1313
1.12.8.2 Testing a Parser....1314
1.12.8.3 Testing a Dictionary....1315
1.12.9 Restrictions....1316
1.13 SQL Syntax....1316
1.13.1 SQL Syntax....1316
1.13.2 DCL Syntax Overview....1317
1.13.3 DDL Syntax Overview....1318
1.13.4 DML Syntax Overview....1332
1.13.5 Other Syntax List....1335
1.13.6 Online DDL and Related Operations....1337
1.13.7 A....1341
1.13.7.1 ABORT....1341
1.13.7.2 ALTER AGGREGATE....1342
1.13.7.3 ALTER ASYNC ENCRYPTION KEY ROTATION....1344
1.13.7.4 ALTER AUDIT POLICY....1344
1.13.7.5 ALTER COLUMN ENCRYPTION KEY....1348
1.13.7.6 ALTER COORDINATOR....1349
1.13.7.7 ALTER DATABASE....1350
1.13.7.8 ALTER DATABASE LINK....1356
1.13.7.9 ALTER DEFAULT PRIVILEGES....1358
1.13.7.10 ALTER DIRECTORY....1361
1.13.7.11 ALTER FOREIGN DATA WRAPPER....1361
1.13.7.12 ALTER FOREIGN TABLE....1363
1.13.7.13 ALTER FUNCTION....1364
1.13.7.14 ALTER GLOBAL CONFIGURATION....1369
1.13.7.15 ALTER GROUP....1371
1.13.7.16 ALTER INDEX....1373
1.13.7.17 ALTER LANGUAGE....1380
1.13.7.18 ALTER MASKING POLICY....1380
1.13.7.19 ALTER MATERIALIZED VIEW....1384
1.13.7.20 ALTER NODE....1386
1.13.7.21 ALTER NODE GROUP....1390
1.13.7.22 ALTER PACKAGE....1392
1.13.7.23 ALTER PROCEDURE....1394
1.13.7.24 ALTER RESOURCE LABEL....1398
1.13.7.25 ALTER RESOURCE POOL....1400
1.13.7.26 ALTER ROLE....1403
1.13.7.27 ALTER ROW LEVEL SECURITY POLICY....1407

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.7.28 ALTER SCHEMA 1409
1.13.7.29 ALTER SEQUENCE 1413
1.13.7.30 ALTER SERVER 1415
1.13.7.31 ALTER SESSION 1418
1.13.7.32 ALTER SYNONYM 1421
1.13.7.33 ALTER SYSTEM KILL SESSION 1422
1.13.7.34 ALTER TABLE 1423
1.13.7.35 ALTER TABLE PARTITION 1459
1.13.7.36 ALTER TABLE SUBPARTITION 1477
1.13.7.37 ALTER TABLESPACE 1488
1.13.7.38 ALTER TEXT SEARCH CONFIGURATION 1491
1.13.7.39 ALTER TEXT SEARCH DICTIONARY 1495
1.13.7.40 ALTER TRIGGER 1497
1.13.7.41 ALTER TYPE 1499
1.13.7.42 ALTER USER 1504
1.13.7.43 ALTER VIEW 1507
1.13.7.44 ANALYZE | ANALYSE 1511
1.13.7.45 AUTOHINT 1519
1.13.7.46 AUTOHINT DROP MODEL 1521
1.13.7.47 AUTOHINT PURGE 1522
1.13.8 B 1523
1.13.8.1 BEGIN 1523
1.13.9 C 1525
1.13.9.1 CALL 1525
1.13.9.2 CHECKPOINT 1528
1.13.9.3 CLEAN CONNECTION 1528
1.13.9.4 CLOSE 1530
1.13.9.5 CLUSTER 1531
1.13.9.6 COMMENT 1535
1.13.9.7 COMMIT | END 1537
1.13.9.8 COMMIT PREPARED 1539
1.13.9.9 COPY 1540
1.13.9.10 CREATE AGGREGATE 1564
1.13.9.11 CREATE AUDIT POLICY 1567
1.13.9.12 CREATE BARRIER 1571
1.13.9.13 CREATE CLIENT MASTER KEY 1572
1.13.9.14 CREATE COLUMN ENCRYPTION KEY 1574
1.13.9.15 CREATE CONVERSION 1576
1.13.9.16 CREATE DATABASE 1576
1.13.9.17 CREATE DATABASE LINK 1589
1.13.9.18 CREATE DIRECTORY 1591
1.13.9.19 CREATE EXTENSION 1593

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.9.20 CREATE FOREIGN DATA WRAPPER....1595
1.13.9.21 CREATE FOREIGN TABLE....1597
1.13.9.22 CREATE FUNCTION....1611
1.13.9.23 CREATE GLOBAL INDEX....1623
1.13.9.24 CREATE GROUP....1631
1.13.9.25 CREATE INCREMENTAL MATERIALIZED VIEW....1633
1.13.9.26 CREATE INDEX....1635
1.13.9.27 CREATE LANGUAGE....1656
1.13.9.28 CREATE MASKING POLICY....1657
1.13.9.29 CREATE MATERIALIZED VIEW....1661
1.13.9.30 CREATE MODEL....1663
1.13.9.31 CREATE NODE....1664
1.13.9.32 CREATE NODE GROUP....1668
1.13.9.33 CREATE PACKAGE....1669
1.13.9.34 CREATE PROCEDURE....1673
1.13.9.35 CREATE RESOURCE LABEL....1679
1.13.9.36 CREATE RESOURCE POOL....1681
1.13.9.37 CREATE ROLE....1685
1.13.9.38 CREATE ROW LEVEL SECURITY POLICY....1693
1.13.9.39 CREATE SCHEMA....1697
1.13.9.40 CREATE SECURITY LABEL....1701
1.13.9.41 CREATE SEQUENCE....1702
1.13.9.42 CREATE SERVER....1706
1.13.9.43 CREATE SYNONYM....1708
1.13.9.44 CREATE TABLE....1711
1.13.9.45 CREATE TABLESPACE....1749
1.13.9.46 CREATE TABLE AS....1753
1.13.9.47 CREATE TABLE PARTITION....1761
1.13.9.48 CREATE TABLE PARTITION | SUBPARTITION AS....1780
1.13.9.49 CREATE TABLE SUBPARTITION....1788
1.13.9.50 CREATE TEXT SEARCH CONFIGURATION....1803
1.13.9.51 CREATE TEXT SEARCH DICTIONARY....1805
1.13.9.52 CREATE TRIGGER....1811
1.13.9.53 CREATE TYPE....1819
1.13.9.54 CREATE USER....1828
1.13.9.55 CREATE VIEW....1833
1.13.9.56 CREATE WEAK PASSWORD DICTIONARY....1842
1.13.9.57 CURSOR....1843
1.13.10 D....1846
1.13.10.1 DEALLOCATE....1846
1.13.10.2 DECLARE....1847
1.13.10.3 DELETE....1850

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.10.4 DO ... 1857
1.13.10.5 DROP AGGREGATE ... 1857
1.13.10.6 DROP AUDIT POLICY ... 1859
1.13.10.7 DROP CLIENT MASTER KEY ... 1860
1.13.10.8 DROP COLUMN ENCRYPTION KEY ... 1861
1.13.10.9 DROP DATABASE ... 1861
1.13.10.10 DROP DATABASE LINK ... 1863
1.13.10.11 DROP DIRECTORY ... 1864
1.13.10.12 DROP EXTENSION ... 1865
1.13.10.13 DROP FOREIGN DATA WRAPPER ... 1866
1.13.10.14 DROP FOREIGN TABLE ... 1867
1.13.10.15 DROP FUNCTION ... 1868
1.13.10.16 DROP GLOBAL CONFIGURATION ... 1870
1.13.10.17 DROP GROUP ... 1871
1.13.10.18 DROP INDEX ... 1871
1.13.10.19 DROP LANGUAGE ... 1874
1.13.10.20 DROP MASKING POLICY ... 1874
1.13.10.21 DROP MATERIALIZED VIEW ... 1876
1.13.10.22 DROP MODEL ... 1877
1.13.10.23 DROP NODE ... 1877
1.13.10.24 DROP NODE GROUP ... 1878
1.13.10.25 DROP OWNED ... 1879
1.13.10.26 DROP PACKAGE ... 1880
1.13.10.27 DROP PROCEDURE ... 1881
1.13.10.28 DROP RESOURCE LABEL ... 1882
1.13.10.29 DROP RESOURCE POOL ... 1883
1.13.10.30 DROP ROLE ... 1884
1.13.10.31 DROP ROW LEVEL SECURITY POLICY ... 1885
1.13.10.32 DROP SCHEMA ... 1886
1.13.10.33 DROP SECURITY LABEL ... 1887
1.13.10.34 DROP SEQUENCE ... 1888
1.13.10.35 DROP SERVER ... 1889
1.13.10.36 DROP SYNONYM ... 1890
1.13.10.37 DROP TABLE ... 1891
1.13.10.38 DROP TABLESPACE ... 1893
1.13.10.39 DROP TEXT SEARCH CONFIGURATION ... 1894
1.13.10.40 DROP TEXT SEARCH DICTIONARY ... 1895
1.13.10.41 DROP TRIGGER ... 1896
1.13.10.42 DROP TYPE ... 1899
1.13.10.43 DROP USER ... 1900
1.13.10.44 DROP VIEW ... 1902
1.13.10.45 DROP WEAK PASSWORD DICTIONARY ... 1903

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.11 E... 1904
1.13.11.1 EXECUTE... 1904
1.13.11.2 EXECUTE DIRECT... 1905
1.13.11.3 EXPDP DATABASE... 1907
1.13.11.4 EXPDP TABLE... 1908
1.13.11.5 EXPLAIN... 1908
1.13.11.6 EXPLAIN PLAN... 1919
1.13.11.7 EXPLAIN AUTOHINT... 1921
1.13.12 F... 1922
1.13.12.1 FETCH... 1922
1.13.13 G... 1925
1.13.13.1 GENERATED UPDATE SYSTEM OBJECT... 1925
1.13.13.2 GRANT... 1926
1.13.14 I... 1939
1.13.14.1 IMPDP DATABASE CREATE... 1939
1.13.14.2 IMPDP RECOVER... 1939
1.13.14.3 IMPDP TABLE... 1940
1.13.14.4 IMPDP TABLE PREPARE... 1941
1.13.14.5 INSERT... 1941
1.13.14.6 INSERT ALL... 1958
1.13.15 L... 1966
1.13.15.1 LOAD DATA... 1967
1.13.15.2 LOCK... 1974
1.13.15.3 LOCK BUCKETS... 1980
1.13.16 M... 1980
1.13.16.1 MARK BUCKETS... 1981
1.13.16.2 MERGE INTO... 1981
1.13.16.3 MOVE... 1984
1.13.17 P... 1986
1.13.17.1 PREDICT BY... 1986
1.13.17.2 PREPARE... 1986
1.13.17.3 PREPARE TRANSACTION... 1987
1.13.17.4 PURGE... 1988
1.13.18 R... 1990
1.13.18.1 REASSIGN OWNED... 1990
1.13.18.2 REFRESH INCREMENTAL MATERIALIZED VIEW... 1992
1.13.18.3 REFRESH MATERIALIZED VIEW... 1993
1.13.18.4 REFRESH SYSTEM OBJECT... 1994
1.13.18.5 REINDEX... 1994
1.13.18.6 RELEASE SAVEPOINT... 1999
1.13.18.7 RENAME TABLE... 2000
1.13.18.8 REPLACE... 2001

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.18.9 RESET 2007
1.13.18.10 REVOKE 2008
1.13.18.11 ROLLBACK 2013
1.13.18.12 ROLLBACK PREPARED 2014
1.13.18.13 ROLLBACK TO SAVEPOINT 2015
1.13.19 S 2016
1.13.19.1 SAVEPOINT 2016
1.13.19.2 SECURITY LABEL ON 2018
1.13.19.3 SELECT 2019
1.13.19.3.1 Simple Queries 2048
1.13.19.3.2 Conditional Queries 2049
1.13.19.3.3 Group Queries 2049
1.13.19.3.4 Pagination Queries 2051
1.13.19.3.5 Partition Queries 2053
1.13.19.3.6 Join Queries 2053
1.13.19.3.7 Subqueries 2054
1.13.19.3.8 Hierarchical Queries 2057
1.13.19.3.9 Compound Queries 2059
1.13.19.3.10 Rows to Columns and Columns to Rows 2060
1.13.19.3.11 Window Function Queries 2060
1.13.19.3.12 Simplified Queries 2061
1.13.19.3.13 Queries Using Hints 2061
1.13.19.4 SELECT INTO 2062
1.13.19.5 SET 2064
1.13.19.6 SET CONSTRAINTS 2067
1.13.19.7 SET ROLE 2068
1.13.19.8 SET SESSION AUTHORIZATION 2070
1.13.19.9 SET TRANSACTION 2071
1.13.19.10 SHOW 2073
1.13.19.11 SHUTDOWN 2075
1.13.19.12 START TRANSACTION 2075
1.13.20 T 2077
1.13.20.1 TIMECAPSULE TABLE 2078
1.13.20.2 TRUNCATE 2083
1.13.21 U 2086
1.13.21.1 UPDATE 2086
1.13.21.1.1 Pseudo-update of Distribution Keys 2095
1.13.22 V 2101
1.13.22.1 VACUUM 2101
1.13.22.2 VALUES 2108
1.14 Appendix 2110
1.14.1 Extended Functions 2110

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.14.2 Dollar-Quoted String Constants....2110
1.14.3 Row Expression Function Whitelist....2111
2 Stored Procedures....2177
2.1 Overview....2177
2.2 Data Types....2177
2.3 Data Type Conversion....2177
2.4 Arrays, Collections, and Records....2179
2.4.1 Arrays....2179
2.4.1.1 Use of Array Types....2179
2.4.1.2 Array Functions....2182
2.4.2 Collections....2188
2.4.2.1 Use of Collection Types....2188
2.4.2.2 Collection Functions....2195
2.4.3 Records....2208
2.5 DECLARE Syntax....2213
2.5.1 Basic Structure....2213
2.5.2 Anonymous Blocks....2214
2.5.3 Subprograms....2216
2.5.3.1 Standalone Subprograms....2216
2.5.3.2 Package Subprograms....2217
2.5.3.3 Nested Subprograms....2217
2.6 Basic Statements....2220
2.6.1 Variable Definition Statements....2220
2.6.2 Assignment Statements....2223
2.6.3 Call Statements....2225
2.7 Dynamic Statements....2226
2.7.1 Dynamically Executing Query Statements....2226
2.7.2 Dynamically Executing Non-Query Statements....2230
2.7.3 Dynamically Calling Stored Procedures....2231
2.7.4 Dynamically Calling Anonymous Blocks....2233
2.8 Control Statements....2237
2.8.1 RETURN Statements....2237
2.8.1.1 RETURN....2237
2.8.1.2 RETURN NEXT/RETURN QUERY....2238
2.8.2 Conditional Statements....2239
2.8.3 Loop Statements....2241
2.8.4 Branching Statements....2246
2.8.5 NULL Statements....2248
2.8.6 Error Trapping Statements....2248
2.8.6.1 Obtaining Exception Information....2251
2.8.6.2 Obtaining Current Execution Information....2252
2.8.7 GOTO Statements....2253

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2.9 Transaction Statements 2255
2.10 Other Statements 2265
2.10.1 Lock Operations 2265
2.10.2 Cursor Operations 2265
2.11 Cursors 2265
2.11.1 Overview 2265
2.11.2 Explicit Cursors 2266
2.11.3 Implicit Cursors 2272
2.11.4 Cursor Loop 2273
2.12 Advanced Packages 2274
2.12.1 Basic APIs 2274
2.12.1.1 PKG_SERVICE 2274
2.12.1.2 PKG_UTIL 2285
2.12.1.3 DBE_XML 2330
2.12.2 Secondary Encapsulation APIs 2356
2.12.2.1 DBE_APPLICATION_INFO 2356
2.12.2.2 DBE_COMPRESSION 2359
2.12.2.3 DBE_DESCRIBE 2365
2.12.2.4 DBE_FILE 2371
2.12.2.5 DBE_HEAT_MAP 2398
2.12.2.6 DBE_ILM 2400
2.12.2.7 DBE_ILM_ADMIN 2404
2.12.2.8 DBE_LICENSE 2410
2.12.2.9 DBE_LOB 2414
2.12.2.10 DBE_MATCH 2441
2.12.2.11 DBE_OBFUSCATION_TOOLKIT 2442
2.12.2.12 DBE_OUTPUT 2451
2.12.2.13 DBE_RANDOM 2456
2.12.2.14 DBE_RAW 2457
2.12.2.15 DBE_SCHEDULER 2473
2.12.2.16 DBE_SESSION 2512
2.12.2.17 DBE_SQL 2516
2.12.2.18 DBE_STATS 2561
2.12.2.19 DBE_TASK 2644
2.12.2.20 DBE_UTILITY 2656
2.12.2.21 DBE_XMLDOM 2673
2.12.2.22 DBE_XMLGEN 2724
2.12.2.23 DBE_XMLPARSER 2733
2.12.2.24 PRVT_ILM 2740
2.12.3 Internal Functions 2743
2.13 Retry 2745
2.14 Package 2745

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2.15 Debugging....2746
2.16 Keyword....2750
3 Compatibility Description....2756
3.1 Overview of GaussDB Compatibility....2756
3.2 Oracle Compatibility Description....2756
3.2.1 Overview of Oracle Database Compatibility....2756
3.2.2 Basic SQL Elements....2756
3.2.2.1 Data Types....2757
3.2.2.2 Data Type Comparison Rules....2766
3.2.2.3 Literals....2767
3.2.2.4 Format Models....2768
3.2.2.5 Nulls....2768
3.2.2.6 Comments....2769
3.2.2.7 Database Objects....2769
3.2.2.8 Database Object Names and Qualifiers....2772
3.2.2.9 Syntax for Schema Objects and Parts in SQL Statements....2772
3.2.3 Pseudocolumns....2773
3.2.4 Operators....2774
3.2.5 Expressions....2777
3.2.6 Conditions....2781
3.2.7 Drivers....2783
3.2.7.1 JDBC....2783
3.2.7.1.1 Array....2783
3.2.7.1.2 Struct....2800
3.2.8 Common SQL DDL Clauses....2815
3.2.9 SQL Queries and Subqueries....2825
3.2.10 PL/SQL Language....2830
3.2.10.1 Basic PL/SQL Syntax....2830
3.2.10.2 Data Type Compatibility....2833
3.2.10.3 Control Statements....2833
3.2.10.4 Collections and Records....2835
3.2.10.5 Static SQL Statements....2840
3.2.10.6 Dynamic SQL Statements....2844
3.2.10.7 Triggers....2845
3.2.11 System Functions....2853
3.2.11.1 Single-Row Functions....2854
3.2.11.2 Other Functions....2897
3.2.12 System Views....2899
3.2.13 Advanced Packages....2905
3.3 MySQL Compatibility Description....2964
3.3.1 Overview of MySQL Compatibility....2965
3.3.2 M-compatible Mode....2966

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.2.1 Data Types 2966
3.3.2.1.1 Numeric Data Types 2966
3.3.2.1.2 Date and Time Data Types 2968
3.3.2.1.3 String Data Types 2970
3.3.2.1.4 Binary Data Types 2973
3.3.2.1.5 Attributes Supported by Data Types 2977
3.3.2.1.6 Data Type Conversion 2979
3.3.2.2 System Functions 2985
3.3.2.2.1 System Function Compatibility Overview 2985
3.3.2.2.2 Flow Control Functions 2987
3.3.2.2.3 Date and Time Functions 2987
3.3.2.2.4 String Functions 2994
3.3.2.2.5 Forced Conversion Functions 2998
3.3.2.2.6 Encryption Functions 3000
3.3.2.2.7 Comparison Functions 3000
3.3.2.2.8 Aggregate Functions 3001
3.3.2.2.9 Numeric Operation Functions 3004
3.3.2.2.10 Other Functions 3006
3.3.2.3 Operators 3006
3.3.2.4 Character Sets 3019
3.3.2.5 Collation Rules 3020
3.3.2.6 Transactions 3021
3.3.2.7 SQL 3025
3.3.2.7.1 SQL Compatibility Overview 3026
3.3.2.7.2 Keywords 3027
3.3.2.7.3 Identifiers 3028
3.3.2.7.4 DDL 3030
3.3.2.7.5 DML 3055
3.3.2.7.6 DCL 3096
3.3.2.7.7 Other Statements 3096
3.3.2.7.8 Users and Permissions 3099
3.3.2.7.9 System Catalogs and System Views 3106
3.3.3 MYSQL-compatible Mode 3110
3.3.3.1 Data Types 3110
3.3.3.1.1 Numeric Data Types 3111
3.3.3.1.2 Date and Time Data Types 3119
3.3.3.1.3 String Data Types 3134
3.3.3.1.4 Binary Data Types 3139
3.3.3.1.5 JSON Data Type 3142
3.3.3.1.6 Attributes Supported by Data Types 3142
3.3.3.1.7 Data Type Conversion 3142
3.3.3.2 System Functions 3146

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.3.2.1 System Function Compatibility Overview 3146
3.3.3.2.2 Flow Control Functions 3146
3.3.3.2.3 Date and Time Functions 3149
3.3.3.2.4 String Functions 3162
3.3.3.2.5 Forced Conversion Functions 3168
3.3.3.2.6 Encryption Functions 3168
3.3.3.2.7 JSON Functions 3168
3.3.3.2.8 Aggregate Functions 3171
3.3.3.2.9 Numeric Operation Functions 3173
3.3.3.2.10 Other Functions 3174
3.3.3.3 Operators 3174
3.3.3.4 Character Sets 3176
3.3.3.5 Collation Rules 3176
3.3.3.6 SQL 3177
3.3.3.6.1 DDL 3177
3.3.3.6.2 DML 3188
3.3.3.6.3 DCL 3204
3.3.3.7 Drivers 3204
3.3.3.7.1 JDBC API Reference 3204
4 Tool Reference 3206
4.1 Downloading and Installing Tools 3206
4.2 Database Connection Tools 3207
4.2.1 gsql for Connecting to a Database 3208
4.2.1.1 gsql Usage Guide 3208
4.2.1.2 Meta-Command Reference 3229
4.2.1.3 FAQ 3266
4.3 Data Import and Export Tools 3271
4.3.1 Overview 3271
4.3.2 gs_dump for Exporting Database Information 3277
4.3.3 gs_dumpall for Exporting All Database Information 3298
4.3.4 gs_restore for Importing Data 3306
5 Log Reference 3315
5.1 Log Overview 3315
5.2 System Logs 3318
5.3 Operation Logs 3326
5.4 Audit Logs 3332
5.5 WALs 3334
5.6 Memory Logs 3338
6 Database GUC Parameters 3347
6.1 Viewing Parameters 3347
6.2 Setting Parameters 3348

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3 GUC Parameters 3352
6.3.1 GUC Parameter Usage 3352
6.3.2 File Location 3381
6.3.3 Connection and Authentication 3384
6.3.3.1 Connection Settings 3384
6.3.3.2 Security and Authentication 3396
6.3.3.3 Communications Library Parameters 3415
6.3.4 Resource Consumption 3428
6.3.4.1 Memory 3428
6.3.4.2 Disk Space 3457
6.3.4.3 Kernel Resource Usage 3458
6.3.4.4 Cost-based Vacuum Delay 3460
6.3.4.5 Background Writer 3463
6.3.4.6 Asynchronous I/O 3470
6.3.5 Parallel Data Import and Export 3473
6.3.6 Write Ahead Log 3481
6.3.6.1 Settings 3481
6.3.6.2 Checkpoints 3494
6.3.6.3 Log Replay 3500
6.3.6.4 Archiving 3513
6.3.7 HA Replication 3514
6.3.7.1 Sending Server 3514
6.3.7.2 Primary Node 3536
6.3.7.3 Standby Node 3552
6.3.7.4 Logical Replication 3565
6.3.8 Query Planning 3570
6.3.8.1 Optimizer Method Configuration 3570
6.3.8.2 Optimizer Cost Constants 3602
6.3.8.3 Genetic Query Optimizer 3605
6.3.8.4 DSL Rule Engine 3609
6.3.8.5 Other Optimizer Options 3611
6.3.9 Error Reporting and Logging 3680
6.3.9.1 Logging Destination 3680
6.3.9.2 Logging Time 3686
6.3.9.3 Logging Content 3690
6.3.9.4 Using CSV Log Output 3705
6.3.10 Alarm Reporting 3707
6.3.11 Statistics During the Database Running 3710
6.3.11.1 Query and Index Statistics Collector 3710
6.3.11.2 Hotspot Key Statistics 3716
6.3.12 Workload Management 3717
6.3.13 Autovacuum 3743

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3.14 Default Settings of Client Connection 3756
6.3.14.1 Statement Behavior 3756
6.3.14.2 Locale and Formatting 3766
6.3.14.3 Other Default Parameters 3781
6.3.15 Lock Management 3784
6.3.16 Version and Platform Compatibility 3795
6.3.16.1 Compatibility with Earlier Versions 3795
6.3.16.2 Platform and Client Compatibility 3803
6.3.16.3 Product Version of the Cloud Service 3945
6.3.17 Fault Tolerance 3946
6.3.18 Connection Pool Parameters 3949
6.3.19 Cluster Transaction Parameters 3956
6.3.20 Dual-Cluster Replication Parameters 3971
6.3.21 Developer Options 3976
6.3.22 Auditing 3995
6.3.22.1 Audit Switch 3995
6.3.22.2 User and Permission Audit 4000
6.3.22.3 Operation Auditing 4005
6.3.23 Transaction Monitoring 4015
6.3.24 GTM Parameters 4016
6.3.25 Upgrade Parameters 4034
6.3.26 Miscellaneous Parameters 4036
6.3.27 Wait Event 4054
6.3.28 Query 4055
6.3.29 System Performance Snapshot 4070
6.3.30 Black Box Parameters 4082
6.3.31 Security Configuration 4083
6.3.32 HyperLogLog 4095
6.3.33 User-defined Functions 4097
6.3.34 Scheduled Jobs 4098
6.3.35 Thread Pool 4100
6.3.36 Full-Text Search 4110
6.3.37 Backup and Restoration 4112
6.3.38 Undo 4115
6.3.39 DCF Parameters Settings 4117
6.3.40 Flashback 4133
6.3.41 Rollback Parameters 4135
6.3.42 AI Features 4136
6.3.43 Global SysCache Parameters 4151
6.3.44 Read Parameters of the Standby Node in a Distributed System 4152
6.3.45 Restoring Data on the Standby Node 4155
6.3.46 Ledger Database 4156

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3.47 Creating an Index Online....4156
6.3.48 Online DDL Operations....4158
6.3.49 Data Lifecycle Management: OLTP Table Compression....4158
6.3.50 Session-level Transaction Pushdown and Related Parameters....4163
6.3.51 Enhanced TOAST....4165
6.3.52 Global Temporary Table....4165
6.3.53 Single-Node Table....4167
6.3.54 HTAP....4169
6.3.55 Reserved Parameters....4177
6.3.56 Vector Database Parameters....4180
6.4 CM Parameters....4186
6.4.1 CM Agent Parameters....4187
6.4.2 CM Server Parameters....4216
7 System Catalogs and System Views....4251
7.1 Overview of System Catalogs and System Views....4251
7.2 System Catalogs....4252
7.2.1 Partitioned Table....4252
7.2.1.1 PG_PARTITION....4252
7.2.1.2 GS_SUBPARTITION_TEMPLATE....4254
7.2.2 Workload Management....4255
7.2.2.1 GS_ABNORMAL_SQL_INFO....4255
7.2.2.2 GS_ABNORMAL_SQL_WHITE_LIST....4255
7.2.2.3 PG_APP_WORKLOADGROUP_MAPPING....4256
7.2.3 OLTP Table Compression....4256
7.2.3.1 GS_ILM....4256
7.2.3.2 GS_ILM_JOBDETAIL....4257
7.2.3.3 GS_ILM_OBJECT....4257
7.2.3.4 GS_ILM_PARAM....4259
7.2.3.5 GS_ILM_POLICY....4262
7.2.3.6 GS_ILM_TASK....4263
7.2.3.7 GS_ILM_TASKDETAIL....4263
7.2.3.8 GS_ILM_TICKER....4264
7.2.3.9 GS_ILM_TURBO_OBJECT....4264
7.2.4 Encrypted Equality Query....4265
7.2.4.1 GS_CLIENT_GLOBAL_KEYS....4265
7.2.4.2 GS_CLIENT_GLOBAL_KEYS_ARGS....4266
7.2.4.3 GS_COLUMN_KEYS....4266
7.2.4.4 GS_COLUMN_KEYS_ARGS....4267
7.2.4.5 GS_ENCRYPTED_COLUMNS....4267
7.2.4.6 GS_ENCRYPTED_PROC....4268
7.2.5 Communications....4269
7.2.5.1 PGXC_NODE....4269

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.6 Ledger Database....4271
7.2.6.1 GS_GLOBAL_CHAIN....4271
7.2.7 SPM....4272
7.2.7.1 GS_SPM_SQL....4272
7.2.7.2 GS_SPM_PARAM....4272
7.2.7.3 GS_SPM_BASELINE....4273
7.2.7.4 GS_SPM_EVOLUTION....4275
7.2.7.5 GS_SPM_ID_HASH_JOIN....4275
7.2.8 AI....4276
7.2.8.1 GS_MODEL_WAREHOUSE....4276
7.2.8.2 GS_OPT_MODEL....4277
7.2.8.3 GS_ABO_MODEL_STATISTIC....4277
7.2.8.4 GS_ABO_HISTORY....4279
7.2.9 Audit....4279
7.2.9.1 GS_AUDITING_POLICY....4280
7.2.9.2 GS_AUDITING_POLICY_ACCESS....4280
7.2.9.3 GS_AUDITING_POLICY_FILTERS....4281
7.2.9.4 GS_AUDITING_POLICY_PRIVILEGES....4281
7.2.10 User and Permission Management....4282
7.2.10.1 GS_DB_PRIVILEGE....4282
7.2.10.2 PG_DB_ROLE_SETTING....4283
7.2.10.3 PG_DEFAULT_ACL....4283
7.2.10.4 PG_RLSPOLICY....4284
7.2.10.5 PG_SECLABEL....4284
7.2.10.6 PG_SHSECLABEL....4285
7.2.10.7 PG_USER_MAPPING....4285
7.2.10.8 PG_USER_STATUS....4286
7.2.11 Connection and Authentication....4286
7.2.11.1 PG_AUTHID....4287
7.2.11.2 PG_AUTH_HISTORY....4289
7.2.11.3 PG_AUTH_MEMBERS....4290
7.2.12 Dynamic Data Masking....4290
7.2.12.1 GS_MASKING_POLICY....4290
7.2.12.2 GS_MASKING_POLICY_ACTIONS....4291
7.2.12.3 GS_MASKING_POLICY_FILTERS....4291
7.2.13 DATABASE LINK....4292
7.2.13.1 GS_DATABASE_LINK....4292
7.2.14 Materialized Views....4293
7.2.14.1 GS_MATVIEW....4293
7.2.14.2 GS_MATVIEW_DEPENDENCY....4293
7.2.15 Logical Decoding....4294
7.2.15.1 GS_LOGICAL_ATTRDEF....4294

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.15.2 GS_LOGICAL_ATTRIBUTE 4295
7.2.15.3 GS_LOGICAL_AUTHID 4296
7.2.15.4 GS_LOGICAL_CLASS 4296
7.2.15.5 GS_LOGICAL_CONSTRAINT 4298
7.2.15.6 GS_LOGICAL_DATABASE 4300
7.2.15.7 GS_LOGICAL_DICTIONARY 4301
7.2.15.8 GS_LOGICAL_INDEX 4302
7.2.15.9 GS_LOGICAL_NAMESPACE 4303
7.2.15.10 GS_LOGICAL_PARTITION 4304
7.2.15.11 GS_LOGICAL_TYPE 4304
7.2.15.12 GS_SQLAPPLY_SKIP 4306
7.2.15.13 GS_TXN_LSN_TIME 4307
7.2.15.14 GS_LOGICAL_SET 4308
7.2.15.15 GS_LOGICAL_ENUM 4309
7.2.15.16 GS_SQLAPPLY_HISTORY 4310
7.2.16 DSL Rule Engine 4310
7.2.16.1 GS_DSL_REWRITERULE 4310
7.2.17 HTAP 4312
7.2.17.1 GS_IMCV 4312
7.2.18 Other System Catalogs 4313
7.2.18.1 GS_ASP 4313
7.2.18.2 GS_COPY_ERROR_OLD 4317
7.2.18.3 GS_COPY_SUMMARY_LOG 4318
7.2.18.4 GS_COPY_ERROR_LOG 4319
7.2.18.5 GS_DEP_SOURCE 4320
7.2.18.6 GS_DEPENDENCIES 4321
7.2.18.7 GS_DEPENDENCIES_OBJ 4321
7.2.18.8 GS_GLOBAL_CONFIG 4322
7.2.18.9 GS_JOB_ATTRIBUTE 4323
7.2.18.10 GS_JOB_ARGUMENT 4323
7.2.18.11 GS_PACKAGE 4324
7.2.18.12 GS_PLAN_TRACE 4324
7.2.18.13 GS_POLICY_LABEL 4325
7.2.18.14 GS_RECYCLEBIN 4326
7.2.18.15 GS_SECURITY_LABEL 4328
7.2.18.16 GS_SQL_PATCH 4328
7.2.18.17 GS_STATISTIC_EXT_HISTORY 4329
7.2.18.18 GS_STATISTIC_HISTORY 4331
7.2.18.19 GS_SYSTEM_OBJECTS_VERSION 4333
7.2.18.20 GS_TABLESTATS_HISTORY 4334
7.2.18.21 GS_TS_DICT_DEFINITION 4335
7.2.18.22 GS_TXN_SNAPSHOT 4335

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.18.23 GS_UID....4336
7.2.18.24 GS_WORKLOAD_RULE....4336
7.2.18.25 PG_AGGREGATE....4337
7.2.18.26 PG_AM....4339
7.2.18.27 PG_AMOP....4342
7.2.18.28 PG_AMPROC....4343
7.2.18.29 PG_ATTRDEF....4344
7.2.18.30 PG_ATTRIBUTE....4344
7.2.18.31 PG_CAST....4347
7.2.18.32 PG_CLASS....4348
7.2.18.33 PG_COLLATION....4353
7.2.18.34 PG_CONSTRAINT....4354
7.2.18.35 PG_CONVERSION....4357
7.2.18.36 PG_DATABASE....4357
7.2.18.37 PG_DEPEND....4359
7.2.18.38 PG_DESCRIPTION....4360
7.2.18.39 PG_DIRECTORY....4361
7.2.18.40 PG_ENUM....4361
7.2.18.41 PG_EXTENSION....4362
7.2.18.42 PG_EXTENSION_DATA_SOURCE....4363
7.2.18.43 PG_FOREIGN_DATA_WRAPPER....4363
7.2.18.44 PG_FOREIGN_SERVER....4364
7.2.18.45 PG_FOREIGN_TABLE....4365
7.2.18.46 PG_HASHBUCKET....4365
7.2.18.47 PG_INDEX....4366
7.2.18.48 PG_INHERITS....4369
7.2.18.49 PG_JOB....4369
7.2.18.50 PG_JOB_PROC....4371
7.2.18.51 PG_LANGUAGE....4372
7.2.18.52 PG_NAMESPACE....4373
7.2.18.53 PG_OBJECT....4374
7.2.18.54 PG_OPCLASS....4375
7.2.18.55 PG_OPERATOR....4376
7.2.18.56 PG_OPFAMILY....4377
7.2.18.57 PG_PLTEMPLATE....4378
7.2.18.58 PG_PROC....4379
7.2.18.59 PG_RANGE....4383
7.2.18.60 PG_REPLICATION_ORIGIN....4384
7.2.18.61 PG_RESOURCE_POOL....4384
7.2.18.62 PG_REWRITE....4385
7.2.18.63 PG_SET....4386
7.2.18.64 PG_SHDEPEND....4386

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.18.65 PG_SHDESCRIPTION....4388
7.2.18.66 PG_STATISTIC....4388
7.2.18.67 PG_STATISTIC_EXT....4390
7.2.18.68 PG_SYNONYM....4393
7.2.18.69 PG_TABLESPACE....4393
7.2.18.70 PG_TRIGGER....4394
7.2.18.71 PG_TS_CONFIG....4396
7.2.18.72 PG_TS_CONFIG_MAP....4397
7.2.18.73 PG_TS_DICT....4397
7.2.18.74 PG_TS_PARSER....4398
7.2.18.75 PG_TS_TEMPLATE....4398
7.2.18.76 PG_TYPE....4399
7.2.18.77 PGXC_CLASS....4403
7.2.18.78 PGXC_GROUP....4404
7.2.18.79 PGXC_SLICE....4405
7.2.18.80 PLAN_TABLE_DATA....4406
7.2.18.81 STATEMENT_HISTORY....4407
7.2.18.82 STREAMING_STREAM....4418
7.2.18.83 STREAMING_CONT_QUERY....4418
7.3 System Views....4419
7.3.1 Partitioned Tables....4419
7.3.1.1 ADM_IND_PARTITIONS....4419
7.3.1.2 ADM_IND_SUBPARTITIONS....4423
7.3.1.3 ADM_PART_COL_STATISTICS....4426
7.3.1.4 ADM_PART_INDEXES....4428
7.3.1.5 ADM_PART_TABLES....4429
7.3.1.6 ADM_SUBPART_KEY_COLUMNS....4432
7.3.1.7 ADM_TAB_PARTITIONS....4433
7.3.1.8 ADM_TAB_SUBPARTITIONS....4436
7.3.1.9 DB_IND_PARTITIONS....4437
7.3.1.10 DB_IND_SUBPARTITIONS....4441
7.3.1.11 DB_PART_COL_STATISTICS....4444
7.3.1.12 DB_PART_INDEXES....4446
7.3.1.13 DB_PART_KEY_COLUMNS....4447
7.3.1.14 DB_PART_TABLES....4448
7.3.1.15 DB_SUBPART_KEY_COLUMNS....4451
7.3.1.16 DB_TAB_PARTITIONS....4451
7.3.1.17 DB_TAB_SUBPARTITIONS....4455
7.3.1.18 MY_IND_PARTITIONS....4458
7.3.1.19 MY_IND_SUBPARTITIONS....4462
7.3.1.20 MY_PART_COL_STATISTICS....4465
7.3.1.21 MY_PART_INDEXES....4467

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.1.22 MY_PART_KEY_COLUMNS 4468
7.3.1.23 MY_PART_TABLES 4469
7.3.1.24 MY_SUBPART_KEY_COLUMNS 4472
7.3.1.25 MY_TAB_PARTITIONS 4472
7.3.1.26 MY_TAB_SUBPARTITIONS 4476
7.3.1.27 GS_STATIO_ALL_PARTITIONS 4479
7.3.1.28 GS_STAT_XACT_ALL_PARTITIONS 4480
7.3.1.29 GS_STAT_ALL_PARTITIONS 4481
7.3.2 Workload Management 4482
7.3.2.1 GS_WLM_CGROUP_INFO 4482
7.3.2.2 GS_WLM_REBUILD_USER_RESOURCE_POOL 4483
7.3.2.3 GS_WLM_USER_INFO 4483
7.3.2.4 GLOBAL_WLM_RESPOOL_CONCURRENCY_INFO 4484
7.3.2.5 GLOBAL_WLM_RESPOOL_CONNECTION_INFO 4485
7.3.2.6 GLOBAL_WLM_RESPOOL_CPU_INFO 4485
7.3.2.7 GLOBAL_WLM_RESPOOL_IO_INFO 4486
7.3.2.8 GLOBAL_WLM_RESPOOL_MEMORY_INFO 4487
7.3.2.9 PGXC_WLM_REBUILD_USER_RESPOOL 4488
7.3.3 OLTP Table Compression 4489
7.3.3.1 GS_ADM_ILMDATAMOVEMENTPOLICIES 4489
7.3.3.2 GS_ADM_ILMOBJECTS 4490
7.3.3.3 GS_ADM_ILMPOLICIES 4491
7.3.3.4 GS_ADM_ILMEVALUATIONDETAILS 4492
7.3.3.5 GS_ADM_ILMPARAMETERS 4493
7.3.3.6 GS_ADM_ILMRESULTS 4493
7.3.3.7 GS_ADM_ILMTASKS 4494
7.3.3.8 GS_MY_ILMEVALUATIONDETAILS 4494
7.3.3.9 GS_MY_ILMRESULTS 4495
7.3.3.10 GS_MY_ILMTASKS 4496
7.3.3.11 GS_MY_ILMDATAMOVEMENTPOLICIES 4497
7.3.3.12 GS_MY_ILMOBJECTS 4498
7.3.3.13 GS_MY_ILMPOLICIES 4499
7.3.4 Communications 4499
7.3.4.1 COMM_CLIENT_INFO 4499
7.3.4.2 GLOBAL_COMM_CLIENT_INFO 4500
7.3.4.3 GS_COMM_LISTEN_ADDRESS_EXT_INFO 4501
7.3.4.4 GS_GET_LISTEN_ADDRESS_EXT_INFO 4501
7.3.4.5 GS_LIBCOMM_FD_INFO 4502
7.3.4.6 GS_LIBCOMM_MEMORY_INFO 4502
7.3.4.7 GS_STATIC_THREADPOOL_CTRL_STATUS 4503
7.3.4.8 PG_COMM_DELAY 4504
7.3.4.9 PG_COMM_RECV_STREAM 4505

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.4.10 PG_COMM_SEND_STREAM....4506
7.3.4.11 PG_COMM_STATUS....4507
7.3.4.12 PG_GET_INVALID_BACKENDS....4508
7.3.4.13 PG_POOLER_STATUS....4508
7.3.4.14 PGXC_COMM_DELAY....4509
7.3.4.15 PGXC_COMM_RECV_STREAM....4510
7.3.4.16 PGXC_COMM_SEND_STREAM....4511
7.3.4.17 PGXC_COMM_STATUS....4512
7.3.5 Segment-Page Storage....4513
7.3.5.1 GLOBAL_GS_SEG_DATAFILES....4513
7.3.5.2 GLOBAL_GS_SEG_EXTENTS....4514
7.3.5.3 GLOBAL_GS_SEG_SEGMENTS....4516
7.3.5.4 GLOBAL_GS_SEG_SPC_EXTENTS....4517
7.3.5.5 GLOBAL_GS_SEG_SPC_SEGMENTS....4519
7.3.5.6 GS_SEG_DATAFILES....4520
7.3.5.7 GS_SEG_DATAFILE_LAYOUT....4522
7.3.5.8 GS_SEG_EXTENTS....4523
7.3.5.9 GS_SEG_SEGMENTS....4524
7.3.5.10 GS_SEG_SEGMENT_LAYOUT....4526
7.3.5.11 GS_SEG_SPC_EXTENTS....4526
7.3.5.12 GS_SEG_SPC_SEGMENTS....4528
7.3.5.13 GS_SEG_SPC_REMAIN_EXTENTS....4529
7.3.5.14 GS_SEG_SPC_REMAIN_SEGMENTS....4531
7.3.6 SPM....4532
7.3.6.1 GS_SPM_SQL_BASELINE....4532
7.3.6.2 GS_SPM_SQL_PARAM....4534
7.3.6.3 GS_SPM_SQL_EVOLUTION....4535
7.3.6.4 GS_SPM_SYS_BASELINE....4536
7.3.7 Audit....4537
7.3.7.1 ADM_AUDIT_OBJECT....4537
7.3.7.2 ADM_AUDIT_SESSION....4539
7.3.7.3 ADM_AUDIT_STATEMENT....4541
7.3.7.4 ADM_AUDIT_TRAIL....4543
7.3.7.5 GS_AUDITING....4546
7.3.7.6 GS_AUDITING_ACCESS....4546
7.3.7.7 GS_AUDITING_PRIVILEGE....4547
7.3.8 User and Permission Management....4548
7.3.8.1 ADM_COL_PRIVS....4548
7.3.8.2 ADM_ROLE_PRIVS....4548
7.3.8.3 ADM_ROLES....4549
7.3.8.4 ADM_SYS_PRIVS....4550
7.3.8.5 ADM_TAB_PRIVS....4551

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.8.6 ADM_USERS....4551
7.3.8.7 DB_COL_PRIVS....4553
7.3.8.8 DB_DIRECTORIES....4554
7.3.8.9 DB_TAB_PRIVS....4555
7.3.8.10 DB_USERS....4555
7.3.8.11 GS_DB_PRIVILEGES....4556
7.3.8.12 GS_LABELS....4556
7.3.8.13 MY_COL_PRIVS....4557
7.3.8.14 MY_ROLE_PRIVS....4557
7.3.8.15 MY_SYS_PRIVS....4558
7.3.8.16 PG_RLSPOLICIES....4559
7.3.8.17 GS_RLSPOLICIES....4560
7.3.8.18 PG_ROLES....4561
7.3.8.19 GS_ROLES....4563
7.3.8.20 PG_SECLABELS....4566
7.3.8.21 GS_SECLABELS....4567
7.3.8.22 PG_SHADOW....4568
7.3.8.23 GS_SHADOW....4569
7.3.8.24 PG_USER....4571
7.3.8.25 GS_USER....4573
7.3.8.26 PG_USER_MAPPINGS....4575
7.3.8.27 ROLE_ROLE_PRIVS....4575
7.3.8.28 ROLE_SYS_PRIVS....4576
7.3.8.29 ROLE_TAB_PRIVS....4577
7.3.9 Dynamic Data Masking....4577
7.3.9.1 GS_MASKING....4577
7.3.10 Transparent Data Encryption (TDE)....4578
7.3.10.1 PG_TDE_INFO....4578
7.3.11 DATABASE LINK....4578
7.3.11.1 GS_DB_LINKS....4578
7.3.11.2 V$DBLINK....4579
7.3.12 Materialized Views....4580
7.3.12.1 GS_MATVIEWS....4580
7.3.13 Other System Views....4581
7.3.13.1 ADM_ARGUMENTS....4581
7.3.13.2 ADM_COL_COMMENTS....4583
7.3.13.3 ADM_COLL_TYPES....4584
7.3.13.4 ADM_CONS_COLUMNS....4585
7.3.13.5 ADM_CONSTRAINTS....4585
7.3.13.6 ADM_DATA_FILES....4587
7.3.13.7 ADM_DEPENDENCIES....4587
7.3.13.8 ADM_DIRECTORIES....4588

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.9 ADM_HIST_SNAPSHOT....4588
7.3.13.10 ADM_HIST_SQL_PLAN....4589
7.3.13.11 ADM_HIST_SQLSTAT....4592
7.3.13.12 ADM_HIST_SQLTEXT....4593
7.3.13.13 ADM_IND_COLUMNS....4593
7.3.13.14 ADM_IND_EXPRESSIONS....4594
7.3.13.15 ADM_INDEXES....4594
7.3.13.16 ADM_OBJECTS....4599
7.3.13.17 ADM_PROCEDURES....4600
7.3.13.18 ADM_SCHEDULER_JOB_ARGS....4602
7.3.13.19 ADM_SCHEDULER_JOBS....4602
7.3.13.20 ADM_SCHEDULER_PROGRAM_ARGS....4607
7.3.13.21 ADM_SCHEDULER_PROGRAMS....4607
7.3.13.22 ADM_SCHEDULER_RUNNING_JOBS....4609
7.3.13.23 ADM_SEGMENTS....4610
7.3.13.24 ADM_SEQUENCES....4612
7.3.13.25 ADM_SOURCE....4613
7.3.13.26 ADM_SYNONYMS....4613
7.3.13.27 ADM_TAB_COL_STATISTICS....4614
7.3.13.28 ADM_TAB_COLS....4616
7.3.13.29 ADM_TAB_COLUMNS....4619
7.3.13.30 ADM_TAB_COMMENTS....4622
7.3.13.31 ADM_TAB_HISTOGRAMS....4623
7.3.13.32 ADM_TAB_STATISTICS....4623
7.3.13.33 ADM_TAB_STATS_HISTORY....4625
7.3.13.34 ADM_TABLES....4626
7.3.13.35 ADM_TABLESPACES....4632
7.3.13.36 ADM_TRIGGERS....4635
7.3.13.37 ADM_TYPE_ATTRS....4636
7.3.13.38 ADM_TYPES....4637
7.3.13.39 ADM_VIEWS....4638
7.3.13.40 DB_ARGUMENTS....4640
7.3.13.41 DB_ALL_TABLES....4642
7.3.13.42 DB_COL_COMMENTS....4643
7.3.13.43 DB_COLL_TYPES....4643
7.3.13.44 DB_CONS_COLUMNS....4644
7.3.13.45 DB_CONSTRAINTS....4644
7.3.13.46 DB_DEPENDENCIES....4645
7.3.13.47 DB_IND_COLUMNS....4646
7.3.13.48 DB_IND_EXPRESSIONS....4646
7.3.13.49 DB_INDEXES....4647
7.3.13.50 DB_OBJECTS....4651

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.51 DB_PROCEDURES....4653
7.3.13.52 DB_SCHEDULER_JOB_ARGS....4653
7.3.13.53 DB_SCHEDULER_PROGRAM_ARGS....4654
7.3.13.54 DB_SEQUENCES....4655
7.3.13.55 DB_SOURCE....4656
7.3.13.56 DB_SYNONYMS....4656
7.3.13.57 DB_TAB_COL_STATISTICS....4657
7.3.13.58 DB_TAB_COLUMNS....4659
7.3.13.59 DB_TAB_COMMENTS....4662
7.3.13.60 DB_TAB_HISTOGRAMS....4662
7.3.13.61 DB_TAB_MODIFICATIONS....4663
7.3.13.62 DB_TAB_STATS_HISTORY....4664
7.3.13.63 DB_TABLES....4665
7.3.13.64 DB_TRIGGERS....4671
7.3.13.65 DB_TYPES....4672
7.3.13.66 DB_VIEWS....4672
7.3.13.67 DICT....4674
7.3.13.68 DICTIONARY....4674
7.3.13.69 DUAL....4675
7.3.13.70 DV_SESSIONS....4675
7.3.13.71 DV_SESSION_LONGOPS....4676
7.3.13.72 GET_GLOBAL_PREPARED_XACTS....4676
7.3.13.73 GLOBAL_BAD_BLOCK_INFO....4676
7.3.13.74 GLOBAL_CLEAR_BAD_BLOCK_INFO....4677
7.3.13.75 GLOBAL_SQL_PATCH....4678
7.3.13.76 GLOBAL_STAT_HOTKEYS_INFO....4679
7.3.13.77 GLOBAL_WAL_SENDER_STATUS....4679
7.3.13.78 GS_ALL_CONTROL_GROUP_INFO....4683
7.3.13.79 GS_ALL_PREPARED_STATEMENTS....4684
7.3.13.80 GS_AUTONOMOUS_TRANSACTION_DETAIL....4685
7.3.13.81 GS_BUCKET_LOCKS....4685
7.3.13.82 GS_GET_CONTROL_GROUP_INFO....4686
7.3.13.83 GS_GLC_MEMORY_DETAIL....4688
7.3.13.84 GS_GLOBAL_ARCHIVE_STATUS....4689
7.3.13.85 GS_GSC_MEMORY_DETAIL....4689
7.3.13.86 GS_GTT_ATTACHED_PIDS....4690
7.3.13.87 GS_GLOBAL_AUTONOMOUS_TRANSACTION_DETAIL....4690
7.3.13.88 GS_GLOBAL_GTT_ATTACHED_PIDS....4691
7.3.13.89 GS_GLOBAL_DIST_GTT_REMAIN_SESSIONS....4691
7.3.13.90 GS_LOGICALSTANDBY_SKIP....4692
7.3.13.91 GS_LOGICALSTANDBY_SKIP_TRANSACTION....4692
7.3.13.92 GS_LSC_MEMORY_DETAIL....4693

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.93 GS_MY_PLAN_TRACE 4693
7.3.13.94 GS_SESSION_ALL_SETTINGS 4694
7.3.13.95 GS_SQL_COUNT 4694
7.3.13.96 GS_SYSTEM_OBJECT_VERSION_DETAILS 4697
7.3.13.97 GS_THRDSTREAM_WORKER_INFO 4698
7.3.13.98 GS_TOTAL_NODEGROUP_MEMORY_DETAIL 4699
7.3.13.99 GS_WORKLOAD_RULE_STAT 4700
7.3.13.100 GV_INSTANCE 4702
7.3.13.101 GV_SESSION 4703
7.3.13.102 MPP_TABLES 4709
7.3.13.103 MY_AUDIT_TRAIL 4709
7.3.13.104 MY_COL_COMMENTS 4712
7.3.13.105 MY_COLL_TYPES 4713
7.3.13.106 MY_CONS_COLUMNS 4714
7.3.13.107 MY_CONSTRAINTS 4714
7.3.13.108 MY_DEPENDENCIES 4715
7.3.13.109 MY_IND_COLUMNS 4715
7.3.13.110 MY_IND_EXPRESSIONS 4716
7.3.13.111 MY_INDEXES 4717
7.3.13.112 MY_JOBS 4721
7.3.13.113 MY_OBJECTS 4722
7.3.13.114 MY_PROCEDURES 4724
7.3.13.115 MY_SCHEDULER_JOB_ARGS 4725
7.3.13.116 MY_SCHEDULER_JOBS 4726
7.3.13.117 MY_SCHEDULER_PROGRAM_ARGS 4730
7.3.13.118 MY_SEQUENCES 4730
7.3.13.119 MY_SOURCE 4732
7.3.13.120 MY_SYNONYMS 4732
7.3.13.121 MY_TAB_COL_STATISTICS 4733
7.3.13.122 MY_TAB_COLUMNS 4734
7.3.13.123 MY_TAB_COMMENTS 4737
7.3.13.124 MY_TAB_HISTOGRAMS 4738
7.3.13.125 MY_TAB_MODIFICATIONS 4738
7.3.13.126 MY_TAB_STATS_HISTORY 4739
7.3.13.127 MY_TAB_STATISTICS 4740
7.3.13.128 MY_TABLES 4741
7.3.13.129 MY_TABLESPACES 4747
7.3.13.130 MY_TRIGGERS 4749
7.3.13.131 MY_TYPE_ATTRS 4751
7.3.13.132 MY_TYPES 4752
7.3.13.133 MY_VIEWS 4753
7.3.13.134 NLS_DATABASE_PARAMETERS 4754

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.135 NLS_INSTANCE_PARAMETERS....4755
7.3.13.136 PG_AVAILABLE_EXTENSION_VERSIONS....4755
7.3.13.137 PG_AVAILABLE_EXTENSIONS....4756
7.3.13.138 PG_CONTROL_GROUP_CONFIG....4756
7.3.13.139 PG_CURSORS....4757
7.3.13.140 PG_EXT_STATS....4757
7.3.13.141 PG_GET_SENDERS_CATCHUP_TIME....4760
7.3.13.142 PG_GROUP....4761
7.3.13.143 PG_GTT_STATS....4761
7.3.13.144 PG_GTT_RELSTATS....4762
7.3.13.145 PG_GTT_ATTACHED_PIDS....4763
7.3.13.146 PG_INDEXES....4763
7.3.13.147 PG_LOCKS....4764
7.3.13.148 PG_NODE_ENV....4766
7.3.13.149 PG_OS_THREADS....4766
7.3.13.150 PG_PREPARED_STATEMENTS....4767
7.3.13.151 PG_PREPARED_XACTS....4768
7.3.13.152 PG_REPLICATION_SLOTS....4768
7.3.13.153 PG_RULES....4770
7.3.13.154 PG_RUNNING_XACTS....4770
7.3.13.155 PG_SESSION_IOSTAT....4771
7.3.13.156 PG_SETTINGS....4772
7.3.13.157 PG_SHARED_MEMORY_DETAIL....4773
7.3.13.158 PG_STATS....4774
7.3.13.159 PG_STAT_ACTIVITY....4777
7.3.13.160 PG_STAT_ACTIVITY_NG....4781
7.3.13.161 PG_STAT_ALL_INDEXES....4784
7.3.13.162 PG_STAT_ALL_TABLES....4785
7.3.13.163 PG_STAT_BAD_BLOCK....4786
7.3.13.164 PG_STAT_BGWRITER....4787
7.3.13.165 PG_STAT_DATABASE....4788
7.3.13.166 PG_STAT_DATABASE_CONFLICTS....4789
7.3.13.167 PG_STAT_REPLICATION....4790
7.3.13.168 PG_STAT_SYS_INDEXES....4792
7.3.13.169 PG_STAT_SYS_TABLES....4793
7.3.13.170 PG_STAT_USER_FUNCTIONS....4794
7.3.13.171 PG_STAT_USER_INDEXES....4795
7.3.13.172 PG_STAT_USER_TABLES....4795
7.3.13.173 PG_STAT_XACT_ALL_TABLES....4797
7.3.13.174 PG_STAT_XACT_SYS_TABLES....4797
7.3.13.175 PG_STAT_XACT_USER_FUNCTIONS....4798
7.3.13.176 PG_STAT_XACT_USER_TABLES....4799

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.177 PG_STATIO_ALL_INDEXES....4799
7.3.13.178 PG_STATIO_ALL_SEQUENCES....4800
7.3.13.179 PG_STATIO_ALL_TABLES....4800
7.3.13.180 PG_STATIO_SYS_INDEXES....4801
7.3.13.181 PG_STATIO_SYS_SEQUENCES....4801
7.3.13.182 PG_STATIO_SYS_TABLES....4802
7.3.13.183 PG_STATIO_USER_INDEXES....4803
7.3.13.184 PG_STATIO_USER_SEQUENCES....4803
7.3.13.185 PG_STATIO_USER_TABLES....4804
7.3.13.186 PG_TABLES....4804
7.3.13.187 PG_THREAD_WAIT_STATUS....4805
7.3.13.188 PG_TIMEZONE_ABBREVS....4896
7.3.13.189 PG_TIMEZONE_NAMES....4896
7.3.13.190 PG_TOTAL_MEMORY_DETAIL....4897
7.3.13.191 PG_TOTAL_USER_RESOURCE_INFO....4897
7.3.13.192 PG_TOTAL_USER_RESOURCE_INFO_OID....4899
7.3.13.193 PG_VARIABLE_INFO....4900
7.3.13.194 PG_VIEWS....4901
7.3.13.195 PGXC_COPY_ERROR_LOG....4901
7.3.13.196 PGXC_GET_STAT_ALL_TABLES....4903
7.3.13.197 PGXC_GET_TABLE_SKEWNESS....4904
7.3.13.198 PGXC_NODE_ENV....4905
7.3.13.199 PGXC_OS_THREADS....4905
7.3.13.200 PGXC_PREPARED_XACTS....4906
7.3.13.201 PGXC_RUNNING_XACTS....4906
7.3.13.202 PGXC_STAT_ACTIVITY....4907
7.3.13.203 PGXC_STAT_BAD_BLOCK....4910
7.3.13.204 PGXC_SQL_COUNT....4911
7.3.13.205 PGXC_THREAD_WAIT_STATUS....4911
7.3.13.206 PGXC_TOTAL_MEMORY_DETAIL....4913
7.3.13.207 PGXC_VARIABLE_INFO....4915
7.3.13.208 PLAN_TABLE....4916
7.3.13.209 PV_FILE_STAT....4918
7.3.13.210 PV_INSTANCE_TIME....4919
7.3.13.211 PV_OS_RUN_INFO....4920
7.3.13.212 PV_REDO_STAT....4920
7.3.13.213 PV_SESSION_MEMORY....4921
7.3.13.214 PV_SESSION_MEMORY_CONTEXT....4921
7.3.13.215 PV_SESSION_MEMORY_DETAIL....4922
7.3.13.216 PV_SESSION_STAT....4924
7.3.13.217 PV_SESSION_TIME....4924
7.3.13.218 PV_THREAD_MEMORY_CONTEXT....4925

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.219 PV_TOTAL_MEMORY_DETAIL....4926
7.3.13.220 SYS_DUMMY....4928
7.3.13.221 V_INSTANCE....4929
7.3.13.222 V_MYSTAT....4930
7.3.13.223 V_SESSION....4930
7.3.13.224 V$GLOBAL_OPEN_CURSOR....4936
7.3.13.225 V$GLOBAL_TRANSACTION....4937
7.3.13.226 V$LOCK....4938
7.3.13.227 V$NLS_PARAMETERS....4939
7.3.13.228 V$OPEN_CURSOR....4940
7.3.13.229 V$SESSION_WAIT....4941
7.3.13.230 V$SYSSTAT....4943
7.3.13.231 V$SYSTEM_EVENT....4943
7.3.13.232 V$VERSION....4944
7.3.14 Deprecated System Views/System Catalogs....4945
8 Schemas....4946
8.1 Information Schema....4948
8.1.1 INFORMATION_SCHEMA._PG_FOREIGN_DATA_WRAPPERS....4949
8.1.2 INFORMATION_SCHEMA._PG_FOREIGN_SERVERS....4949
8.1.3 INFORMATION_SCHEMA._PG_FOREIGN_TABLE_COLUMNS....4950
8.1.4 INFORMATION_SCHEMA._PG_FOREIGN_TABLES....4951
8.1.5 INFORMATION_SCHEMA._PG_USER_MAPPINGS....4951
8.1.6 INFORMATION_SCHEMA.INFORMATION_SCHEMA_CATALOG_NAME....4952
8.1.7 INFORMATION_SCHEMA.TRIGGERS....4952
8.2 DBE_PERF Schema....4954
8.2.1 OS....4954
8.2.1.1 DBE_PERF.OS_RUNTIME....4954
8.2.1.2 DBE_PERF.GLOBAL_OS_RUNTIME....4955
8.2.1.3 DBE_PERF.OS_THREADS....4955
8.2.1.4 DBE_PERF.GLOBAL_OS_THREADS....4956
8.2.1.5 DBE_PERF.PERF_QUERY....4956
8.2.1.6 DBE_PERF.SHAKING_COLLECT_STATUS....4957
8.2.2 Instance....4957
8.2.2.1 DBE_PERF.INSTANCE_TIME....4957
8.2.2.2 DBE_PERF.GLOBAL_INSTANCE_TIME....4958
8.2.3 Memory....4959
8.2.3.1 DBE_PERF.MEMORY_NODE_DETAIL....4959
8.2.3.2 DBE_PERF.GLOBAL_MEMORY_NODE_DETAIL....4961
8.2.3.3 DBE_PERF.MEMORY_NODE_NG_DETAIL....4963
8.2.3.4 DBE_PERF.SHARED_MEMORY_DETAIL....4964
8.2.3.5 DBE_PERF.GLOBAL_SHARED_MEMORY_DETAIL....4965
8.2.3.6 DBE_PERF.TRACK_MEMORY_CONTEXT_DETAIL....4965

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.4 File ... 4965
8.2.4.1 DBE_PERF.FILE_IOSTAT ... 4966
8.2.4.2 DBE_PERF.SUMMARY_FILE_IOSTAT ... 4966
8.2.4.3 DBE_PERF.GLOBAL_FILE_IOSTAT ... 4967
8.2.4.4 DBE_PERF.FILE_REDO_IOSTAT ... 4968
8.2.4.5 DBE_PERF.SUMMARY_FILE_REDO_IOSTAT ... 4969
8.2.4.6 DBE_PERF.GLOBAL_FILE_REDO_IOSTAT ... 4969
8.2.4.7 DBE_PERF.LOCAL_REL_IOSTAT ... 4970
8.2.4.8 DBE_PERF.SUMMARY_REL_IOSTAT ... 4971
8.2.4.9 DBE_PERF.GLOBAL_REL_IOSTAT ... 4971
8.2.5 Object ... 4971
8.2.5.1 DBE_PERF.STAT_USER_TABLES ... 4972
8.2.5.2 DBE_PERF.SUMMARY_STAT_USER_TABLES ... 4973
8.2.5.3 DBE_PERF.GLOBAL_STAT_USER_TABLES ... 4974
8.2.5.4 DBE_PERF.STAT_USER_INDEXES ... 4976
8.2.5.5 DBE_PERF.SUMMARY_STAT_USER_INDEXES ... 4976
8.2.5.6 DBE_PERF.GLOBAL_STAT_USER_INDEXES ... 4977
8.2.5.7 DBE_PERF.STAT_SYS_TABLES ... 4978
8.2.5.8 DBE_PERF.SUMMARY_STAT_SYS_TABLES ... 4979
8.2.5.9 DBE_PERF.GLOBAL_STAT_SYS_TABLES ... 4980
8.2.5.10 DBE_PERF.STAT_SYS_INDEXES ... 4982
8.2.5.11 DBE_PERF.SUMMARY_STAT_SYS_INDEXES ... 4982
8.2.5.12 DBE_PERF.GLOBAL_STAT_SYS_INDEXES ... 4983
8.2.5.13 DBE_PERF.STAT_ALL_TABLES ... 4984
8.2.5.14 DBE_PERF.SUMMARY_STAT_ALL_TABLES ... 4985
8.2.5.15 DBE_PERF.GLOBAL_STAT_ALL_TABLES ... 4986
8.2.5.16 DBE_PERF.STAT_ALL_INDEXES ... 4988
8.2.5.17 DBE_PERF.SUMMARY_STAT_ALL_INDEXES ... 4988
8.2.5.18 DBE_PERF.GLOBAL_STAT_ALL_INDEXES ... 4989
8.2.5.19 DBE_PERF.STAT_DATABASE ... 4990
8.2.5.20 DBE_PERF.SUMMARY_STAT_DATABASE ... 4991
8.2.5.21 DBE_PERF.GLOBAL_STAT_DATABASE ... 4993
8.2.5.22 DBE_PERF.STAT_DATABASE_CONFLICTS ... 4994
8.2.5.23 DBE_PERF.SUMMARY_STAT_DATABASE_CONFLICTS ... 4995
8.2.5.24 DBE_PERF.GLOBAL_STAT_DATABASE_CONFLICTS ... 4995
8.2.5.25 DBE_PERF.STAT_XACT_ALL_TABLES ... 4996
8.2.5.26 DBE_PERF.SUMMARY_STAT_XACT_ALL_TABLES ... 4996
8.2.5.27 DBE_PERF.GLOBAL_STAT_XACT_ALL_TABLES ... 4997
8.2.5.28 DBE_PERF.STAT_XACT_SYS_TABLES ... 4998
8.2.5.29 DBE_PERF.SUMMARY_STAT_XACT_SYS_TABLES ... 4998
8.2.5.30 DBE_PERF.GLOBAL_STAT_XACT_SYS_TABLES ... 4999
8.2.5.31 DBE_PERF.STAT_XACT_USER_TABLES ... 5000

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.5.32 DBE_PERF.SUMMARY_STAT_XACT_USER_TABLES 5000
8.2.5.33 DBE_PERF.GLOBAL_STAT_XACT_USER_TABLES 5001
8.2.5.34 DBE_PERF.STAT_XACT_USER_FUNCTIONS 5002
8.2.5.35 DBE_PERF.SUMMARY_STAT_XACT_USER_FUNCTIONS 5002
8.2.5.36 DBE_PERF.GLOBAL_STAT_XACT_USER_FUNCTIONS 5002
8.2.5.37 DBE_PERF.STAT_BAD_BLOCK 5003
8.2.5.38 DBE_PERF.SUMMARY_STAT_BAD_BLOCK 5003
8.2.5.39 DBE_PERF.GLOBAL_STAT_BAD_BLOCK 5004
8.2.5.40 DBE_PERF.STAT_USER_FUNCTIONS 5005
8.2.5.41 DBE_PERF.SUMMARY_STAT_USER_FUNCTIONS 5005
8.2.5.42 DBE_PERF.GLOBAL_STAT_USER_FUNCTIONS 5006
8.2.6 Workload 5006
8.2.6.1 DBE_PERF.WORKLOAD_SQL_COUNT 5006
8.2.6.2 DBE_PERF.SUMMARY_WORKLOAD_SQL_COUNT 5008
8.2.6.3 DBE_PERF.WORKLOAD_TRANSACTION 5010
8.2.6.4 DBE_PERF.SUMMARY_WORKLOAD_TRANSACTION 5011
8.2.6.5 DBE_PERF.GLOBAL_WORKLOAD_TRANSACTION 5013
8.2.6.6 DBE_PERF.WORKLOAD_SQL_ELAPSE_TIME 5014
8.2.6.7 DBE_PERF.SUMMARY_WORKLOAD_SQL_ELAPSE_TIME 5015
8.2.6.8 DBE_PERF.USER_TRANSACTION 5016
8.2.6.9 DBE_PERF.GLOBAL_USER_TRANSACTION 5018
8.2.7 Session and Thread 5019
8.2.7.1 DBE_PERF.SESSION_STAT 5019
8.2.7.2 DBE_PERF.GLOBAL_SESSION_STAT 5019
8.2.7.3 DBE_PERF.SESSION_TIME 5020
8.2.7.4 DBE_PERF.GLOBAL_SESSION_TIME 5020
8.2.7.5 DBE_PERF.SESSION_MEMORY 5020
8.2.7.6 DBE_PERF.GLOBAL_SESSION_MEMORY 5021
8.2.7.7 DBE_PERF.SESSION_MEMORY_DETAIL 5021
8.2.7.8 DBE_PERF.GLOBAL_SESSION_MEMORY_DETAIL 5022
8.2.7.9 DBE_PERF.SESSION_STAT_ACTIVITY 5022
8.2.7.10 DBE_PERF.GLOBAL_SESSION_STAT_ACTIVITY 5025
8.2.7.11 DBE_PERF.THREAD_WAIT_STATUS 5028
8.2.7.12 DBE_PERF.GLOBAL_THREAD_WAIT_STATUS 5029
8.2.7.13 DBE_PERF.LOCAL_THREADPOOL_STATUS 5030
8.2.7.14 DBE_PERF.GLOBAL_THREADPOOL_STATUS 5031
8.2.7.15 DBE_PERF.SESSION_CPU_RUNTIME 5032
8.2.7.16 DBE_PERF.SESSION_MEMORY_RUNTIME 5032
8.2.7.17 DBE_PERF.STATEMENT_IOSTAT_COMPLEX_RUNTIME 5033
8.2.7.18 DBE_PERF.LOCAL_ACTIVE_SESSION 5034
8.2.7.19 DBE_PERF.GLOBAL_ACTIVE_SESSION 5036
8.2.8 Transaction 5038

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.8.1 DBE_PERF.TRANSACTIONS_RUNNING_XACTS....5039
8.2.8.2 DBE_PERF.SUMMARY_TRANSACTIONS_RUNNING_XACTS....5039
8.2.8.3 DBE_PERF.GLOBAL_TRANSACTIONS_RUNNING_XACTS....5040
8.2.8.4 DBE_PERF.TRANSACTIONS_PREPARED_XACTS....5041
8.2.8.5 DBE_PERF.SUMMARY_TRANSACTIONS_PREPARED_XACTS....5041
8.2.8.6 DBE_PERF.GLOBAL_TRANSACTIONS_PREPARED_XACTS....5041
8.2.9 Query....5042
8.2.9.1 DBE_PERF.STATEMENT....5042
8.2.9.2 DBE_PERF.SUMMARY_STATEMENT....5048
8.2.9.3 DBE_PERF.STATEMENT_COUNT....5053
8.2.9.4 DBE_PERF.GLOBAL_STATEMENT_COUNT....5056
8.2.9.5 DBE_PERF.SUMMARY_STATEMENT_COUNT....5058
8.2.9.6 DBE_PERF.STATEMENT_RESPONSETIME_PERCENTILE....5060
8.2.9.7 DBE_PERF.STATEMENT_HISTORY....5061
8.2.9.8 DBE_PERF.GLOBAL_STATEMENT_COMPLEX_HISTORY....5068
8.2.9.9 DBE_PERF.GLOBAL_STATEMENT_COMPLEX_HISTORY_TABLE....5073
8.2.9.10 DBE_PERF.STATEMENT_WLMSTAT_COMPLEX_RUNTIME....5073
8.2.9.11 DBE_PERF.SESSION_SQL_MEMORY....5075
8.2.9.12 DBE_PERF.GLOBAL_SESSION_SQL_MEMORY....5076
8.2.9.13 DBE_PERF.GS_PLANCACHE_STAT....5076
8.2.9.14 DBE_PERF.GS_DIST_PLANCACHE_STAT....5077
8.2.10 Cache and I/O....5078
8.2.10.1 DBE_PERF.STATIO_USER_TABLES....5078
8.2.10.2 DBE_PERF.SUMMARY_STATIO_USER_TABLES....5079
8.2.10.3 DBE_PERF.GLOBAL_STATIO_USER_TABLES....5080
8.2.10.4 DBE_PERF.STATIO_USER_INDEXES....5080
8.2.10.5 DBE_PERF.SUMMARY_STATIO_USER_INDEXES....5081
8.2.10.6 DBE_PERF.GLOBAL_STATIO_USER_INDEXES....5081
8.2.10.7 DBE_PERF.STATIO_USER_SEQUENCES....5082
8.2.10.8 DBE_PERF.SUMMARY_STATIO_USER_SEQUENCES....5082
8.2.10.9 DBE_PERF.GLOBAL_STATIO_USER_SEQUENCES....5083
8.2.10.10 DBE_PERF.STATIO_SYS_TABLES....5083
8.2.10.11 DBE_PERF.SUMMARY_STATIO_SYS_TABLES....5084
8.2.10.12 DBE_PERF.GLOBAL_STATIO_SYS_TABLES....5085
8.2.10.13 DBE_PERF.STATIO_SYS_INDEXES....5085
8.2.10.14 DBE_PERF.SUMMARY_STATIO_SYS_INDEXES....5086
8.2.10.15 DBE_PERF.GLOBAL_STATIO_SYS_INDEXES....5086
8.2.10.16 DBE_PERF.STATIO_SYS_SEQUENCES....5087
8.2.10.17 DBE_PERF.SUMMARY_STATIO_SYS_SEQUENCES....5087
8.2.10.18 DBE_PERF.GLOBAL_STATIO_SYS_SEQUENCES....5088
8.2.10.19 DBE_PERF.STATIO_ALL_TABLES....5088
8.2.10.20 DBE_PERF.SUMMARY_STATIO_ALL_TABLES....5089

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.10.21 DBE_PERF.GLOBAL_STATIO_ALL_TABLES 5090
8.2.10.22 DBE_PERF.STATIO_ALL_INDEXES 5091
8.2.10.23 DBE_PERF.SUMMARY_STATIO_ALL_INDEXES 5091
8.2.10.24 DBE_PERF.GLOBAL_STATIO_ALL_INDEXES 5092
8.2.10.25 DBE_PERF.STATIO_ALL_SEQUENCES 5092
8.2.10.26 DBE_PERF.SUMMARY_STATIO_ALL_SEQUENCES 5093
8.2.10.27 DBE_PERF.GLOBAL_STATIO_ALL_SEQUENCES 5093
8.2.11 Communications Library 5094
8.2.11.1 DBE_PERF.COMM_DELAY 5094
8.2.11.2 DBE_PERF.GLOBAL_COMM_DELAY 5094
8.2.11.3 DBE_PERF.COMM_RECV_STREAM 5095
8.2.11.4 DBE_PERF.GLOBAL_COMM_RECV_STREAM 5096
8.2.11.5 DBE_PERF.COMM_SEND_STREAM 5097
8.2.11.6 DBE_PERF.GLOBAL_COMM_SEND_STREAM 5098
8.2.11.7 DBE_PERF.COMM_STATUS 5098
8.2.11.8 DBE_PERF.GLOBAL_COMM_STATUS 5099
8.2.11.9 DBE_PERF.GLOBAL_COMM_CHECK_CONNECTION_STATUS 5100
8.2.12 Utility 5101
8.2.12.1 DBE_PERF.REPLICATION_STAT 5101
8.2.12.2 DBE_PERF.GLOBAL_REPLICATION_STAT 5102
8.2.12.3 DBE_PERF.REPLICATION_SLOTS 5103
8.2.12.4 DBE_PERF.GLOBAL_REPLICATION_SLOTS 5104
8.2.12.5 DBE_PERF.BGWRITER_STAT 5105
8.2.12.6 DBE_PERF.GLOBAL_BGWRITER_STAT 5106
8.2.12.7 DBE_PERF.POOLER_STATUS 5107
8.2.12.8 DBE_PERF.POOLER_AGENT 5108
8.2.12.9 DBE_PERF.GLOBAL_CKPT_STATUS 5109
8.2.12.10 DBE_PERF.GLOBAL_DOUBLE_WRITE_STATUS 5109
8.2.12.11 DBE_PERF.GLOBAL_PAGEWRITER_STATUS 5110
8.2.12.12 DBE_PERF.GLOBAL_POOLER_STATUS 5111
8.2.12.13 DBE_PERF.GLOBAL_RECORD_RESET_TIME 5112
8.2.12.14 DBE_PERF.GLOBAL_REDO_STATUS 5112
8.2.12.15 DBE_PERF.GLOBAL_RECOVERY_STATUS 5114
8.2.12.16 DBE_PERF.CLASS_VITAL_INFO 5114
8.2.12.17 DBE_PERF.USER_LOGIN 5115
8.2.12.18 DBE_PERF.SUMMARY_USER_LOGIN 5115
8.2.12.19 DBE_PERF.GLOBAL_GET_BGWRITER_STATUS 5116
8.2.12.20 DBE_PERF.GLOBAL_SINGLE_FLUSH_DW_STATUS 5116
8.2.12.21 DBE_PERF.GLOBAL_CANDIDATE_STATUS 5117
8.2.12.22 DBE_PERF.PARALLEL_DECODE_STATUS 5118
8.2.12.23 DBE_PERF.GLOBAL_PARALLEL_DECODE_STATUS 5118
8.2.12.24 DBE_PERF.PARALLEL_DECODE_THREAD_INFO 5119

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.12.25 DBE_PERF.GLOBAL_PARALLEL_DECODE_THREAD_INFO....5120
8.2.12.26 DBE_PERF.GLOBAL_ADIO_COMPLETER_STATUS....5120
8.2.12.27 DBE_PERF.GLOBAL_AIO_SLOT_USAGE_STATUS....5121
8.2.12.28 DBE_PERF.GLOBAL_XLOG_REDO_STAT....5121
8.2.12.29 DBE_PERF.EMERGENCY_OPERATION_RECORDS....5122
8.2.12.30 DBE_PERF.KERNEL_INFO....5122
8.2.13 Lock....5124
8.2.13.1 DBE_PERF.LOCKS....5124
8.2.13.2 DBE_PERF.GLOBAL_LOCKS....5126
8.2.14 Wait Event....5127
8.2.14.1 DBE_PERF.WAIT_EVENTS....5127
8.2.14.2 DBE_PERF.GLOBAL_WAIT_EVENTS....5128
8.2.14.3 DBE_PERF.WAIT_EVENT_INFO....5129
8.2.14.4 DBE_PERF.GLOBAL_DATANODES_WAIT_EVENTS....5143
8.2.15 Configuration....5144
8.2.15.1 DBE_PERF.CONFIG_SETTINGS....5144
8.2.15.2 DBE_PERF.GLOBAL_CONFIG_SETTINGS....5145
8.2.16 Operator....5146
8.2.16.1 DBE_PERF.OPERATOR_HISTORY_TABLE....5146
8.2.16.2 DBE_PERF.OPERATOR_RUNTIME....5148
8.2.16.3 DBE_PERF.GLOBAL_OPERATOR_HISTORY....5149
8.2.16.4 DBE_PERF.GLOBAL_OPERATOR_HISTORY_TABLE....5151
8.2.16.5 DBE_PERF.GLOBAL_OPERATOR_RUNTIME....5151
8.2.17 Workload Manager....5153
8.2.17.1 DBE_PERF.WLM_CGROUP_CONFIG....5153
8.2.17.2 DBE_PERF.WLM_CLUSTER_RESOURCE_RUNTIME....5153
8.2.17.3 DBE_PERF.WLM_CONTROLGROUP_CONFIG....5154
8.2.17.4 DBE_PERF.WLM_CONTROLGROUP_NG_CONFIG....5154
8.2.17.5 DBE_PERF.WLM_RESOURCEPOOL_RUNTIME....5155
8.2.17.6 DBE_PERF.WLM_USER_RESOURCE_CONFIG....5156
8.2.17.7 DBE_PERF.WLM_USER_RESOURCE_RUNTIME....5156
8.2.17.8 DBE_PERF.WLM_WORKLOAD_HISTORY_INFO....5157
8.2.17.9 DBE_PERF.WLM_WORKLOAD_RUNTIME....5158
8.2.17.10 DBE_PERF.GLOBAL_WLM_WORKLOAD_RUNTIME....5159
8.2.17.11 DBE_PERF.LOCAL_IO_WAIT_INFO....5160
8.2.17.12 DBE_PERF.GLOBAL_IO_WAIT_INFO....5161
8.2.18 Global Plan Cache....5162
8.2.18.1 DBE_PERF.LOCAL_PLANCACHE_STATUS....5162
8.2.18.2 DBE_PERF.GLOBAL_PLANCACHE_STATUS....5163
8.2.18.3 DBE_PERF.LOCAL_PREPARE_STATEMENT_STATUS....5164
8.2.18.4 DBE_PERF.GLOBAL_PREPARE_STATEMENT_STATUS....5164
8.2.18.5 DBE_PERF.LOCAL_PLANCACHE_CLEAN....5164

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.18.6 DBE_PERF.GLOBAL_PLANCACHE_CLEAN....5164
8.2.19 RTO & RPO....5165
8.2.19.1 DBE_PERF.GLOBAL_RTO_STATUS....5165
8.2.19.2 DBE_PERF.GLOBAL_STREAMING_HADR_RTO_AND_RPO_STAT....5165
8.2.20 AI Watchdog....5166
8.2.20.1 DBE_PERF.AI_WATCHDOG_MONITOR_STATUS....5167
8.2.20.2 DBE_PERF.AI_WATCHDOG_DETECTION_WARNINGS....5168
8.2.20.3 DBE_PERF.AI_WATCHDOG_PARAMETERS....5169
8.2.20.4 DBE_PERF.AI_WATCHDOG_FTASK_STATUS....5169
8.2.21 Trace Function....5170
8.2.21.1 DBE_PERF.pl_trace_func_start....5170
8.2.21.2 DBE_PERF.pl_trace_func_end....5171
8.2.21.3 DBE_PERF.pl_trace_info_clean....5171
8.2.21.4 DBE_PERF.pl_get_trace_info....5172
8.2.21.5 DBE_PERF.pl_get_global_trace_info....5172
8.2.22 Deprecated Views....5173
8.2.22.1 Query....5173
8.2.22.2 Operator....5173
8.3 WDR Snapshot Schema....5173
8.3.1 Original Information of WDR Snapshots....5173
8.3.1.1 SNAPSHOT.SNAPSHOT....5173
8.3.1.2 SNAPSHOT.TABLES_SNAP_TIMESTAMP....5174
8.3.1.3 SNAPSHOT.SNAP_SEQ....5175
8.3.1.4 SNAPSHOT.SNAP_QUERY....5175
8.3.1.5 SNAPSHOT.SNAP_PLAN....5176
8.3.2 WDR Snapshot Data Table....5176
8.4 DBE_PLDEBUGGER Schema....5177
8.4.1 DBE_PLDEBUGGER.turn_on....5181
8.4.2 DBE_PLDEBUGGER.turn_off....5181
8.4.3 DBE_PLDEBUGGER.local_debug_server_info....5182
8.4.4 DBE_PLDEBUGGER.attach....5182
8.4.5 DBE_PLDEBUGGER.info_locals....5183
8.4.6 DBE_PLDEBUGGER.next....5184
8.4.7 DBE_PLDEBUGGER.continue....5184
8.4.8 DBE_PLDEBUGGER.abort....5185
8.4.9 DBE_PLDEBUGGER.print_var....5185
8.4.10 DBE_PLDEBUGGER.info_code....5186
8.4.11 DBE_PLDEBUGGER.step....5186
8.4.12 DBE_PLDEBUGGER.add_breakpoint....5186
8.4.13 DBE_PLDEBUGGER.delete_breakpoint....5187
8.4.14 DBE_PLDEBUGGER.info_breakpoints....5187
8.4.15 DBE_PLDEBUGGER.backtrace....5188

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.4.16 DBE_PLDEBUGGER.enable_breakpoint 5188
8.4.17 DBE_PLDEBUGGER.disable_breakpoint 5188
8.4.18 DBE_PLDEBUGGER.finish 5189
8.4.19 DBE_PLDEBUGGER.set_var 5189
8.4.20 DBE_PLDEBUGGER.error_backtrace 5189
8.4.21 DBE_PLDEBUGGER.error_end 5190
8.4.22 DBE_PLDEBUGGER.error_info_locals 5190
8.5 DBE_PLDEVELOPER 5191
8.5.1 DBE_PLDEVELOPER.GS_SOURCE 5191
8.6 DBE_COMPRESSION Schema 5192
8.7 DBE_HEAT_MAP Schema 5192
8.8 DBE_ILM Schema 5193
8.9 DBE_ILM_ADMIN Schema 5193
8.10 PRVT_ILM Schema 5194
9 WDR 5197
9.1 Database Stat 5199
9.2 Load Profile 5200
9.3 Instance Efficiency Percentages 5201
9.4 Top 10 Events by Total Wait Time 5201
9.5 Wait Classes by Total Wait Time 5202
9.6 Host CPU 5202
9.7 IO Profile 5203
9.8 Memory Statistics 5203
9.9 Time Model 5204
9.10 SQL Statistics 5205
9.11 Wait Events 5207
9.12 Cache IO Stats 5207
9.13 Utility status 5209
9.14 Object Stats 5210
9.15 Configuration settings 5212
9.16 SQL Detail 5213
10 ASP 5214

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1 SQL Reference

1.1 GaussDB SQL

What Is SQL?

SQL is a standard computer language used to control the access to databases and manage data in databases.

SQL provides different statements to enable you to:

- Query data.
- Insert, update, and delete rows.
- Create, replace, modify, and delete objects.
- Control the access to a database and its objects.
- Maintain the consistency and integrity of a database.

SQL consists of commands and functions that are used to manage databases and database objects. This language can also forcibly implement the rules for data types, expressions, and texts. Therefore, **SQL Reference** describes data types, expressions, functions, and operators in addition to SQL syntax.

Development of SQL Standards

The development history of SQL standards is as follows:

- 1986: ANSI X3.135-1986, ISO/IEC 9075:1986, SQL-86
- 1989: ANSI X3.135-1989, ISO/IEC 9075:1989, SQL-89
- 1992: ANSI X3.135-1992, ISO/IEC 9075:1992, SQL-92
- 1999: ISO/IEC 9075:1999, SQL:1999
- 2003: ISO/IEC 9075:2003, SQL:2003
- 2011: ISO/IEC 9075:2011, SQL:2011
- 2016: ISO/IEC 9075:2016, SQL:2016
- 2019: ISO/IEC 9075:2019, SQL:2019

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

SQL Standards Supported by GaussDB

By default, GaussDB supports most features of SQL:2016.

1.2 Keywords

SQL statements are classified into reserved keywords and non-reserved keywords. For details about common SQL keywords, see **Table 1-1**. Standards require that reserved keywords not be used as other identifiers. Non-reserved keywords have special meanings only in a specific environment and can be used as identifiers in other environments.

NOTICE

- Currently, the non-reserved keywords have the following restrictions when being used as the identifier of a database object:
  - It cannot be directly used as a column alias. That is, usage similar to SELECT 1 ABORT may cause errors.
  - Keywords ENTITYESCAPING, NOENTITYESCAPING, and WELLFORMED cannot be used as identifiers of table names, column names, table aliases, column aliases, and function names if they are not enclosed in double quotation marks.
  - The RAW keyword without double quotation marks cannot be used as the identifier of a table name or function name.
  - The SET keyword without double quotation marks cannot be used as an identifier of a table alias. That is, usage similar to SELECT * FROM T1 SET may cause errors.
  - Keywords such as BEGIN, BY, CLOSE, CURSOR, DECLARE, DELETE, EXECUTE, FUNCTION, IF, IMMEDIATE, INSERT, LOOP, MOVE, OF, REF, RELEASE, RETURN, SAVEPOINT, STRICT, TYPE, and UPDATE without double quotation marks cannot be used as variable names.
  - When the SYS_REFCURSOR keyword is used as the identifier of a database object, if double quotation marks are not attached, a database object named **REFCURSOR** is created. If double quotation marks are attached, a database object named **SYS_REFCURSOR** is created.
- Similar to the non-reserved keywords, the non-reserved (cannot be a function or type) keywords cannot be directly used as column aliases, either.
- The reserved keyword CURRENT_TIMESTAMP without double quotation marks ("") cannot be used as a function name.
- CONVERT, TIMESTAMPADD, ADDDATE, DATE_ADD, SUBDATE, DATE_SUB, and GET_FORMAT are used as keywords only in MySQL-compatible and M-compatible modes. JSON_OBJECT, LAST_DAY, and IFNULL are used as keywords only in MySQL-compatible mode.
- The keyword NOORDER is used only in ORA-compatible mode.

Identifier Naming Conventions

Identifier naming must comply with the following rules:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- An identifier name can only contain letters, digits (0–9), underscores (_), and dollar signs ($).
- An identifier name must start with a letter or an underscore (_).

  NOTE

  - The naming rules are recommended but not required.
  - In special cases, double quotation marks (") can be used to avoid special character errors.

SQL Keywords

Table 1-1 SQL keywords

Keyword	GaussDB	SQL:1999	SQL-92
ABORT	Non-reserved	-	-
ABS	-	Non-reserved	-
ABSOLUTE	Non-reserved	Reserved	Reserved
ACCESS	Non-reserved	-	-
ACCOUNT	Non-reserved	-	-
ACTION	Non-reserved	Reserved	Reserved
ADA	-	Non-reserved	Non-reserved
ADD	Non-reserved	Reserved	Reserved
ADDDATE	Non-reserved	-	-
ADMIN	Non-reserved	Reserved	-
ADVANCED	Non-reserved	-	-
AFTER	Non-reserved	Reserved	-
AGGREGATE	Non-reserved	Reserved	-
ALGORITHM	Non-reserved	-	-
ALIAS	-	Reserved	-
ALL	Reserved	Reserved	Reserved
ALLOCATE	-	Reserved	Reserved
ALSO	Non-reserved	-	-
ALTER	Non-reserved	Reserved	Reserved
ALWAYS	Non-reserved	-	-
ANALYSE	Reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
ANALYZE	Reserved	-	-
AND	Reserved	Reserved	Reserved
ANY	Reserved	Reserved	Reserved
APP	Non-reserved	-	-
APPEND	Non-reserved	-	-
ARCHIVE	Non-reserved	-	-
ARE	-	Reserved	Reserved
ARRAY	Reserved	Reserved	-
AS	Reserved	Reserved	Reserved
ASC	Reserved	Reserved	Reserved
ASENSITIVE	-	Non-reserved	-
ASSERTION	Non-reserved	Reserved	Reserved
ASSIGNMENT	Non-reserved	Non-reserved	-
ASYMMETRIC	Reserved	Non-reserved	-
AT	Non-reserved	Reserved	Reserved
ATOMIC	-	Non-reserved	-
ATTRIBUTE	Non-reserved	-	-
AUDIT	Non-reserved	-	-
AUTHID	Reserved	-	-
AUTHORIZATION	Reserved (functions and types allowed)	Reserved	Reserved
AUTO	Non-reserved	-	-
AUTO_INCREMENT	Non-reserved	-	-
AUTOEXTEND	Non-reserved	-	-
AUTOHINT	Non-reserved	-	-
AUTOMAPPED	Non-reserved	-	-
AVG	-	Non-reserved	Reserved

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
BACKWARD	Non-reserved	-	-
BAD_PATH	Non-reserved	-	-
BARRIER	Non-reserved	-	-
BEFORE	Non-reserved	Reserved	-
BEGIN	Non-reserved	Reserved	Reserved
BEGIN_NON_ANOYBLOCK	Non-reserved	-	-
BETWEEN	Non-reserved (cannot be functions or types)	Non-reserved	Reserved
BIGINT	Non-reserved (cannot be functions or types)	-	-
BINARY	Reserved (functions and types allowed)	Reserved	-
BINARY_DOUBLE	Non-reserved (cannot be functions or types)	-	-
BINARY_INTEGER	Non-reserved (cannot be functions or types)	-	-
BIT	Non-reserved (cannot be functions or types)	Reserved	Reserved
BIT_LENGTH	-	Non-reserved	Reserved
BITVAR	-	Non-reserved	-
BLANKS	Non-reserved	-	-
BLOB	Non-reserved	Reserved	-
BLOCKCHAIN	Non-reserved	-	-
BODY	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
BOOLEAN	Non-reserved (cannot be functions or types)	Reserved	-
BOTH	Reserved	Reserved	Reserved
BREADTH	-	Reserved	-
BUCKETS	Reserved	-	-
BY	Non-reserved	Reserved	Reserved
BYTE	Non-reserved	-	-
BYTEAWITHOUTORDER	Non-reserved (cannot be functions or types)	-	-
BYTEAWITHOUTORDER-WITHEQUAL	Non-reserved (cannot be functions or types)	-	-
C	-	Non-reserved	Non-reserved
CACHE	Non-reserved	-	-
CALL	Non-reserved	Reserved	-
CALLED	Non-reserved	Non-reserved	-
CANCELABLE	Non-reserved	-	-
CARDINALITY	-	Non-reserved	-
CASCADE	Non-reserved	Reserved	Reserved
CASCADED	Non-reserved	Reserved	Reserved
CASE	Reserved	Reserved	Reserved
CAST	Reserved	Reserved	Reserved
CATALOG	Non-reserved	Reserved	Reserved
CATALOG_NAME	-	Non-reserved	Non-reserved
CHAIN	Non-reserved	Non-reserved	-
CHANGE	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
CHAR	Non-reserved (cannot be functions or types)	Reserved	Reserved
CHAR_LENGTH	-	Non-reserved	Reserved
CHARACTER	Non-reserved (cannot be functions or types)	Reserved	Reserved
CHARACTER_LENGTH	-	Non-reserved	Reserved
CHARACTER_SET_CATALOG	-	Non-reserved	Non-reserved
CHARACTER_SET_NAME	-	Non-reserved	Non-reserved
CHARACTER_SET_SCHEMA	-	Non-reserved	Non-reserved
CHARACTERISTICS	Non-reserved	-	-
CHARACTERSET	Non-reserved	-	-
CHARSET	Non-reserved	-	-
CHECK	Reserved	Reserved	Reserved
CHECKED	-	Non-reserved	-
CHECKPOINT	Non-reserved	-	-
CLASS	Non-reserved	Reserved	-
CLASS_ORIGIN	-	Non-reserved	Non-reserved
CLEAN	Non-reserved	-	-
CLIENT	Non-reserved	-	-
CLIENT_MASTER_KEY	Non-reserved	-	-
CLIENT_MASTER_KEYS	Non-reserved	-	-
CLOB	Non-reserved	Reserved	-
CLOSE	Non-reserved	Reserved	Reserved
CLUSTER	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
COALESCE	Non-reserved (cannot be functions or types)	Non-reserved	Reserved
COBOL	-	Non-reserved	Non-reserved
COLLATE	Reserved	Reserved	Reserved
COLLATION	Reserved (functions and types allowed)	Reserved	Reserved
COLLATION_CATALOG	-	Non-reserved	Non-reserved
COLLATION_NAME	-	Non-reserved	Non-reserved
COLLATION_SCHEMA	-	Non-reserved	Non-reserved
COLUMN	Reserved	Reserved	Reserved
COLUMN_ENCRYPTION_KEY	Non-reserved	-	-
COLUMN_ENCRYPTION_KEYS	Non-reserved	-	-
COLUMN_NAME	-	Non-reserved	Non-reserved
COLUMNS	Non-reserved	-	-
COLVIEW	Non-reserved	-	-
COMMAND_FUNCTION	-	Non-reserved	Non-reserved
COMMAND_FUNCTION_CODE	-	Non-reserved	-
COMMENT	Non-reserved	-	-
COMMENTS	Non-reserved	-	-
COMMIT	Non-reserved	Reserved	Reserved
COMMITTED	Non-reserved	Non-reserved	Non-reserved
COMPACT	Reserved (functions and types allowed)	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
COMPATIBLE_ILLEGAL_CHARS	Non-reserved	-	-
COMPILE	Non-reserved	-	-
COMPLETE	Non-reserved	-	-
COMPLETION	Non-reserved	Reserved	-
COMPRESS	Non-reserved	-	-
CONCURRENTLY	Reserved (functions and types allowed)	-	-
CONDITION	Non-reserved	-	-
CONDITION_NUMBER	-	Non-reserved	Non-reserved
CONFIGURATION	Non-reserved	-	-
CONFLICT	Non-reserved	-	-
CONNECT	Non-reserved	Reserved	Reserved
CONNECTION	Non-reserved	Reserved	Reserved
CONNECTION_NAME	-	Non-reserved	Non-reserved
CONSTANT	Non-reserved	-	-
CONSTRAINT	Reserved	Reserved	Reserved
CONSTRAINT_CATALOG	-	Non-reserved	Non-reserved
CONSTRAINT_NAME	-	Non-reserved	Non-reserved
CONSTRAINT_SCHEMA	-	Non-reserved	Non-reserved
CONSTRAINTS	Non-reserved	Reserved	Reserved
CONSTRUCTOR	-	Reserved	-
CONTAINING	Non-reserved	-	-
CONTAINS	-	Non-reserved	-
CONTENT	Non-reserved	-	-
CONTINUE	Non-reserved	Reserved	Reserved
CONVERSION	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
CONVERT	Non-reserved	Non-reserved	Reserved
COORDINATOR	Non-reserved	-	-
COORDINATORS	Non-reserved	-	-
COPY	Non-reserved	-	-
CORRESPONDING	-	Reserved	Reserved
COST	Non-reserved	-	-
COUNT	-	Non-reserved	Reserved
CREATE	Reserved	Reserved	Reserved
CROSS	Reserved (functions and types allowed)	Reserved	Reserved
CROSSBUCKET	Reserved	-	-
CSN	Reserved (functions and types allowed)	-	-
CSV	Non-reserved	-	-
CUBE	Non-reserved	Reserved	-
CURRENT	Non-reserved	Reserved	Reserved
CURRENT_CATALOG	Reserved	-	-
CURRENT_DATE	Reserved	Reserved	Reserved
CURRENT_PATH	-	Reserved	-
CURRENT_ROLE	Reserved	Reserved	-
CURRENT_SCHEMA	Reserved (functions and types allowed)	-	-
CURRENT_TIME	Reserved	Reserved	Reserved
CURRENT_TIMESTAMP	Reserved	Reserved	Reserved
CURRENT_USER	Reserved	Reserved	Reserved
CURSOR	Non-reserved	Reserved	Reserved
CURSOR_NAME	-	Non-reserved	Non-reserved
CYCLE	Non-reserved	Reserved	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
DATA	Non-reserved	Reserved	Non-reserved
DATABASE	Non-reserved	-	-
DATAFILE	Non-reserved	-	-
DATANODE	Non-reserved	-	-
DATANODES	Non-reserved	-	-
DATATYPE_CL	Non-reserved	-	-
DATE	Non-reserved (cannot be functions or types)	Reserved	Reserved
DATE_ADD	Non-reserved	-	-
DATE_FORMAT	Non-reserved	-	-
DATE_SUB	Non-reserved	-	-
DATETIME	Non-reserved	-	-
DATETIME_INTERVAL_CODE	-	Non-reserved	Non-reserved
DATETIME_INTERVAL_PRECISION	-	Non-reserved	Non-reserved
DAY	Non-reserved	Reserved	Reserved
DAY_HOUR	Non-reserved	-	-
DAY_MICROSECOND	Non-reserved	-	-
DAY_MINUTE	Non-reserved	-	-
DAY_SECOND	Non-reserved	-	-
DAYS	Non-reserved	-	-
DB4AISHOT	Non-reserved	-	-
DBCOMPATIBILITY	Non-reserved	-	-
DBTIMEZONE	Reserved	-	-
DEALLOCATE	Non-reserved	Reserved	Reserved
DEC	Non-reserved (cannot be functions or types)	Reserved	Reserved

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
DECIMAL	Non-reserved (cannot be functions or types)	Reserved	Reserved
DECLARE	Non-reserved	Reserved	Reserved
DECODE	Non-reserved (cannot be functions or types)	-	-
DEFAULT	Reserved	Reserved	Reserved
DEFAULTS	Non-reserved	-	-
DEFERRABLE	Reserved	Reserved	Reserved
DEFERRED	Non-reserved	Reserved	Reserved
DEFINED	-	Non-reserved	-
DEFINER	Non-reserved	Non-reserved	-
DELETE	Non-reserved	Reserved	Reserved
DELETE_ALL	Non-reserved	-	-
DELIMITER	Non-reserved	-	-
DELIMITERS	Non-reserved	-	-
DELTA	Non-reserved	-	-
DELTAMERGE	Reserved (functions and types allowed)	-	-
DEPTH	-	Reserved	-
DEREF	-	Reserved	-
DESC	Reserved	Reserved	Reserved
DESCRIBE	-	Reserved	Reserved
DESCRIPTOR	-	Reserved	Reserved
DESTROY	-	Reserved	-
DESTRUCTOR	-	Reserved	-
DETERMINISTIC	Non-reserved	Reserved	-
DIAGNOSTICS	-	Reserved	Reserved

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
DICTIONARY	Non-reserved	Reserved	-
DIRECT	Non-reserved	-	-
DIRECTORY	Non-reserved	-	-
DISABLE	Non-reserved	-	-
DISABLE_ALL	Non-reserved	-	-
DISCARD	Non-reserved	-	-
DISCARD_PATH	Non-reserved	-	-
DISCONNECT	Non-reserved	Reserved	Reserved
DISPATCH	-	Non-reserved	-
DISTINCT	Reserved	Reserved	Reserved
DISTRIBUTE	Non-reserved	-	-
DISTRIBUTED	Non-reserved	-	-
DISTRIBUTION	Non-reserved	-	-
DO	Reserved	-	-
DOCUMENT	Non-reserved	-	-
DOMAIN	Non-reserved	Reserved	Reserved
DOUBLE	Non-reserved	Reserved	Reserved
DROP	Non-reserved	Reserved	Reserved
DUMPFILE	Non-reserved	-	-
DUPLICATE	Non-reserved	-	-
DYNAMIC	-	Reserved	-
DYNAMIC_FUNCTION	-	Non-reserved	Non-reserved
DYNAMIC_FUNCTION_CODE	-	Non-reserved	-
EACH	Non-reserved	Reserved	-
ELASTIC	Non-reserved	-	-
ELSE	Reserved	Reserved	Reserved
EMPTY	Non-reserved	-	-
ENABLE	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
ENABLE_ALL	Non-reserved	-	-
ENCLOSED	Non-reserved	-	-
ENCODING	Non-reserved	-	-
ENCRYPTED	Non-reserved	-	-
ENCRYPTED_VALUE	Non-reserved	-	-
ENCRYPTION	Non-reserved	-	-
ENCRYPTION_TYPE	Non-reserved	-	-
END	Reserved	Reserved	Reserved
END-EXEC	-	Reserved	Reserved
ENDS	Non-reserved	-	-
ENFORCED	Non-reserved	-	-
ENGINE	Non-reserved	-	-
ENTITYESCAPING	Non-reserved	-	-
ENUM	Non-reserved	-	-
EOL	Non-reserved	-	-
EQUALS	-	Reserved	-
ERROR	Non-reserved	-	-
ERRORS	Non-reserved	-	-
ESCAPE	Non-reserved	Reserved	Reserved
ESCAPED	Non-reserved	-	-
ESCAPING	Non-reserved	-	-
EVALNAME	Non-reserved	-	-
EVENT	Non-reserved	-	-
EVENTS	Non-reserved	-	-
EVERY	Non-reserved	Reserved	-
EXCEPT	Reserved	Reserved	Reserved
EXCEPTION	-	Reserved	Reserved
EXCHANGE	Non-reserved	-	-
EXCLUDE	Non-reserved	-	-
EXCLUDED	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
EXCLUDING	Non-reserved	-	-
EXCLUSIVE	Non-reserved	-	-
EXEC	-	Reserved	Reserved
EXECUTE	Non-reserved	Reserved	Reserved
EXISTING	-	Non-reserved	-
EXISTS	Non-reserved (cannot be functions or types)	Non-reserved	Reserved
EXPDP	Non-reserved	-	-
EXPIRED	Non-reserved	-	-
EXPLAIN	Non-reserved	-	-
EXTEND	Non-reserved	-	-
EXTENSION	Non-reserved	-	-
EXTERNAL	Non-reserved	Reserved	Reserved
EXTRACT	Non-reserved (cannot be functions or types)	Non-reserved	Reserved
FALSE	Reserved	Reserved	Reserved
FAMILY	Non-reserved	-	-
FAST	Non-reserved	-	-
FEATURES	Non-reserved	-	-
FENCED	Reserved	-	-
FETCH	Reserved	Reserved	Reserved
FIELDS	Non-reserved	-	-
FILEHEADER	Non-reserved	-	-
FILL_MISSING_FIELDS	Non-reserved	-	-
FILLER	Non-reserved	-	-
FILTER	Non-reserved	-	Reserved
FINAL	-	Non-reserved	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
FINISH	Non-reserved	-	-
FIRST	Non-reserved	Reserved	Reserved
FIXED	Non-reserved	-	Reserved
FLOAT	Non-reserved (cannot be functions or types)	Reserved	Reserved
FOLLOWING	Non-reserved	-	-
FOR	Reserved	Reserved	Reserved
FORCE	Reserved	-	-
FOREIGN	Reserved	Reserved	Reserved
FORMATTER	Non-reserved	-	-
FORTRAN	-	Non-reserved	Non-reserved
FORWARD	Non-reserved	-	-
FOUND	-	Reserved	Reserved
FREE	-	Reserved	-
FREEZE	Reserved (functions and types allowed)	-	-
FROM	Reserved	Reserved	Reserved
FULL	Reserved (functions and types allowed)	Reserved	Reserved
FUNCTION	Non-reserved	Reserved	-
FUNCTIONS	Non-reserved	-	-
G	-	Non-reserved	-
GENERAL	-	Reserved	-
GENERATED	Non-reserved	Non-reserved	-
GET	-	Reserved	Reserved
GET_FORMAT	Non-reserved	-	-
GLOBAL	Non-reserved	Reserved	Reserved

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
GO	-	Reserved	Reserved
GOTO	-	Reserved	Reserved
GRANT	Reserved	Reserved	Reserved
GRANTED	Non-reserved	Non-reserved	-
GREATEST	Non-reserved (cannot be functions or types)	-	-
GROUP	Reserved	Reserved	Reserved
GROUPING	Non-reserved (cannot be functions or types)	Reserved	-
GSIUSABLE	Non-reserved	-	-
GSIVALID	Non-reserved	-	-
GSIWAITALL	Non-reserved	-	-
HANDLER	Non-reserved	-	-
HAVING	Reserved	Reserved	Reserved
HDFSDIRECTORY	Reserved (functions and types allowed)	-	-
HEADER	Non-reserved	-	-
HIERARCHY	-	Non-reserved	-
HIGH	Non-reserved	-	-
HOLD	Non-reserved	Non-reserved	-
HOST	-	Reserved	-
HOUR	Non-reserved	Reserved	Reserved
HOUR_MICROSECOND	Non-reserved	-	-
HOUR_MINUTE	Non-reserved	-	-
HOUR_SECOND	Non-reserved	-	-
IDENTIFIED	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
IDENTITY	Non-reserved	Reserved	Reserved
IF	Non-reserved	-	-
IFNULL	Non-reserved	-	-
IGNORE	Non-reserved	Reserved	-
IGNORE_EXTRA_DATA	Non-reserved	-	-
ILIKE	Reserved (functions and types allowed)	-	-
ILM	Non-reserved	-	-
ILM_PIDX_LIST	Non-reserved	-	-
IMCVTABLE	Non-reserved	-	-
IMMEDIATE	Non-reserved	Reserved	Reserved
IMMUTABLE	Non-reserved	-	-
IMPDP	Non-reserved	-	-
IMPLEMENTATION	-	Non-reserved	-
IMPLICIT	Non-reserved	-	-
IN	Reserved	Reserved	Reserved
INCLUDE	Non-reserved	-	-
INCLUDING	Non-reserved	-	-
INCREMENT	Non-reserved	-	-
INCREMENTAL	Non-reserved	-	-
INDEX	Non-reserved	-	-
INDEXES	Non-reserved	-	-
INDICATOR	-	Reserved	Reserved
INFILE	Non-reserved	-	-
INFIX	-	Non-reserved	-
INHERIT	Non-reserved	-	-
INHERITS	Non-reserved	-	-
INITIAL	Non-reserved	-	-
INITIALIZE	-	Reserved	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
INITIALLY	Reserved	Reserved	Reserved
INITRANS	Non-reserved	-	-
INLINE	Non-reserved	-	-
INNER	Reserved (functions and types allowed)	Reserved	Reserved
INOUT	Non-reserved (cannot be functions or types)	Reserved	-
INPUT	Non-reserved	Reserved	Reserved
INSENSITIVE	Non-reserved	Non-reserved	Reserved
INSERT	Non-reserved	Reserved	Reserved
INSTANCE	-	Non-reserved	-
INSTANTIABLE	-	Non-reserved	-
INSTEAD	Non-reserved	-	-
INT	Non-reserved (cannot be functions or types)	Reserved	Reserved
INTEGER	Non-reserved (cannot be functions or types)	Reserved	Reserved
INTERNAL	Non-reserved	-	-
INTERSECT	Reserved	Reserved	Reserved
INTERVAL	Non-reserved (cannot be functions or types)	Reserved	Reserved
INTO	Reserved	Reserved	Reserved
INVISIBLE	Non-reserved	-	-
INVOKER	Non-reserved	Non-reserved	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
IP	Non-reserved	-	-
IS	Reserved	Reserved	Reserved
ISNULL	Non-reserved	-	-
ISOLATION	Non-reserved	Reserved	Reserved
ITERATE	-	Reserved	-
JOIN	Reserved (functions and types allowed)	Reserved	Reserved
JSON_OBJECT	Non-reserved	-	-
K	-	Non-reserved	-
KEY	Non-reserved	Reserved	Reserved
KEY_MEMBER	-	Non-reserved	-
KEY_PATH	Non-reserved	-	-
KEY_STORE	Non-reserved	-	-
KEY_TYPE	-	Non-reserved	-
KILL	Non-reserved	-	-
LABEL	Non-reserved	-	-
LANGUAGE	Non-reserved	Reserved	Reserved
LARGE	Non-reserved	Reserved	-
LAST	Non-reserved	Reserved	Reserved
LAST_DAY	Non-reserved	-	-
LATERAL	-	Reserved	-
LC_COLLATE	Non-reserved	-	-
LC_CTYPE	Non-reserved	-	-
LEADING	Reserved	Reserved	Reserved
LEAKPROOF	Non-reserved	-	-
LEAST	Non-reserved (cannot be functions or types)	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
LEFT	Reserved (functions and types allowed)	Reserved	Reserved
LENGTH	-	Non-reserved	Non-reserved
LESS	Reserved	Reserved	-
LEVEL	Non-reserved	Reserved	Reserved
LIKE	Reserved (functions and types allowed)	Reserved	Reserved
LIMIT	Reserved	Reserved	-
LINES	Non-reserved	-	-
LINK	Non-reserved	-	-
LIST	Non-reserved	-	-
LISTEN	Non-reserved	-	-
LNNVL	Non-reserved (cannot be functions or types)	-	-
LOAD	Non-reserved	-	-
LOAD_BAD	Non-reserved	-	-
LOAD_DISCARD	Non-reserved	-	-
LOAD_INTEGER_LEN	Non-reserved	-	-
LOAD_SESSION_ID	Non-reserved	-	-
LOAD_SMALLINT_LEN	Non-reserved	-	-
LOAD_UNFIXED_LEN	Non-reserved	-	-
LOAD_UNFIXED_START_POS	Non-reserved	-	-
LOCAL	Non-reserved	Reserved	Reserved
LOCALTIME	Reserved	Reserved	-
LOCALTIMESTAMP	Reserved	Reserved	-
LOCATION	Non-reserved	-	-
LOCATOR	-	Reserved	-
LOCK	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
LOCKED	Non-reserved	-	-
LOG	Non-reserved	-	-
LOGGING	Non-reserved	-	-
LOGIN_ANY	Non-reserved	-	-
LOGIN_FAILURE	Non-reserved	-	-
LOGIN_SUCCESS	Non-reserved	-	-
LOGOUT	Non-reserved	-	-
LOOP	Non-reserved	-	-
LOW	Non-reserved	-	-
LOWER	-	Non-reserved	Reserved
M	-	Non-reserved	-
MAP	-	Reserved	-
MAPPING	Non-reserved	-	-
MARK	Non-reserved	-	-
MASKING	Non-reserved	-	-
MASTER	Reserved	-	-
MATCH	Non-reserved	Reserved	Reserved
MATCHED	Non-reserved	-	-
MATERIALIZED	Non-reserved	-	-
MAX	-	Non-reserved	Reserved
MAXEXTENTS	Non-reserved	-	-
MAXSIZE	Non-reserved	-	-
MAXTRANS	Non-reserved	-	-
MAXVALUE	Reserved	-	-
MEDIUM	Non-reserved	-	-
MEDIUMINT	Non-reserved (cannot be functions or types)	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
MERGE	Non-reserved	-	-
MESSAGE_LENGTH	-	Non-reserved	Non-reserved
MESSAGE_OCTET_LENGTH	-	Non-reserved	Non-reserved
MESSAGE_TEXT	-	Non-reserved	Non-reserved
METHOD	-	Non-reserved	-
MICROSECOND	Non-reserved	-	-
MIN	-	Non-reserved	Reserved
MINEXTENTS	Non-reserved	-	-
MINUS	Reserved	-	-
MINUTE	Non-reserved	Reserved	Reserved
MINUTE_MICROSECOND	Non-reserved	-	-
MINUTE_SECOND	Non-reserved	-	-
MINVALUE	Non-reserved	-	-
MOD	-	Non-reserved	-
MODE	Non-reserved	-	-
MODEL	Non-reserved	-	-
MODIFICATION	Non-reserved	-	-
MODIFIES	-	Reserved	-
MODIFY	Reserved	Reserved	-
MODULE	-	Reserved	Reserved
MONTH	Non-reserved	Reserved	Reserved
MONTHS	Non-reserved	-	-
MORE	-	Non-reserved	Non-reserved
MOVE	Non-reserved	-	-
MOVEMENT	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
MUMPS	-	Non-reserved	Non-reserved
NAME	Non-reserved	Non-reserved	Non-reserved
NAMES	Non-reserved	Reserved	Reserved
NATIONAL	Non-reserved (cannot be functions or types)	Reserved	Reserved
NATURAL	Reserved (functions and types allowed)	Reserved	Reserved
NCHAR	Non-reserved (cannot be functions or types)	Reserved	Reserved
NCLOB	-	Reserved	-
NEW	-	Reserved	-
NEXT	Non-reserved	Reserved	Reserved
NO	Non-reserved	Reserved	Reserved
NOCACHE	Non-reserved	-	-
NOCOLVIEW	Non-reserved	-	-
NOCOMPRESS	Non-reserved	-	-
NOCYCLE	Reserved	-	-
NODE	Non-reserved	-	-
NOENTITYESCAPING	Non-reserved	-	-
NOEXTEND	Non-reserved	-	-
NOLOGGING	Non-reserved	-	-
NOMAXVALUE	Non-reserved	-	-
NOMINVALUE	Non-reserved	-	-
NONE	Non-reserved (cannot be functions or types)	Reserved	-
NOORDER	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
NOSCALE	Non-reserved	-	-
NOT	Reserved	Reserved	Reserved
NOTHING	Non-reserved	-	-
NOTIFY	Non-reserved	-	-
NOTNULL	Reserved (functions and types allowed)	-	-
NOW	Non-reserved	-	-
NOWAIT	Non-reserved	-	-
NULL	Reserved	Reserved	Reserved
NULLABLE	-	Non-reserved	Non-reserved
NULLCOLS	Non-reserved	-	-
NULLIF	Non-reserved (cannot be functions or types)	Non-reserved	Reserved
NULLS	Non-reserved	-	-
NUMBER	Non-reserved (cannot be functions or types)	Non-reserved	Non-reserved
NUMERIC	Non-reserved (cannot be functions or types)	Reserved	Reserved
NUMSTR	Non-reserved	-	-
NVARCHAR2	Non-reserved (cannot be functions or types)	-	-
NVL	Non-reserved (cannot be functions or types)	-	-
NVL2	Non-reserved (cannot be functions or types)	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
OBJECT	Non-reserved	Reserved	-
OCTET_LENGTH	-	Non-reserved	Reserved
OF	Non-reserved	Reserved	Reserved
OFF	Non-reserved	Reserved	-
OFFLINE	Reserved	-	-
OFFSET	Reserved	-	-
OIDS	Non-reserved	-	-
OLD	-	Reserved	-
ON	Reserved	Reserved	Reserved
ONLINE	Reserved	-	-
ONLY	Reserved	Reserved	Reserved
OPEN	-	Reserved	Reserved
OPERATION	-	Reserved	-
OPERATOR	Non-reserved	-	-
OPTIMIZATION	Non-reserved	-	-
OPTION	Non-reserved	Reserved	Reserved
OPTIONALLY	Non-reserved	-	-
OPTIONS	Non-reserved	Non-reserved	-
OR	Reserved	Reserved	Reserved
ORDER	Reserved	Reserved	Reserved
ORDINALITY	Non-reserved	Reserved	-
OUT	Non-reserved (cannot be functions or types)	Reserved	-
OUTER	Reserved (functions and types allowed)	Reserved	Reserved
OUTFILE	Non-reserved	-	-
OUTPUT	-	Reserved	Reserved
OVER	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
OVERLAPS	Reserved (functions and types allowed)	Non-reserved	Reserved
OVERLAY	Non-reserved (cannot be functions or types)	Non-reserved	-
OVERRIDING	-	Non-reserved	-
OWNED	Non-reserved	-	-
OWNER	Non-reserved	-	-
PACKAGE	Non-reserved	-	-
PACKAGES	Non-reserved	-	-
PAD	-	Reserved	Reserved
PARAMETER	-	Reserved	-
PARAMETER_MODE	-	Non-reserved	-
PARAMETER_NAME	-	Non-reserved	-
PARAMETER_ORDINAL_POSITION	-	Non-reserved	-
PARAMETER_SPECIFIC_CATALOG	-	Non-reserved	-
PARAMETER_SPECIFIC_NAME	-	Non-reserved	-
PARAMETER_SPECIFIC_SCHEMA	-	Non-reserved	-
PARAMETERS	-	Reserved	-
PARSER	Non-reserved	-	-
PARTIAL	Non-reserved	Reserved	Reserved
PARTITION	Non-reserved	-	-
PARTITIONS	Non-reserved	-	-
PASCAL	-	Non-reserved	Non-reserved
PASSING	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
PASSWORD	Non-reserved	-	-
PATH	-	Reserved	-
PCTFREE	Non-reserved	-	-
PER	Non-reserved	-	-
PERCENT	Non-reserved	-	-
PERFORMANCE	Reserved	-	-
PERM	Non-reserved	-	-
PIVOT	Non-reserved	-	-
PLACING	Reserved	-	-
PLAN	Non-reserved	-	-
PLANS	Non-reserved	-	-
PLI	-	Non-reserved	Non-reserved
POLICY	Non-reserved	-	-
POOL	Non-reserved	-	-
POPULATE	Non-reserved	-	-
POSITION	Non-reserved (cannot be functions or types)	Non-reserved	Reserved
POSTFIX	-	Reserved	-
PRECEDING	Non-reserved	-	-
PRECISION	Non-reserved (cannot be functions or types)	Reserved	Reserved
PREDICT	Non-reserved	-	-
PREFERRED	Non-reserved	-	-
PREFIX	Non-reserved	Reserved	-
PREORDER	-	Reserved	-
PREPARE	Non-reserved	Reserved	Reserved
PREPARED	Non-reserved	-	-
PRESERVE	Non-reserved	Reserved	Reserved

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
PRIMARY	Reserved	Reserved	Reserved
PRIOR	Non-reserved	Reserved	Reserved
PRIORER	Reserved	-	-
PRIORITY	Non-reserved	-	-
PRIVATE	Non-reserved	-	-
PRIVILEGE	Non-reserved	-	-
PRIVILEGES	Non-reserved	Reserved	Reserved
PROCEDURAL	Non-reserved	-	-
PROCEDURE	Reserved	Reserved	Reserved
PROFILE	Non-reserved	-	-
PUBLIC	Non-reserved	Reserved	Reserved
PUBLISH	Non-reserved	-	-
PURGE	Non-reserved	-	-
QUARTER	Non-reserved	-	-
QUERY	Non-reserved	-	-
QUOTE	Non-reserved	-	-
RANDOMIZED	Non-reserved	-	-
RANGE	Non-reserved	-	-
RATIO	Non-reserved	-	-
RAW	Non-reserved	-	-
READ	Non-reserved	Reserved	Reserved
READS	-	Reserved	-
REAL	Non-reserved (cannot be functions or types)	Reserved	Reserved
REASSIGN	Non-reserved	-	-
REBUILD	Non-reserved	-	-
RECHECK	Non-reserved	-	-
RECOVER	Non-reserved	-	-
RECURSIVE	Non-reserved	Reserved	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
RECYCLEBIN	Reserved (functions and types allowed)	-	-
REDISANYVALUE	Non-reserved	-	-
REF	Non-reserved	Reserved	-
REFERENCES	Reserved	Reserved	Reserved
REFERENCING	-	Reserved	-
REFRESH	Non-reserved	-	-
REGEXP	Reserved (functions and types allowed)	-	-
REGEXP_LIKE	Non-reserved (cannot be functions or types)	-	-
REINDEX	Non-reserved	-	-
REJECT	Reserved	-	-
RELATIVE	Non-reserved	Reserved	Reserved
RELEASE	Non-reserved	-	-
RELOPTIONS	Non-reserved	-	-
REMOTE	Non-reserved	-	-
REMOVE	Non-reserved	-	-
RENAME	Non-reserved	-	-
REPEATABLE	Non-reserved	Non-reserved	Non-reserved
REPLACE	Non-reserved	-	-
REPLICA	Non-reserved	-	-
RESET	Non-reserved	-	-
RESIZE	Non-reserved	-	-
RESOURCE	Non-reserved	-	-
RESPECT	Non-reserved	-	-
RESTART	Non-reserved	-	-
RESTRICT	Non-reserved	Reserved	Reserved

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
RESULT	-	Reserved	-
RETURN	Non-reserved	Reserved	-
RETURNED_LENGTH	-	Non-reserved	Non-reserved
RETURNED_OCTET_LENGTH	-	Non-reserved	Non-reserved
RETURNED_SQLSTATE	-	Non-reserved	Non-reserved
RETURNING	Reserved	-	-
RETURNS	Non-reserved	Reserved	-
REUSE	Non-reserved	-	-
REVOKE	Non-reserved	Reserved	Reserved
RIGHT	Reserved (functions and types allowed)	Reserved	Reserved
RLIKE	Reserved (functions and types allowed)	-	-
ROLE	Non-reserved	Reserved	-
ROLES	Non-reserved	-	-
ROLLBACK	Non-reserved	Reserved	Reserved
ROLLUP	Non-reserved	Reserved	-
ROTATION	Non-reserved	-	-
ROUTINE	-	Reserved	-
ROUTINE_CATALOG	-	Non-reserved	-
ROUTINE_NAME	-	Non-reserved	-
ROUTINE_SCHEMA	-	Non-reserved	-
ROW	Non-reserved (cannot be functions or types)	Reserved	-
ROW_COUNT	-	Non-reserved	Non-reserved

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
ROWNUM	Reserved	-	-
ROWS	Non-reserved	Reserved	Reserved
ROWTYPE	Non-reserved	-	-
RULE	Non-reserved	-	-
SAMPLE	Non-reserved	-	-
SAVEPOINT	Non-reserved	Reserved	-
SCALE	Non-reserved	Non-reserved	Non-reserved
SCHEDULE	Non-reserved	-	-
SCHEMA	Non-reserved	Reserved	Reserved
SCHEMA_NAME	-	Non-reserved	Non-reserved
SCOPE	-	Reserved	-
SCROLL	Non-reserved	Reserved	Reserved
SEARCH	Non-reserved	Reserved	-
SECOND	Non-reserved	Reserved	Reserved
SECOND_MICROSECOND	Non-reserved	-	-
SECTION	-	Reserved	Reserved
SECURITY	Non-reserved	Non-reserved	-
SELECT	Reserved	Reserved	Reserved
SELF	-	Non-reserved	-
SENSITIVE	-	Non-reserved	-
SEPARATOR	Non-reserved	-	-
SEQUENCE	Non-reserved	Reserved	-
SEQUENCES	Non-reserved	-	-
SERIALIZABLE	Non-reserved	Non-reserved	Non-reserved
SERVER	Non-reserved	-	-
SERVER_NAME	-	Non-reserved	Non-reserved

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
SESSION	Non-reserved	Reserved	Reserved
SESSION_USER	Reserved	Reserved	Reserved
SESSIONTIMEZONE	Reserved	-	-
SET	Non-reserved	Reserved	Reserved
SETOF	Non-reserved (cannot be functions or types)	-	-
SETS	Non-reserved	Reserved	-
SHARE	Non-reserved	-	-
SHIPPABLE	Non-reserved	-	-
SHOW	Non-reserved	-	-
SHRINK	Reserved	-	-
SHUTDOWN	Non-reserved	-	-
SIBLINGS	Non-reserved	-	-
SIGNED	Non-reserved (cannot be functions or types)	-	-
SIMILAR	Reserved (functions and types allowed)	Non-reserved	-
SIMPLE	Non-reserved	Non-reserved	-
SIZE	Non-reserved	Reserved	Reserved
SKIP	Non-reserved	-	-
SLAVE	Non-reserved	-	-
SLICE	Non-reserved	-	-
SMALLDATETIME	Non-reserved (cannot be functions or types)	-	-
SMALLDATETIME_FORMAT	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
SMALLINT	Non-reserved (cannot be functions or types)	Reserved	Reserved
SNAPSHOT	Non-reserved	-	-
SOME	Reserved	Reserved	Reserved
SOURCE	Non-reserved	Non-reserved	-
SPACE	Non-reserved	Reserved	Reserved
SPECIFIC	-	Reserved	-
SPECIFIC_NAME	-	Non-reserved	-
SPECIFICATION	Non-reserved	-	-
SPECIFICTYPE	-	Reserved	-
SPILL	Non-reserved	-	-
SPLIT	Non-reserved	-	-
SQL	-	Reserved	Reserved
SQLCODE	-	-	Reserved
SQLERROR	-	-	Reserved
SQLEXCEPTION	-	Reserved	-
SQLSTATE	-	Reserved	Reserved
SQLWARNING	-	Reserved	-
STABLE	Non-reserved	-	-
STANDALONE	Non-reserved	-	-
START	Non-reserved	Reserved	-
STARTING	Non-reserved	-	-
STARTS	Non-reserved	-	-
STATE	-	Reserved	-
STATEMENT	Non-reserved	Reserved	-
STATEMENT_ID	Non-reserved	-	-
STATIC	-	Reserved	-
STATISTICS	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
STDIN	Non-reserved	-	-
STDOUT	Non-reserved	-	-
STORAGE	Non-reserved	-	-
STORE	Non-reserved	-	-
STORED	Non-reserved	-	-
STR	Non-reserved	-	-
STRATIFY	Non-reserved	-	-
STRICT	Non-reserved	-	-
STRIP	Non-reserved	-	-
STRUCTURE	-	Reserved	-
STYLE	-	Non-reserved	-
SUBCLASS_ORIGIN	-	Non-reserved	Non-reserved
SUBDATE	Non-reserved	-	-
SUBLIST	-	Non-reserved	-
SUBPARTITION	Non-reserved	-	-
SUBPARTITIONS	Non-reserved	-	-
SUBSTR	Non-reserved	-	-
SUBSTRING	Non-reserved (cannot be functions or types)	Non-reserved	Reserved
SUM	-	Non-reserved	Reserved
SYMMETRIC	Reserved	Non-reserved	-
SYNONYM	Non-reserved	-	-
SYS_REFCURSOR	Non-reserved	-	-
SYSDATE	Reserved	-	-
SYSID	Non-reserved	-	-
SYSTEM	Non-reserved	Non-reserved	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
SYSTEM_USER	-	Reserved	Reserved
TABLE	Reserved	Reserved	Reserved
TABLE_COMPRESS_CLAUSE	Non-reserved	-	-
TABLE_NAME	-	Non-reserved	Non-reserved
TABLES	Non-reserved	-	-
TABLESAMPLE	Reserved (functions and types allowed)	-	-
TABLESPACE	Non-reserved	-	-
TARGET	Non-reserved	-	-
TEMP	Non-reserved	-	-
TEMPLATE	Non-reserved	-	-
TEMPORARY	Non-reserved	Reserved	Reserved
TERMINATE	-	Reserved	-
TERMINATED	Non-reserved	-	-
TEXT	Non-reserved	-	-
THAN	Non-reserved	Reserved	-
THEN	Reserved	Reserved	Reserved
TIME	Non-reserved (cannot be functions or types)	Reserved	Reserved
TIME_FORMAT	Non-reserved	-	-
TIMECAPSULE	Reserved (functions and types allowed)	-	-
TIMESTAMP	Non-reserved (cannot be functions or types)	Reserved	Reserved
TIMESTAMP_FORMAT	Non-reserved	-	-
TIMESTAMPADD	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
TIMESTAMPDIFF	Non-reserved (cannot be functions or types)	-	-
TIMEZONE_HOUR	-	Reserved	Reserved
TIMEZONE_MINUTE	-	Reserved	Reserved
TINYINT	Non-reserved (cannot be functions or types)	-	-
TO	Reserved	Reserved	Reserved
TRAILING	Reserved	Reserved	Reserved
TRANSACTION	Non-reserved	Reserved	Reserved
TRANSACTION_ACTIVE	-	Non-reserved	-
TRANSACTIONS_COMMITTED	-	Non-reserved	-
TRANSACTIONS_ROLLED_BACK	-	Non-reserved	-
TRANSFORM	Non-reserved	Non-reserved	-
TRANSFORMS	-	Non-reserved	-
TRANSLATE	-	Non-reserved	Reserved
TRANSLATION	-	Reserved	Reserved
TREAT	Non-reserved (cannot be functions or types)	Reserved	-
TRIGGER	Non-reserved	Reserved	-
TRIGGER_CATALOG	-	Non-reserved	-
TRIGGER_NAME	-	Non-reserved	-
TRIGGER_SCHEMA	-	Non-reserved	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
TRIM	Non-reserved (cannot be functions or types)	Non-reserved	Reserved
TRUE	Reserved	Reserved	Reserved
TRUNCATE	Non-reserved	-	-
TRUSTED	Non-reserved	-	-
TSFIELD	Non-reserved	-	-
TSTAG	Non-reserved	-	-
TSTIME	Non-reserved	-	-
TYPE	Non-reserved	Non-reserved	Non-reserved
TYPES	Non-reserved	-	-
UNBOUNDED	Non-reserved	-	-
UNCOMMITTED	Non-reserved	Non-reserved	Non-reserved
UNDER	-	Reserved	-
UNENCRYPTED	Non-reserved	-	-
UNION	Reserved	Reserved	Reserved
UNIQUE	Reserved	Reserved	Reserved
UNKNOWN	Non-reserved	Reserved	Reserved
UNLIMITED	Non-reserved	-	-
UNLISTEN	Non-reserved	-	-
UNLOCK	Non-reserved	-	-
UNLOGGED	Non-reserved	-	-
UNNAMED	-	Non-reserved	Non-reserved
UNNEST	-	Reserved	-
UNPIVOT	Non-reserved	-	-
UNSIGNED	Non-reserved (cannot be functions or types)	-	-
UNTIL	Non-reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
UNUSABLE	Non-reserved	-	-
UPDATE	Non-reserved	Reserved	Reserved
UPPER	-	Non-reserved	Reserved
USAGE	-	Reserved	Reserved
USEEOF	Non-reserved	-	-
USER	Reserved	Reserved	Reserved
USER_DEFINED_TYPE_CATALOG	-	Non-reserved	-
USER_DEFINED_TYPE_NAME	-	Non-reserved	-
USER_DEFINED_TYPE_SCHEMA	-	Non-reserved	-
USING	Reserved	Reserved	Reserved
VACUUM	Non-reserved	-	-
VALID	Non-reserved	-	-
VALIDATE	Non-reserved	-	-
VALIDATION	Non-reserved	-	-
VALIDATOR	Non-reserved	-	-
VALUE	Non-reserved	Reserved	Reserved
VALUES	Non-reserved (cannot be functions or types)	Reserved	Reserved
VARCHAR	Non-reserved (cannot be functions or types)	Reserved	Reserved
VARCHAR2	Non-reserved (cannot be functions or types)	-	-
VARIABLE	-	Reserved	-
VARIABLES	Non-reserved	-	-
VARIADIC	Reserved	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
VARYING	Non-reserved	Reserved	Reserved
VCGROUP	Non-reserved	-	-
VERBOSE	Reserved (functions and types allowed)	-	-
VERIFY	Reserved	-	-
VERSION	Non-reserved	-	-
VIEW	Non-reserved	Reserved	Reserved
VISIBLE	Non-reserved	-	-
VOLATILE	Non-reserved	-	-
WAIT	Non-reserved	-	-
WEAK	Non-reserved	-	-
WEEK	Non-reserved	-	-
WELLFORMED	Non-reserved	-	-
WHEN	Reserved	Reserved	Reserved
WHENEVER	-	Reserved	Reserved
WHERE	Reserved	Reserved	Reserved
WHITESPACE	Non-reserved	-	-
WINDOW	Reserved	-	-
WITH	Reserved	Reserved	Reserved
WITHIN	Non-reserved	-	-
WITHOUT	Non-reserved	Reserved	-
WORK	Non-reserved	Reserved	Reserved
WORKLOAD	Non-reserved	-	-
WRAPPER	Non-reserved	-	-
WRITE	Non-reserved	Reserved	Reserved
XML	Non-reserved	-	-
XMLATTRIBUTES	Non-reserved (cannot be functions or types)	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
XMLCONCAT	Non-reserved (cannot be functions or types)	-	-
XMLELEMENT	Non-reserved (cannot be functions or types)	-	-
XMLEXISTS	Non-reserved (cannot be functions or types)	-	-
XMLFOREST	Non-reserved (cannot be functions or types)	-	-
XMLNAMESPACES	Non-reserved	-	-
XMLPARSE	Non-reserved (cannot be functions or types)	-	-
XMLPI	Non-reserved (cannot be functions or types)	-	-
XMLROOT	Non-reserved (cannot be functions or types)	-	-
XMLSERIALIZE	Non-reserved (cannot be functions or types)	-	-
XMLTABLE	Non-reserved	-	-
XMLQUERY	Non-reserved	-	-
XMLCAST	Non-reserved	-	-
XMLTYPE	Non-reserved (cannot be functions or types)	-	-
YEAR	Non-reserved	Reserved	Reserved

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	GaussDB	SQL:1999	SQL-92
YEAR_MONTH	Non-reserved	-	-
YEARS	Non-reserved	-	-
YES	Non-reserved	-	-
ZEROFILL	Non-reserved (cannot be functions or types)	-	-
ZONE	Non-reserved	Reserved	Reserved

Columns listed in the following table cannot be used as column names during table creation.

CTID	XMIN	CMIN	XMAX	CMAX
TABLEOID	XC_NODE_ID	XC_NODE_HASH	TABLEBUCKETID	GS_TUPLE_UID

1.3 Data Types

A data type is a basic data attribute. Occupied storage space and allowed operations vary according to data types. Data is stored in tables in a database. Each column of a data table specifies a data type and data must be stored according to data types.

GaussDB supports implicit conversions between certain data types. For details, see **PG_CAST**.

1.3.1 Numeric Types

Table 1-2 lists all available types. For arithmetic operators and related built-in functions, see **Arithmetic Functions and Operators**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-2 Integer types

Name	Description	Storage Space	Range
TINYINT [UNSIGNED]	Tiny integer. The signed alias is INT1, and the unsigned alias is UINT1. The UNSIGNED type can be used only in M- or MySQL-compatible mode.	1 byte	• The signed value ranges from 0 to +255. • The unsigned value ranges from 0 to +255. • When **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, the signed value ranges from –128 to +127.
SMALLINT [UNSIGNED]	Small integer. The signed alias is INT2, and the unsigned alias is UINT2. The UNSIGNED type can be used only in M- or MySQL-compatible mode.	2 bytes	• The signed value ranges from –32,768 to +32,767. • The unsigned value ranges from 0 to +65,535.
MEDIUMINT [UNSIGNED]	Medium-sized integer. The signed alias is INT4, and the unsigned alias is UINT4. The UNSIGNED type can be used only in M- or MySQL-compatible mode.	4 bytes	• The signed value ranges from –2147483648 to +2147483647. • The unsigned value ranges from 0 to +4294967295.
INTEGER [UNSIGNED]	Common integer. The signed alias is INT4, and the unsigned alias is UINT4. The UNSIGNED type can be used only in M- or MySQL-compatible mode.	4 bytes	• The signed value ranges from –2147483648 to +2147483647. • The unsigned value ranges from 0 to +4294967295.
BINARY_INTEGER	Common integer. The signed alias is INTEGER.	4 bytes	–2,147,483,648 to +2,147,483,647.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space	Range
BIGINT [UNSIGNED]	Large integer. The signed alias is INT8, and the unsigned alias is UINT8. The UNSIGNED type can be used only in M- or MySQL-compatible mode.	8 bytes	• The signed value ranges from –9,223,372,036,854,775,808 to +9,223,372,036,854,775,807. • The unsigned value ranges from 0 to +18,446,744,073,709,551,615.
INT16 [SIGNED]	A 16-byte integer cannot be used to create tables.	16 bytes	–170,141,183,460,469,231,731,687,303,715,884,105,728 to +170,141,183,460,469,231,731,687,303,715,884,105,727

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, integers support the display width and **ZEROFILL** attribute.
- The display width does not limit the range of values that can be stored in a column, nor does it affect the correct display of values that exceed the display width. For example, a column defined as SMALLINT(3) can store the full range of values for the SMALLINT type (–32768 to 32767). Even if some values have more than three digits, they will be displayed in full without being truncated or altered.
- When the display width is used together with the **ZEROFILL** attribute, zeros are padded on the left of the value until the total length reaches the specified display width. For example, if you insert the value **5** into a column defined as INT(4) ZEROFILL, the query result will be displayed as **0005**.
- If you specify **ZEROFILL** for a numeric column, the **UNSIGNED** attribute is automatically added.
- If **ZEROFILL** is not specified for a numeric column and only the width information is specified, the width information is not displayed in the table definition description.
- If **UNSIGNED** is specified, negative values are not allowed.
- Numbers of the TINYINT, SMALLINT, INTEGER, BIGINT, or INT16 type, that is, integers can be stored. Saving a number with a decimal in any of the data types will result in errors.
- The INTEGER type is the most common choice, as it offers the best balance between range, storage size, and performance. Generally, use the SMALLINT type only if you are sure that the value range is within the SMALLINT value range. The storage speed of INTEGER is much faster. BIGINT is used only when the range of INTEGER is not large enough.
- When a minus sign, plus sign, or multiplication sign is used between integer values (one of which is of the UNSIGNED type), the result is unsigned.
- The return value of the addition (+), subtraction (–), and multiplication (x) operations of the INT1, UINT1, UINT2, UINT4, or UINT8 type can exceed the range of the corresponding type. The return value of the addition (+), subtraction (–), and multiplication (x) operations of the INT2, INT4, or INT8 type cannot exceed the range of the corresponding type.
- The UNSIGNED type cannot be converted to the SET type. Do not calculate or compare the UNSIGNED type with the SET type.
- When **sql_compatibility** is set to **'MYSQL'**, non-numeric characters are automatically truncated or the value **0** is returned.

Example:

```
-- Create a database.
gaussdb=# CREATE DATABASE b_database dbcompatibility = 'MYSQL';
gaussdb=# \c b_database

-- Create a table containing TINYINT data.
b_database=# CREATE TABLE int_type_t1
        (
         IT_COL1 TINYINT,
         IT_COL2 TINYINT UNSIGNED
        );

-- Insert data.
b_database=# INSERT INTO int_type_t1 VALUES(10,20);

-- View data.
b_database=# SELECT * FROM int_type_t1;
it_col1 | it_col2
--------+---------
     10 |      20
```

```
(1 row)

-- Drop the table.
b_database=# DROP TABLE int_type_t1;

-- Create a table containing TINYINT, INTEGER, and BIGINT data.
b_database=# CREATE TABLE int_type_t2
(
    a TINYINT,
    b TINYINT,
    c INTEGER,
    d INTEGER UNSIGNED,
    e BIGINT,
    f BIGINT UNSIGNED
);

-- Insert data.
b_database=# INSERT INTO int_type_t2 VALUES(100, 10, 1000, 10000, 200, 2000);

-- View data.
b_database=# SELECT * FROM int_type_t2;
  a  | b  |  c   |   d    |  e  |  f
-----+----+------+--------+-----+------
 100 | 10 | 1000 |  10000 | 200 | 2000
(1 row)

-- Drop the table.
b_database=# DROP TABLE int_type_t2;

-- Drop the database.
b_database=# \c postgres
gaussdb=# DROP DATABASE b_database;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-3 Arbitrary precision types

Name	Description	Storage Space	Range
NUMERIC[(p[,s])], DECIMAL[(p[,s])]	• The value range of **p** is [1,1000], and the value range of **s** is [0,*p*]. • When **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, if the precision and scale are not specified, the default precision **p** is **10** and the scale **s** is **0**. If a numeric type is not attached with parentheses, it is set to numeric (10,0) only when being applied to table columns. In other scenarios, the numeric type is used based on the original range by default.	The precision is specified by users. Every four decimal digits occupy two bytes, and an extra eight-byte overhead is added to the entire data.	Up to 131,072 digits before the decimal point, and up to 16,383 digits after the decimal point when no precision is specified.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space	Range
	NOTE • **p** indicates the total digits, and **s** indicates the decimal digit. • If the number of decimal places in the inserted data exceeds the specified scale **s**, the number will be rounded off to the specified scale.		
NUMBER[(p[,s])]	Alias of the NUMERIC type.	The precision is specified by users. Every four decimal digits occupy two bytes, and an extra eight-byte overhead is added to the entire data.	Up to 131,072 digits before the decimal point, and up to 16,383 digits after the decimal point when no precision is specified.

Example:

```
-- Create a table.
gaussdb=# CREATE TABLE decimal_type_t1
(
    DT_COL1 DECIMAL(10,4)
);

-- Insert data.
gaussdb=# INSERT INTO decimal_type_t1 VALUES(123456.122331);

-- Query data in the table.
gaussdb=# SELECT * FROM decimal_type_t1;
   dt_col1
-------------
 123456.1223
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE decimal_type_t1;

-- Create a table.
gaussdb=# CREATE TABLE numeric_type_t1
(
    NT_COL1 NUMERIC(10,4)
);

-- Insert data.
gaussdb=# INSERT INTO numeric_type_t1 VALUES(123456.12354);

-- Query data in the table.
gaussdb=# SELECT * FROM numeric_type_t1;
   nt_col1
-------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 123456.1235
(1 row)

-- Insert data.
gaussdb=# INSERT INTO numeric_type_t1 VALUES(10000.12355);

-- Query data in the table.
gaussdb=# SELECT * FROM numeric_type_t1;
   nt_col1
-------------
 123456.1235
  10000.1236
(2 rows)

-- Drop the table.
gaussdb=# DROP TABLE numeric_type_t1;
```

NOTE

- Compared to the integer types, the arbitrary precision numbers require larger storage space and have lower storage efficiency, operation efficiency, and poorer compression ratio results. The integer type is the common choice when number types are defined. Arbitrary precision numbers are used when numbers exceed the maximum range indicated by the integers.
- When NUMERIC/DECIMAL is used for defining a column, you are advised to specify the precision (**p**) and scale (**s**) for the column.
- When **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, the input and output range specifications of the NUMERIC[(p[,s])] and DECIMAL[(p[,s])] types are different.
  - Input format: The value can be a number or a numeric string.
    - If the value of **sql_mode** contains **strict_trans_tables**, an error is reported if the input value is invalid or exceeds the range.
    - If the value of **sql_mode** does not contain **strict_trans_tables** and the input value is invalid or exceeds the range, a warning message is reported and **0** or the truncated value is returned.
  - Output format: The output format of the DECIMAL[(p[,s])] type is different from that of the NUMERIC[(p[,s])] type when the precision and scale are not specified. If NUMERIC has no precision or scale, the decimal part is retained during type conversion. For details, see the output format in the scenario where no parameter is set. If DECIMAL has no precision or scale, the precision is **10** and the scale is **0** during type conversion.

Table 1-4 Sequence integer

Name	Description	Storage Space	Range
SMALLSERIAL	2-byte serial integer.	2 bytes.	–32,768 to +32,767.
SERIAL	4-byte serial integer.	4 bytes.	–2,147,483,648 to +2,147,483,647.
BIGSERIAL	8-byte serial integer.	8 bytes.	–9,223,372,036,854,775,808 to +9,223,372,036,854,775,807.

Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a table.
gaussdb=# CREATE TABLE smallserial_type_tab(a SMALLSERIAL);

-- Insert data.
gaussdb=# INSERT INTO smallserial_type_tab VALUES(default);

-- Insert data again.
gaussdb=# INSERT INTO smallserial_type_tab VALUES(default);

-- View data.
gaussdb=# SELECT * FROM smallserial_type_tab;
 a
---
 1
 2
(2 rows)

-- Create a table.
gaussdb=# CREATE TABLE serial_type_tab(b SERIAL);

-- Insert data.
gaussdb=# INSERT INTO serial_type_tab VALUES(default);

-- Insert data again.
gaussdb=# INSERT INTO serial_type_tab VALUES(default);

-- View data.
gaussdb=# SELECT * FROM serial_type_tab;
 b
---
 1
 2
(2 rows)

-- Create a table.
gaussdb=# CREATE TABLE bigserial_type_tab(c BIGSERIAL);

-- Insert data.
gaussdb=# INSERT INTO bigserial_type_tab VALUES(default);

-- Insert data again.
gaussdb=# INSERT INTO bigserial_type_tab VALUES(default);

-- View data.
gaussdb=# SELECT * FROM bigserial_type_tab;
 c
---
 1
 2
(2 rows)

-- Drop the table.
gaussdb=# DROP TABLE smallserial_type_tab;

gaussdb=# DROP TABLE serial_type_tab;

gaussdb=# DROP TABLE bigserial_type_tab;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

SMALLSERIAL, SERIAL, and BIGSERIAL are not real types. They are concepts used for setting a unique identifier for a table. Therefore, an integer column is created and its default value is set to be read from a sequencer. A NOT NULL constraint is used to ensure NULL is not inserted. In most cases, you may also want to attach a UNIQUE or PRIMARY KEY constraint to prevent duplicate values from being inserted unexpectedly. Finally, the sequencer belongs to the column. When the column or table is deleted, the sequencer is deleted at the same time. Currently, the **SERIAL** column can be specified only when you create a table. You cannot add the **SERIAL** column in an existing table. In addition, **SERIAL** columns cannot be created in temporary tables because **SERIAL** is not a real type. Therefore, column types in temporary tables cannot be converted to **SERIAL**.

Table 1-5 lists the floating-point types.

NOTE

- For the behavior of the REAL, FLOAT, and DEC[(p[,s])] data types when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, see **Table 1-5**.
- In **Table 1-5**, **p** is the precision, indicating the minimum acceptable total number of integral places, and **s** indicates the decimal digit.

Table 1-5 Floating-point types

Name	Description	Storage Space	Range
REAL, FLOAT4	Single-precision floating-point number, which has precision errors. The REAL data type is mapped to the double-precision floating-point number FLOAT8 when the scenario described in **NOTE** is met. For details about the application scenario, see FLOAT8.	4 bytes.	–3.402E+38 to +3.402E +38, 6-bit decimal digits.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space	Range
REAL(p,s)	The value range of **p** is [1,1000], and the value range of **s** is [0,*p*]. If the precision and scale are not specified, the precision **p** is **10** and the scale **s** is **0** by default. This type maps to NUMERIC. For details about the application scenario, see NUMERIC. This type can be used only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.	The precision is specified by users. Every four decimal digits occupy two bytes, and an extra eight-byte overhead is added to the entire data.	Maximum 131,072 digits before the decimal point and 16,383 digits after the decimal point when the precision and scale are specified to the maximum.
DOUBLE PRECISION, FLOAT8	Double-precision floating-point number, which has precision errors.	8 bytes.	–1.79E+308 to +1.79E+308, 15-bit decimal digits.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space	Range
DOUBLE	Double-precision floating-point number, which has precision errors. This type maps to double-precision floating-point number FLOAT8. For details about the application scenario, see FLOAT8. This type can be used only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.	8 bytes.	–1.79E+308 to +1.79E+308, 15-bit decimal digits.
DOUBLE(p, s)	The value range of **p** is [1,1000], and the value range of **s** is [0,*p*]. If the precision and scale are not specified, the precision **p** is **10** and the scale **s** is **0** by default. This type maps to NUMERIC. For details about the application scenario, see NUMERIC. This type can be used only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.	The precision is specified by users. Every four decimal digits occupy two bytes, and an extra eight-byte overhead is added to the entire data.	Maximum 131,072 digits before the decimal point and 16,383 digits after the decimal point when the precision and scale are specified to the maximum.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space	Range
FLOAT[(p)]	Floating-point number, which has precision errors. The value range of precision (p) is [1,53]. The type is selected based on the precision **p**. When **p** is less than or equal to 24, the mapping type is REAL. When **p** is greater than 24 or not specified, the mapping type is DOUBLE PRECISION. In the scenario described in **NOTE**, if the precision **p** is less than or equal to 24, or it is not specified, the mapping type is FLOAT4. If **p** is greater than 24, the mapping type is DOUBLE PRECISION.	4 bytes or 8 bytes.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space	Range
FLOAT(p,s)	The value range of **p** is [1,1000], and the value range of **s** is [0,*p*]. If the precision and scale are not specified, the precision **p** is **10** and the scale **s** is **0** by default. This type maps to NUMERIC. For details about the application scenario, see NUMERIC. This type can be used only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.	The precision is specified by users. Every four decimal digits occupy two bytes, and an extra eight-byte overhead is added to the entire data.	Maximum 131,072 digits before the decimal point and 16,383 digits after the decimal point when the precision and scale are specified to the maximum.
BINARY_DOUBLE	DOUBLE PRECISION alias, compatible with Oracle Database.	8 bytes.	–1.79E+308 to +1.79E+308, 15-bit decimal digits.
DEC[(p[,s])]	The value range of **p** is [1,1000], and the value range of **s** is [0,*p*]. If the scenario in **Note** is met and the precision and scale are not specified, the precision **p** is **10** and the scale **s** is **0** by default. This type maps to NUMERIC. For details about the application scenario, see NUMERIC.	The precision is specified by users. Every four decimal digits occupy two bytes, and an extra eight-byte overhead is added to the entire data.	Maximum 131,072 digits before the decimal point and 16,383 digits after the decimal point when the precision and scale are specified to the maximum.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space	Range
INTEGER[(p[,s])]	The value range of **p** is [1,1000], and the value range of **s** is [0,*p*]. If the precision and scale are not specified, the precision **p** is **10** and the scale **s** is **0** by default. If the precision and scale are not specified, this type is mapped to INTEGER. If the precision and scale are specified, this type is mapped to NUMERIC.	The precision is specified by users. Every four decimal digits occupy two bytes, and an extra eight-byte overhead is added to the entire data.	Maximum 131,072 digits before the decimal point and 16,383 digits after the decimal point when the precision and scale are specified to the maximum. If the precision and scale are not specified, the value ranges from –2,147,483,648 to +2,147,483,647.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The binary floating-point data types REAL, FLOAT4, DOUBLE, DOUBLE PRECISION, FLOAT8, FLOAT[(p)], and BINARY_DOUBLE are imprecise numeric types. Their internal storage is approximate, which may result in slight differences during storage and retrieval. When using binary floating-point data types, pay attention to the following points:
  - Precise storage and calculation: If precise storage and calculation are required (such as for monetary values), use exact data types (for example, numeric) instead.
  - Complex calculation: When you perform complex calculation with imprecise data types to obtain critical data, it is important to carefully evaluate the results.
  - Floating-point number comparison: The result of comparing two floating-point numbers for equality may differ from expectations.
  - Underflow error: If a floating-point number is too close to zero, it cannot be accurately represented. As a result, an underflow error occurs.
- For the precision of the floating-point type, only the number of precision bits can be ensured when the data is directly read. When distributed computing is involved, the computation is executed on each DN and is finally aggregated to a CN. Therefore, the error may be amplified as the number of compute nodes increases.
- When **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, the input and output ranges of the FLOAT[(p[,s])], DOUBLE[(p[,s])], DEC[(p[,s])], and REAL[(p[,s])] types are different.
  - Input format: The value can be a number or a numeric string.
    - If the value of **sql_mode** contains **strict_trans_tables**, an error is reported if the input value is invalid or exceeds the range.
    - If the value of **sql_mode** does not contain **strict_trans_tables** and the input value is invalid or exceeds the range, a warning message is reported and **0** or the truncated value is returned.
  - Output format: The output format of the DEC[(p[,s])] type is different from that of the mapping NUMERIC[(p[,s])] type when the precision and scale are not specified. If NUMERIC has no precision or scale, the decimal part is retained during type conversion. For details, see the output format in the scenario where no parameter is set. If DEC has no precision or scale, the precision is **10** and the scale is **0** during type conversion.

Example:

```
-- Create a table.
gaussdb=# CREATE TABLE float_type_t2
(
    FT_COL1 INTEGER,
    FT_COL2 FLOAT4,
    FT_COL3 FLOAT8,
    FT_COL4 FLOAT(3),
    FT_COL5 BINARY_DOUBLE,
    FT_COL6 DECIMAL(10,4),
    FT_COL7 INTEGER(6,3)
)DISTRIBUTE BY HASH ( ft_col1);

-- Insert data.
gaussdb=# INSERT INTO float_type_t2 VALUES(10,10.365456,123456.1234,10.3214, 321.321, 123.123654,
123.123654);

-- View data.
gaussdb=# SELECT * FROM float_type_t2 ;
 ft_col1 | ft_col2 |   ft_col3   | ft_col4 | ft_col5 | ft_col6  | ft_col7
---------+---------+-------------+---------+---------+----------+---------
      10 | 10.3655 | 123456.1234 | 10.3214 | 321.321 | 123.1237 | 123.124
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop the table.
gaussdb=# DROP TABLE float_type_t2;

-- Example: REAL(p,s), FLOAT(p,s), DOUBLE, and DOUBLE(p,s).
gaussdb=# CREATE DATABASE gaussdb_m  WITH dbcompatibility  'MYSQL';
gaussdb=# \c gaussdb_m
-- Set compatible version control parameters.
gaussdb_m=# SET b_format_version='5.7';
gaussdb_m=# SET b_format_dev_version='s1';
-- Create a table.
gaussdb_m=# CREATE TABLE t1(a real(10,2), b float(10,2), c double, d double(10,2));
-- Insert data.
gaussdb_m=# INSERT INTO t1 VALUES (1000.12, 2000.23, 3000.34, 4000.45);
-- Query data in the table.
gaussdb_m=# SELECT * FROM t1;
   a    |    b    |    c    |    d
--------+---------+---------+---------
 1000.12 | 2000.23 | 3000.34 | 4000.45
(1 row)

-- Drop the table and database.
gaussdb_m=# DROP TABLE t1;
gaussdb_m=# \c postgres;
gaussdb=# DROP DATABASE gaussdb_m;

-- Reset parameters.
gaussdb=# RESET ALL;
```

1.3.2 Monetary Types

The monetary type stores a currency amount with fixed fractional precision.

The range displayed in **Table 1-6** is based on two decimal places. This type supports multiple input formats, including integers, floating-point numbers, and typical currency formats (such as $1,000.00). Output is generally in the last format but depends on the locale.

Table 1-6 Monetary type

Name	Description	Storage Space	Range
money	Currency amount.	8 bytes	-92233720368547758.08 ~ +92233720368547758.07

Values of the numeric, int, and bigint data types can be directly cast to money. For the real and double precision data types, you are advised to convert the data type to numeric first, and then to money. The following is an example:

```
gaussdb=# SELECT '12.34'::float8::numeric::money;
 money
--------
 $12.34
(1 row)
```

This usage is not recommended. Floating-point numbers have a risk of precision loss. Therefore, they should not be used to process currency types.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

A money value can be cast to numeric without loss of precision. Conversion to other types could potentially lose precision, and must also be done in two stages:

```
gaussdb=# SELECT '52093.89'::money::numeric::float8;
  float8
----------
 52093.89
(1 row)
```

When two values of the money type are divided, the result is of the double precision type (a pure number, not the money type). The currency units offset each other during the operation.

1.3.3 Boolean Types

Table 1-7 lists the Boolean types supported by GaussDB.

Table 1-7 Boolean Types

Name	Description	Storage Space	Value
BOOLEAN	Boolean type.	1 byte	• **true** • **false** • **null**: unknown

- Valid literal values for the "true" state include:
  TRUE, **'t'**, **'true'**, **'y'**, **'yes'**, **'1'**, **'TRUE'**, **true**, **'on'**, and all non-zero values.
- Valid literal values for the "false" state include:
  FALSE, **'f'**, **'false'**, **'n'**, **'no'**, **'0'**, **0**, **'FALSE'**, **false**, and **'off'**.

If the query result is of the Boolean type, only **t** or **f** is displayed.

TRUE and **FALSE** are standard expressions, compatible with SQL statements.

Examples

```
-- Create a table.
gaussdb=# CREATE TABLE bool_type_t1
(
    BT_COL1 BOOLEAN,
    BT_COL2 TEXT
)DISTRIBUTE BY HASH(BT_COL2);

-- Insert data.
gaussdb=# INSERT INTO bool_type_t1 VALUES (TRUE, 'sic est');

gaussdb=# INSERT INTO bool_type_t1 VALUES (FALSE, 'non est');

-- View data.
gaussdb=# SELECT * FROM bool_type_t1;
 bt_col1 | bt_col2
---------+---------
 t       | sic est
 f       | non est
(2 rows)

gaussdb=# SELECT * FROM bool_type_t1 WHERE bt_col1 = 't';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 bt_col1 | bt_col2
---------+---------
 t       | sic est
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE bool_type_t1;
```

1.3.4 Character Types

Table 1-8 lists the character types supported by GaussDB. For string operators and related built-in functions, see **Character Processing Functions and Operators**.

Table 1-8 Character types

Name	Description	Storage Space
CHAR(n) CHARACTER(n) NCHAR(n)	The string is fixed-length. If the length is insufficient, empty characters are filled in with blank spaces. **n** indicates the number of bytes. If **n** is not specified, the default value **1** is used.	The maximum value of **n** is **10485760** (10 MB).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
	When **sql_compatibility** is set to **'MYSQL'**, if **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s1'**: ● Type: **n** indicates the character length and the range is [0,10485760]. If the precision **n** is not specified, the default value **1** is used. The string is fixed-length. If the length is insufficient, empty characters are filled in with blank spaces. ● Input: – Data type of table columns and temporary variables: If the number of characters in an input string is within the range, the string can be entered normally. If the value of **sql_mode** contains **strict_trans_tables**, an error is reported. Otherwise, the string is truncated based on the maximum string length specified by **n** and an alarm is generated. ● Output: – Data type of table columns and temporary variables: If the value of **sql_mode** contains **pad_char_to_full_length**, the character string containing spaces at the end is output. Otherwise, the character string without spaces at the end is output. – Function parameter and return value or stored procedure parameter: The length cannot be verified. For example, if the input parameter of a user-defined function is of the CHAR(5) type and a string **'123456'** is input, the parameter can be set without length verification.	The value contains a maximum of 10485760 characters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
	When **sql_compatibility** is set to **'ORA'**: ● CHAR(n) and CHARACTER(n) can explicitly specify the unit of the length parameter **n** after it. The unit can only be byte or char (character). The default unit is byte. ● NCHAR(n) cannot specify the unit of the length parameter **n**. ● In the single-byte character set, even if **char** is specified as the unit, the unit of **n** is still byte. For example, in the SQL_ASCII character set, if **CHAR(1 char)** is specified, an error is reported when a Chinese character is entered. ● The storage space of a type is limited to 10 MB. Therefore, when **CHAR(*n* char)** or **CHARACTER(*n* char)** is specified, it does not mean that the type can contain *n* characters. – When the storage space limit is triggered, the number of characters that can be contained in a type depends on the current character set. For example, if **CHAR(10485760 char)** is specified in the GBK character set, a maximum of 5242880 Chinese characters can be contained. – When a fixed-length character string is padded with spaces, it will be padded up to the maximum storage space limit.	The maximum size is 10 MB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
VARCHAR(n) CHARACTER VARYING(n)	Variable-length string. In PG-compatible mode, **n** indicates the string length. In other compatibility modes, **n** indicates the byte length.	The maximum value of **n** is 10 MB. If **n** is not specified, the maximum storage length is 1 GB – 85 – 4 (space for storing length parameter) – Length of other columns. For example, in a table with columns (a int, b varchar, c int), the maximum length for the varchar column is 1 GB – 85 – 4 (space for storing length parameter) – 4 (length of column **a** int) – 4 (length of column **c** int) = 1,073,741,727.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
	When **sql_compatibility** is set to **'MYSQL'**, if **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s1'**: ● **n** indicates the string length. The value range is [0,10485760]. If the precision **n** is not specified, there is no length limit by default. The length is the same as that of the TEXT type. ● Input: – Data type of table columns and temporary variables: If the number of characters in an input string is within the range, the string can be entered normally. If the value of **sql_mode** contains **strict_trans_tables**, an error is reported. Otherwise, the string is truncated based on the maximum string length specified by **n** and an alarm is generated. ● Output: The original character string is output.	The maximum value of **n** is **10485760**. If **n** is not specified, the maximum storage length is 1 GB – 85 – 4 (space for storing length parameter) – Length of other columns. For example, in a table with columns (a int, b varchar, c int), the maximum length for the varchar column is 1 GB – 85 – 4 (space for storing length parameter) – 4 (length of column **a** int) – 4 (length of column **c** int) = 1,073,741,727.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
	When **sql_compatibility** is set to **'ORA'**: ● You can explicitly specify the unit of the length parameter **n** after it. The unit can only be byte or char (character). The unit is byte by default. ● In the single-byte character set, even if **char** is specified as the unit, the unit of **n** is still byte. For example, in the SQL_ASCII character set, if **VARCHAR(1 char)** is specified, an error is reported when a Chinese character is inserted. ● The storage space of a type is limited to 10 MB. Therefore, when **VARCHAR(*n* char)** or **CHARACTER VARYING(*n* char)** is specified, it does not mean that the type can contain *n* characters. When the storage space limit is triggered, the number of characters that can be contained in a type depends on the current character set. For example, if **VARCHAR(10485760 char)** is specified in the GBK character set, a maximum of 5242880 Chinese characters can be contained.	The maximum size is 10 MB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
VARCHAR2(n)	Variable-length string. It is the alias of the VARCHAR(n) type.	The maximum value of **n** is **10485760** (10 MB). If **n** is not specified, the maximum storage length is 1 GB – 85 – 4 (space for storing length parameter) – Length of other columns. For example, in a table with columns (a int, b varchar2, c int), the maximum length for the varchar2 column is 1 GB – 85 – 4 (space for storing length parameter) – 4 (length of column **a** int) – 4 (length of column **c** int) = 1,073,741,727.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
NVARCHAR2(n)	Variable-length string. In the SQL_ASCII character set, **n** indicates bytes. In the non-SQL_ASCII character set, **n** indicates characters.	The maximum value of **n** is 10 MB. If **n** is not specified, the maximum storage length is 1 GB – 85 – 4 (space for storing length parameter) – Length of other columns. For example, in a table with columns (a int, b nvarchar2, c int), the maximum length for the nvarchar2 column is 1 GB – 85 – 4 (space for storing length parameter) – 4 (length of column **a** int) – 4 (length of column **c** int) = 1,073,741,727.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
TEXT	Variable-length string.	The maximum storage length is 1 GB – 85 – 4 (space for storing length parameter) – Length of other columns. For example, in a table with columns (a int, b text, c int), the maximum length for the text column is 1 GB – 85 – 4 (space for storing length parameter) – 4 (length of column **a** int) – 4 (length of column **c** int) = 1,073,741,727.
CLOB	Big text object.	The maximum size is 1 GB minus 1 byte. However, the size of the column description header and the size of the tuple (less than 1 GB minus 1 byte) where the column is located must also be considered. Therefore, the maximum size of the CLOB type may be less than 1 GB minus 1 byte.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
TINYTEXT MEDIUMTEXT LONGTEXT	When **sql_compatibility** is set to **'MYSQL'**, if **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s1'**, these types can be used. Converts a data type to the TEXT type. The application scenario is the same as that of the TEXT type.	The maximum size is 1 GB minus 1 byte. However, the size of the column description header and the size of the tuple (less than 1 GB minus 1 byte) where the column is located must also be considered. Therefore, the maximum size of the TEXT type may be less than 1 GB minus 1 byte.

NOTE

- In addition to the restriction on the size of each column, the total size of each tuple cannot exceed 1 GB minus 1 byte and is affected by the control header information of the column, the control header information of the tuple, and whether null fields exist in the tuple.
- NCHAR is the alias of the bpchar type, and VARCHAR2(n) is the alias of the VARCHAR(n) type.
- In ORA-compatible mode, the received empty string is converted to null by default.

Table 1-9 lists the two other fixed-length character types supported by GaussDB. The name type exists only for the storage of identifiers in the internal system catalogs and is not intended for use by general users. Its length is currently defined as 64 bytes (63 usable characters plus terminator). The char type uses only one byte of storage. It is internally used in the system catalogs as a simplistic enumeration type.

Table 1-9 Special character types

Name	Description	Storage Space
name	Internal type for object names	64 bytes
"char"	Single-byte internal type	1 byte

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.3.5 Binary Types

Table 1-10 lists the binary types supported by GaussDB.

Table 1-10 Binary types

Name	Description	Storage Space
BLOB	Binary large object (BLOB). Currently, BLOB supports only the following external access APIs: • DBE_LOB.GET_LENGTH • DBE_LOB.READ • DBE_LOB.WRITE • DBE_LOB.WRITE_APPEND • DBE_LOB.COPY • DBE_LOB.ERASE For details about the APIs, see DBE_LOB.	The maximum size is 1 GB minus 1 byte. However, the size of the column description header and the size of the tuple (less than 1 GB minus 1 byte) where the column is located must also be considered. Therefore, the maximum size of the BLOB type may be less than 1 GB minus 1 byte.
	When **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, the BLOB type is mapped to the BYTEA type, and the alias is BYTEA.	For details about the storage specifications, see the BYTEA type.
TINYBLOB MEDIUMBLOB LONGBLOB	Binary large object (BLOB).	For details about the storage specifications, see the BYTEA type.
	This type can be used only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**. The type is mapped to the BYTEA type, and the alias is BYTEA.	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
RAW	Variable-length hexadecimal string.	4 bytes plus the actual binary string. The maximum size is 1 GB minus 1 byte. However, the size of the column description header and the size of the tuple (less than 1 GB minus 1 byte) where the column is located must also be considered. Therefore, the maximum size of this type may be less than 1 GB minus 1 byte.
BYTEA	Variable-length binary string.	4 bytes plus the actual binary string. The maximum size is 1 GB minus 1 byte. However, the size of the column description header and the size of the tuple (less than 1 GB minus 1 byte) where the column is located must also be considered. Therefore, the maximum size of this type may be less than 1 GB minus 1 byte.
BYTEAWITHOUTORDERWITHEQUALCOL	Variable-length binary character string (new type for the encryption feature. If the encryption type of an encrypted column is specified as deterministic encryption, the column type is BYTEAWITHOUTORDERWITHEQUALCOL). The original data type is displayed when an encrypted table is printed by running the meta-command.	4 bytes plus the actual binary string. The maximum value is 1,073,741,771 bytes (1 GB minus 53 bytes).
BYTEAWITHOUTORDERCOL	Variable-length binary character string (new type for the encryption feature. If the encryption type of an encrypted column is specified as random encryption, the column type is BYTEAWITHOUTORDERCOL). The original data type is displayed when an encrypted table is printed by running the meta-command.	4 bytes plus the actual binary string. The maximum value is 1,073,741,771 bytes (1 GB minus 53 bytes).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
_BYTEAWITHOUTORDERWITHEQUALCOL	Variable-length binary string, which is a new type for the encryption feature.	4 bytes plus the actual binary string. The maximum value is 1,073,741,771 bytes (1 GB minus 53 bytes).
_BYTEAWITHOUTORDERCOL	Variable-length binary string, which is a new type for the encryption feature.	4 bytes plus the actual binary string. The maximum value is 1,073,741,771 bytes (1 GB minus 53 bytes).

NOTE

- In addition to the size limit of each column, the total size of each tuple cannot exceed 1 GB minus 1 byte.
- BYTEAWITHOUTORDERWITHEQUALCOL, BYTEAWITHOUTORDERCOL, _BYTEAWITHOUTORDERWITHEQUALCOL, and _BYTEAWITHOUTORDERCOL cannot be directly used to create a table.
- *n* in RAW(*n*) indicates the recommended byte length and is not used to verify the byte length of the input raw type.
- When **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, the TINYBLOB, BLOB, MEDIUMBLOB, and LONGBLOB types are displayed as BYTEA. For example, the table structure query result shows the BYTEA type.

Example:

```
-- Create a table.
gaussdb=# CREATE TABLE blob_type_t1
(
    BT_COL1 INTEGER,
    BT_COL2 BLOB,
    BT_COL3 RAW,
    BT_COL4 BYTEA
) DISTRIBUTE BY REPLICATION;

-- Insert data.
gaussdb=# INSERT INTO blob_type_t1 VALUES(10,empty_blob(),
HEXTORAW('DEADBEEF'),E'\\xDEADBEEF');

-- Query data in the table.
gaussdb=# SELECT * FROM blob_type_t1;
 bt_col1 | bt_col2 | bt_col3  |  bt_col4
---------+---------+----------+------------
      10 |         | DEADBEEF | \xdeadbeef
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE blob_type_t1;

-- Example: TINYBLOB, MEDIUMBLOB, and LONGBLOB types
gaussdb=# CREATE DATABASE gaussdb_m WITH dbcompatibility  'MYSQL';
gaussdb=# \c gaussdb_m
-- Set compatible version control parameters.
gaussdb_m=# SET b_format_version='5.7';
gaussdb_m=# SET b_format_dev_version='s1';
-- Create a table.
gaussdb_m=# CREATE TABLE t1(num int, a tinyblob, b blob, m mediumblob, l longblob);
-- Insert data.
gaussdb_m=# INSERT INTO t1 VALUES (1, 'tinyblobtest', 'blobtest', 'mediumblobtest', 'longblobtest');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Query data in the table.
gaussdb_m=# SELECT * FROM t1;
 num |      a       |    b     |       m        |      l
-----+--------------+----------+----------------+--------------
   1 | tinyblobtest | blobtest | mediumblobtest | longblobtest
(1 row)

-- Drop the table and database.
gaussdb_m=# DROP TABLE t1;
gaussdb_m=# \c postgres;
gaussdb=# DROP DATABASE gaussdb_m;
-- Reset parameters.
gaussdb=# RESET ALL;
```

1.3.6 Date/Time Types

Table 1-11 lists the date/time types that can be used in GaussDB. For the operators and built-in functions of the types, see **Date and Time Processing Functions and Operators**.

NOTE

- If the time format of another database is different from that of GaussDB, modify the value of the **DateStyle** parameter to keep them consistent.
- The current date is expressed according to the ISO standard. The year consists of four digits. For example, 1 A.D. is 0001, 1 B.C. is 0000, and 2 B.C. is -0001. The rest may be deduced by analogy.
- In Oracle-compatible mode, when the GUC parameter **mapping_date_to_datea** is set to **off**, the DATE type is replaced with TIMESTAMP(0) WITHOUT TIME ZONE. When the GUC parameter **mapping_date_to_datea** is set to **on**, the DATE type is replaced with the DATEA type. The differences are as follows:
  - The OID of the DATEA type is different from that of the TIMESTAMP(0) type. The DATEA type is a new data type implemented in GaussDB and has corresponding operators and conversion functions.
  - The value ranges supported by the DATEA and TIMESTAMP(0) types are different.
  - When the DATEA type is input, the string format needs to be verified. Only strings in the format specified by **nls_date_format** can be converted to the DATEA type. The TIMESTAMP(0) type supports only several fixed formats. The DATEA type does not support special value inputs, such as NOW, EPOCH, and TODAY.
  - In terms of driver usage, metadata (including OID and language data type) of the TIMESTAMP(0) type is different from metadata of the DATEA type. When setting precompilation parameters in JDBC, you need to use **SetTimestamp** or **GetTimestamp** for the TIMESTAMP(0) type and use **SetDate** or **GetDate** for the DATEA type.
  - When COPY is used to import data, you can set the **WITH** option to specify the format in which the imported data is parsed. For the TIMESTAMP(0) type, you need to use **TIMESTAMP_FORMAT** to specify it. For the DATEA type, you need to use **DATEA_FORMAT** to specify it. For details, see **Syntax**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-11 Date/Time types

Name	Description	Storage Space
DATE	Date and time. Minimum value: **4714-11-24 BC** (4713 B.C.). Maximum value: **5874897-12-31AD** (5874897 A.D.) After **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, the input and output formats and ranges are different. • Input format: YYYY-MM-DD with separators and YYYYMMDD without separators are supported. – In the scenario where separators are used, the plus sign (+) and colon (:) cannot be used as separators between the year, month, and day of a date. (Some separators are not supported when they are used together. The specifications inherit the existing implementation. Example: **date'2020-12?12'**.) The input format is not affected by the **DateStyle** parameter. – In the scenario where separators are not used, the output result may be incorrect though no error is reported. • Output format: Only YYYY-MM-DD is supported and the format is not affected by the **DateStyle** parameter. • Value range: 4713-01-01 BC to 5874897-12-31 AD. – If the value of **sql_mode** contains **strict_trans_tables**, an error is reported if the value is invalid or exceeds the range. – If the value of **sql_mode** does not contain **strict_trans_tables**, the values	4 bytes (8 bytes in ORA-compatible mode)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
	of year, month, and day can be 0. However, the values will be converted to valid values in the sequence of year, month, and day. For example, '0000-00-10' will be converted to 0002-12-10 BC. If the input is invalid or exceeds the range, a warning message is reported and the value **0000-00-00 00:00:00** is returned. **NOTE** In ORA-compatible mode, if **mapping_date_to_datea** is disabled, the DATE type is replaced with TIMESTAMP(0) WITHOUT TIME ZONE. In ORA-compatible mode, if **mapping_date_to_datea** is enabled, the DATE type is replaced with DATEA.	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
TIME [(p)] [WITHOUT TIME ZONE]	Time within one day (without a time zone). **p** indicates the precision after the decimal point. The value ranges from **0** to **6**. Minimum value: **00:00:00**. Maximum value: **24:00:00**. After **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, they are not limited to the time in a day. For example, they can indicate concepts such as duration and relative time, and the format, range, and precision are different. ● Input format: [D] hh:mi:ss.ff with separators and hhmiss.ff without separators are supported. **D** indicates the number of days. The value is an integer (negative numbers are supported). This parameter is optional. The value is converted into the number of hours on the basis of 24 hours in a day. The value is displayed after hh is added. Only colons (:) can be used as separators. ● Output format: hh:mi:ss.ff. All zeros at the end of the decimal part are automatically ignored during display. ● Value range: –838:59:59.000000 to +838:59:59.000000. – If the value of **sql_mode** contains **strict_trans_tables**, **00:00:00** is returned for invalid input. If the value exceeds the range, an error is reported. – If the value of **sql_mode** does not contain **strict_trans_tables**, **00:00:00** is returned for invalid input. If the input is valid but exceeds the range, the nearest boundary value is returned.	8 bytes

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
	For example, inputting 838:59:59.000001 returns 838:59:59, and inputting –838:59:59.000001 returns –838:59:59. • Precision: **p** indicates the precision after the decimal point. The value ranges from 0 to 6. If **p** is set to a value greater than 6, the value **6** is used. – When it is used as the data type of a table column, the default precision is **0**. – When it is used as an expression (for example, time '10:10:10.123456'), the default precision is **6**.	
TIME [(p)] [WITH TIME ZONE]	Time within one day (with a time zone). **p** indicates the precision after the decimal point. The value ranges from **0** to **6**. Minimum value: **00:00:00+15:59**. Maximum value: **24:00:00–15:59**.	12 bytes

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
TIMESTAMP[(p)] [WITHOUT TIME ZONE]	Date and time (without a time zone). **p** indicates the precision after the decimal point. The value ranges from **0** to **6**. Minimum value: **4714-11-24 BC 00:00:00.000000** (4713 B.C.). Maximum value: **294277-01-09 AD 04:00:54.775807** (294277 A.D.), representing infinity After **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, they will be replaced with the TIMESTAMP[(p)] [WITH TIME ZONE] type. However, the format, range, and precision specifications are different. ● Input format: YYYY-MM-DD hh:mi:ss.ff+timezone with separators and YYYYMMDDhhmiss.ff without separators are supported. – If separators are used, the plus sign (+) and colon (:) cannot be used as separators between the year, month, and day. (Some separators are not supported when they are used together. The specifications inherit the existing implementation. For example, timestamp '2020-12?12 00:00:00'.) Only colons (:) can be used as separators in the time part. The input format is not affected by the **DateStyle** parameter. – In the scenario where separators are not used and the input is incomplete, the output result may be incorrect though no error is reported. – If the year has only two digits, 00–69 corresponds to 2000–2069, and 70–99 corresponds to 1970–1999.	8 bytes

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
	– If the year contains only one digit, for example, timestamp '1-1-1 00:00:00', the result is '0001-01-01 00:00:00'. ● Output format: Only YYYY-MM-DD hh:mi:ss.ff is supported and the format is not affected by the **DateStyle** parameter. All zeros at the end of the decimal part of the time are automatically ignored. ● Range – If the value of **sql_mode** contains **strict_trans_tables**, the value range is the same as that of TIMESTAMP[(p)] WITH TIME ZONE. If the value is invalid or exceeds the range, an error is reported. – If the value of **sql_mode** does not contain **strict_trans_tables**, the values of year, month, and day can be 0. However, the values will be converted to valid values in the sequence of year, month, and day. For example, '0000-00-10 00:00:00' will be converted to 0002-12-10 00:00:00 BC. If the input is invalid or exceeds the range, a warning message is reported and the value **0000-00-00 00:00:00** is returned. ● Precision: **p** indicates the precision after the decimal point. The value ranges from 0 to 6. If **p** is set to a value greater than 6, the value **6** is used. – When it is used as the data type of a table column, the default precision is **0**. – When it is used as an expression (for example, timestamp '2000-01-01 00:00:00.123456'), the default precision is **6**.	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
	• Time zone: The value ranges from –15:59 to +15:59. You can also use the time zone name (see the system view **PG_TIMEZONE_NAMES**). The time zone specifications are the same as those of the TIMESTAMP[(p)] WITH TIME ZONE type. Converts the time value of a specified time zone to the time of the UTC-0 time zone. When it is displayed, the time in the UTC-0 time zone is converted to the time in the current time zone of the server. You can run **SHOW TIME ZONE** to view the time zone of the server. Therefore, when you use the SET TIME ZONE statement to change the time zone of the server, the displayed result also changes.	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
TIMESTAMP[(p)] [WITH TIME ZONE]	Date and time (with time zone). TIMESTAMP is also called TIMESTAMPTZ. **p** indicates the precision after the decimal point. The value ranges from **0** to **6**. Minimum value: **4714-11-24 BC 00:00:00.000000 +0** (4713 B.C.). Maximum value: **294277-01-09 AD 04:00:54.775807 +0** (294277 A.D.), representing infinity. Note that the original date and time, the date and time converted to the UTC-0 time zone, and the date and time converted to the current server time zone cannot be greater than 294277-01-09 AD 04:00:54.775807 or less than 4714-11-24 BC 00:00:00.000000. Otherwise, an error may be reported during usage, indicating that the date and time are out of range. Time zone update: In some countries or regions, the time zone information is frequently updated due to political, economic, and war factors. Therefore, the database system needs to modify the time zone file to ensure that the time is correct. Currently, the GaussDB time zone type involves only timestamp with timezone. When a new time zone file takes effect, the existing data is not changed, and the new data is adjusted based on the time zone file information.	8 bytes
SMALLDATETIME	Date and time (without a time zone). The precision level is minute. 30s to 59s are rounded into one minute. Minimum value: **4714-11-24 BC 00:00:00** (4713 B.C.). Maximum value: **294277-01-09 AD 04:00:00** (294277 A.D.).	8 bytes

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
INTERVAL DAY (l) TO SECOND (p)	Specifies the time interval (*X* days *X* hours *X* minutes *X* seconds). • **l**: indicates the precision of days. The value ranges from 0 to 6. For compatibility, the precision functions are not supported. • **p**: indicates the precision of seconds. The value ranges from 0 to 6. The digit 0 at the end of a decimal number is not displayed.	16 bytes
INTERVAL [FIELDS] [(p)]	Time interval. • **FIELDS**: **YEAR**, **MONTH**, **DAY**, **HOUR**, **MINUTE**, **SECOND**, **DAY TO HOUR**, **DAY TO MINUTE**, **DAY TO SECOND**, **HOUR TO MINUTE**, **HOUR TO SECOND**, or **MINUTE TO SECOND**. • **p**: indicates the precision of seconds. The value ranges from 0 to 6. **p** takes effect only when **FIELDS** is set to **SECOND**, **DAY TO SECOND**, **HOUR TO SECOND**, or **MINUTE TO SECOND**. The digit 0 at the end of a decimal number is not displayed.	12 bytes
reltime	Relative time interval. • The format is as follows: X years X mons X days XX:XX:XX. • The Julian calendar is used. It specifies that a year has 365.25 days and a month has 30 days. The relative time interval needs to be calculated based on the input value.	4 bytes
abstime	Date and time. • Format: YYYY-MM-DD hh:mi:ss +timezone. • The value range is from 1901-12-13 20:45:53 to 2038-01-18 23:59:59 in the current time zone, with precision of seconds. If the inserted data exceeds the range, the data is converted to an invalid value for storage.	4 bytes

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
datetime[(p)]	Date and time. This parameter takes effect only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**. The datetime[(p)] type is replaced by the TIMESTAMP[(p)] WITHOUT TIME ZONE type, but the format, range, precision, and time zone processing specifications are different. ● Input format: YYYY-MM-DD hh:mi:ss.ff+timezone with separators and YYYYMMDDhhmiss.ff without separators are supported. – If separators are used, the plus sign (+) and colon (:) cannot be used as separators between the year, month, and day. (Some separators are not supported when they are used together. The specifications inherit the existing implementation. For example, **datetime '2020-12?12 00:00:00'**.) Only colons (:) can be used as separators in the time part. The input format is not affected by the **DateStyle** parameter. – In the scenario where separators are not used and the input is incomplete, the output result may be incorrect though no error is reported. – If the year has only two digits, 00–69 corresponds to 2000–2069, and 70–99 corresponds to 1970–1999. – If the year contains only one digit, for example, datetime '1-1-1 00:00:00', the result is '0001-01-01 00:00:00'. ● Output format: Only YYYY-MM-DD hh:mi:ss.ff is supported and the format is not affected by the **DateStyle** parameter. All zeros at	8 bytes

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
	the end of the decimal part of the time are automatically ignored. • Range – If the value of **sql_mode** contains **strict_trans_tables**, the value range is the same as that of TIMESTAMP[(p)] WITHOUT TIME ZONE. If the value is invalid or exceeds the range, an error is reported. – If the value of **sql_mode** does not contain **strict_trans_tables**, the values of year, month, and day can be 0. However, the values will be converted to valid values in the sequence of year, month, and day. For example, '0000-00-10 00:00:00' will be converted to 0002-12-10 00:00:00 BC. If the input is invalid or exceeds the range, a warning message is reported and the value **0000-00-00 00:00:00** is returned. • Precision: **p** indicates the precision after the decimal point. The value ranges from 0 to 6. If the specified precision exceeds 6, the value **6** is used. – When it is used as the data type of a table column, the default precision is **0**. – When it is used as an expression (for example, datetime '2000-01-01 00:00:00.123456'), the default precision is **6**. • Time zone: The value ranges from –15:59 to +15:59. You can also use the time zone name (see the system view **PG_TIMEZONE_NAMES**). The time zone specifications are the same as those of the TIMESTAMP[(p)] WITH TIME ZONE type. It converts the time	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
	value of a specified time zone to the time of the current time zone of the server. You can run **SHOW TIME ZONE** to view the time zone of the server. If the converted date and time are saved to the table, the time zone is not converted when the time zone of the server changes.	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
year[(w)]	Year. It takes effect only after **sql_compatibility** is set to **'MYSQL'**. ● Input formats: – A string of four digits, ranging from '1901' to '2155' (that is, the years from 1901 to 2155). – Four digits, ranging from 1901 to 2155 (that is, the years from 1901 to 2155). – A string of one or two digits, ranging from '0' to '99'. The values '0' to '69' indicate the years from 2000 to 2069, and the values '70' to '99' indicate the years from 1970 to 1999. **'0'** and **'00'** indicate the year 2000. – One or two digits, ranging from **0** to **99**. The values **1** to **69** indicate the years from 2001 to 2069, and **70** to **99** indicate the years from 1970 to 1999. The value **0** indicates 0000. – Return values of other time functions, for example, now(). ● Output format: Only the YYYY format is supported. **w** indicates the number of digits in the output format. Only four digits are supported. If not specified, the default value is **4**. ● Value range: **1901** to **2155**. – If the value of **sql_mode** contains **strict_trans_tables**, an error is reported if the value is invalid or exceeds the range. – If the value of **sql_mode** does not contain **strict_trans_tables**, the invalid value is converted to **0000** and inserted, and an alarm is generated.	1 byte

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space
DATEA	Date and time. It takes effect only after **sql_compatibility** is set to **'ORA'**. Minimum value: 4712-01-01 00:00:00 B.C. Maximum value: 9999-12-31 23:59:59 A.C. Input format: The input format must comply with the format specified by **nls_date_format**. For details about the formats supported by **nls_date_format**, see **Table 1-12**. • You can use the following special characters or strings enclosed in double quotation marks as the separator: - / , . % : • In the scenario where separators are not used and the input is incomplete, the output result may be incorrect though no error is reported. • If the input format contains *YY*, datea attempts to parse the two-digit year first. If the parsing fails, datea attempts to parse the 4-digit year. The final error information is returned when the four-digit year fails to be parsed. Output format: Only the YYYY-MM-DD HH24:MI:SS AD/BC format is supported and is not affected by the **DateStyle** parameter. Value range: 4712-01-01 00:00:00 B.C. to 9999-12-31 23:59:59 A.C. If an invalid value is entered or the entered value exceeds the range, an error is reported. Precision: The precision is not accurate to the decimal point, with the minimum precision being seconds.	8 bytes

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space

	NOTE
	• It replaces the DATE type when **mapping_date_to_datea** is enabled in ORA-compatible mode.
	• The DATEA type is used for internal function implementation. You are advised not to use it directly.
	• The DATEA type cannot be used as a user-defined range type.
	• If the input format does not comply with the format specified by **nls_date_format**, the following results may occur:
	  • An error message is displayed, indicating that the input format does not meet the requirements.
	  • The input is successful, but the input time is not as expected.	

NOTE

- The data of the time type automatically ignores all zeros at the end of the data when it is displayed.
- The default value of **p** is **6**.
- For the INTERVAL type, the date and time are stored in the int32 and double types in the system. Therefore, the value ranges of the two types are the same as those of the corresponding data type.
- If the insertion time is out of the range, the system may not report an error, but the behavior may not be normal.
- If the values of **a_format_version** and **a_format_dev_version** are **10c** and **s1**, the default DATE value is determined by the following:
  - Year: returned through SYSDATE
  - Month: returned through SYSDATE
  - Day: 01 (first day of the month)
  - Hour, minute, and second: all 0

Example:

```
-- Create a table.
gaussdb=# CREATE TABLE date_type_tab(coll date);

-- Insert data.
gaussdb=# INSERT INTO date_type_tab VALUES (date '12-10-2010');

-- View data.
gaussdb=# SELECT * FROM date_type_tab;
    coll
------------
  2010-12-10
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE date_type_tab;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a table.
gaussdb=# CREATE TABLE time_type_tab (da time without time zone ,dai time with time zone,dfgh
timestamp without time zone,dfga timestamp with time zone, vbg smalldatetime);

-- Insert data.
gaussdb=# INSERT INTO time_type_tab VALUES ('21:21:21','21:21:21 pst','2010-12-12','2013-12-11
pst','2003-04-12 04:05:06');

-- View data.
gaussdb=# SELECT * FROM time_type_tab;
    da    |     dai     |        dfgh         |          dfga          |         vbg
----------+-------------+---------------------+------------------------+---------------------
 21:21:21 | 21:21:21-08 | 2010-12-12 00:00:00 | 2013-12-11 16:00:00+08 | 2003-04-12 04:05:00
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE time_type_tab;

-- Create a table.
gaussdb=# CREATE TABLE day_type_tab (a int,b INTERVAL DAY(3) TO SECOND (4));

-- Insert data.
gaussdb=# INSERT INTO day_type_tab VALUES (1, INTERVAL '3' DAY);

-- View data.
gaussdb=# SELECT * FROM day_type_tab;
 a |   b
---+--------
 1 | 3 days
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE day_type_tab;

-- Create a table.
gaussdb=# CREATE TABLE year_type_tab(a int, b interval year (6));

-- Insert data.
gaussdb=# INSERT INTO year_type_tab VALUES(1,interval '2' year);

-- View data.
gaussdb=# SELECT * FROM year_type_tab;
 a |    b
---+---------
 1 | 2 years
(1 row)

gaussdb=# SELECT TIME 'allballs';
   time
----------
 00:00:00
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE year_type_tab;

-- Example of the datea data type in Oracle-compatible mode when mapping_date_to_datea is set to on.
-- Create a database in Oracle-compatible mode.
gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA';
CREATE DATABASE

-- Switch to the Oracle-compatible database.
gaussdb=# \c ora_db

-- View the mapping_date_to_datea parameter.
ora_db=# SHOW mapping_date_to_datea;
 mapping_date_to_datea
-----------------------
 on
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)

-- Create a table.
ora_db=# CREATE TABLE datea_type_tab(a int, b date);

-- Set the nls_date_format parameter.
ora_db=# SET nls_date_format='YYMMDD HH:MI:SS';
SET

-- Insert data.
ora_db=# INSERT INTO datea_type_tab VALUES(1, '20210512 05:52:23'::date); -- YY is parsed based on a 4-
digit year.
INSERT 0 1
ora_db=# INSERT INTO datea_type_tab VALUES(2, '201012 11:12:13'::date); -- YY is parsed based on a 2-
digit year.
INSERT 0 1

ora_db=# INSERT INTO datea_type_tab VALUES(3, '251126 00:00:00'::date); -- The final error information is
generated when YY fails to be parsed based on a 4-digit year.
ERROR:  The data of "month = 26" is invalid. The value must be between 1 and 12.
LINE 1: INSERT INTO datea_type_tab VALUES(3, '251126 00:00:00'::date...
                                             ^
-- View data.
ora_db=# SELECT * FROM datea_type_tab ORDER BY a;
 a |           b
---+------------------------
 1 | 2021-05-12 05:52:23 AD
 2 | 2020-10-12 11:12:13 AD
(2 rows)

-- Drop the table.
ora_db=# DROP TABLE datea_type_tab;
ora_db=# \c postgres

-- Drop the database.
gaussdb=# DROP DATABASE ora_db;
DROP DATABASE

-- Example of the datetime and timestamp data types in MySQL-compatible mode.
-- Create a database in MySQL-compatible mode.
-- Switch to the database in MySQL-compatible mode.
gaussdb=# CREATE DATABASE gaussdb_m dbcompatibility='MYSQL';
gaussdb=# \c gaussdb_m;

-- Set compatible version control parameters.
gaussdb_m=# SET b_format_version = '5.7';
gaussdb_m=# SET b_format_dev_version = 's1';

-- Create a table.
gaussdb_m=# CREATE TABLE datetime_typ_tab(col1 datetime, col2 timestamp);

-- Insert data.
gaussdb_m=# INSERT INTO datetime_typ_tab VALUES ('2003-04-12 04:05:06+09:00', '2003-04-12
04:05:06+09:00');

-- View data.
gaussdb_m=# SELECT * FROM datetime_typ_tab;
        col1         |        col2
---------------------+---------------------
 2003-04-12 03:05:06 | 2003-04-12 03:05:06
(1 row)

-- Drop the table.
gaussdb_m=# DROP TABLE datetime_typ_tab;

-- Create a table.
gaussdb_m=# CREATE TABLE year_typ_tab(col1 year, col2 year(4));

-- Insert data.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb_m=# INSERT INTO year_typ_tab VALUES ('2023', now());

-- View data.
gaussdb_m=# SELECT * FROM year_typ_tab;
 col1 | col2
------+------
 2023 | 2023
(1 row)

-- Delete the table and database.
gaussdb_m=# DROP TABLE year_typ_tab;
gaussdb_m=# \c postgres;
gaussdb=# DROP DATABASE gaussdb_m;

-- Reset parameters.
gaussdb=# RESET ALL;
```

Table 1-12 Formats supported by nls_date_format

Format	Description
A.D.	A.D.
AD	A.D.
A.M.	Morning.
AM	Morning.
B.C.	B.C.
BC	B.C.
CC	Century in a 2-digit format. The 21st century starts from 2001-01-01.
DAY	Full uppercase date name (padded to nine characters).
DL	Returns a specified long date format.
DS	Returns a specified short date format.
DDD	Day of the year (001–366).
DD	Day of the month (01–31).
DY	Abbreviated uppercase date name (three characters).
D	Day of the week (1–7; Sunday is 1).
FX	Forces a function to parse or format a date strictly according to the specified format.
HH24	Hour of the day (00–23).
HH12	Hour of the day (01–12).
HH	Hour of the day (01–12).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Format	Description
IW	ISO week number of the year (the first week contains the first Thursday).
IYYY	ISO year (with four digits or more).
IYY	Last three digits of the ISO year.
IY	Last two digits of the ISO year.
I	Last one digit of the ISO year.
J	Julian day (number of days after January 1, 4713 B.C.).
MI	Minutes (00–59).
MM	Month number (01–12).
MONTH	Full uppercase month name (padded to nine characters).
MON	Abbreviated uppercase month name (three characters).
P.M.	Afternoon.
PM	Afternoon.
Q	Quarter.
RM	Roman numeral representation of the month (I–XII, where I = JAN).
RRRR	Returns the four-digit year of a date.
RR	Returns the two-digit year of a date.
SSSSS	Seconds since midnight (0–86399).
SCC	Century, B.C.
SS	Seconds (00–59).
SYEAR	Returns the year as a string, with a hyphen (-) for B.C.
SYYYY	Returns the year as a number, with a hyphen (-) for B.C.
TS	Returns a specified time format.
WW	Week number of the year (1–53; the first week starts on the first day of the year).
W	Week number of the month (1–5; the first week starts on the first day of the month).
Y,YYY	Year with a comma (with four digits or more).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Format	Description
YEAR	Returns the year as a string.
YYYY	Year (with four digits or more).
YYY	Last three digits of the year.
YY	Last two digits of the year.
Y	Last one digit of the year.

NOTE

In GaussDB, the value of **nls_date_format** can be suffixed with **TH**. This suffix is used to skip two bytes in the specified part of the format when the input time string is parsed. The formats that support the **TH** suffix include *HH*, *HH12*, *HH24*, *MI*, *SS*, *SSSSS*, *MM*, *DDD*, *DD*, *D*, *Y,YYY*, *YYYY*, *YYY*, *YY*, *Y*, *RRRR*, *RR*, *J*, and *SYYYY*. If the value of the **nls_date_format** parameter contains the **TH** suffix and the suffix is not at the end of the string, two bytes must be reserved for the input time. Otherwise, a parsing error occurs or the result is abnormal. The following are examples:

- The parsing is normal.
```
gaussdb=# select to_datea('2000TH-10-24', 'YYYYTH-MM-DD');   -- When TH is placed following YYYY, two bytes (which can be any characters) must be reserved after the input year 2000.
to_datea
------------------------
2000-10-24 00:00:00 AD
(1 row)

gaussdb=# select to_datea('2000TH-10-24', 'YYYYTH-MM-DDTH');   -- When TH is placed following the nls_date_format parameter, the parsing is normal.
       to_datea
------------------------
 2000-10-24 00:00:00 AD
(1 row)
```
- An error occurs during parsing.
```
gaussdb=# select to_datea('2000-10-24', 'YYYYTH-MM-DD');
ERROR:  The data of "month = 0" is invalid. The value must be between 1 and 12.
CONTEXT:  referenced column: to_datea
```
- The result is abnormal.
```
gaussdb=# select to_datea('2000-11-24', 'YYYYTH-MM-DD');   -- When TH is placed following YYYY, the input year 2000 is not followed by two reserved bytes. During parsing, -1 is skipped, and MM is parsed as January. As a result, the result is abnormal.
       to_datea
------------------------
 2000-01-24 00:00:00 AD
(1 row)
```

Date Inputs

Date and time input is accepted in almost any reasonable formats, including ISO 8601 and SQL-compatible. The system allows you to customize the sequence of day, month, and year in the date input. Set the **DateStyle** parameter to **MDY** to select month-day-year interpretation, **DMY** to select day-month-year interpretation, or **YMD** to select year-month-day interpretation.

Remember that any date or time literal input needs to be enclosed with single quotation marks (''), and the syntax is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
type [ ( p ) ] 'value'
```

The **p** that can be selected in the precision statement is an integer, indicating the number of fractional digits in the **seconds** column. **Table 1-13** lists the input formats of the date type.

Table 1-13 Date input formats

Example	Description
1999-01-08	ISO 8601 (recommended format). January 8, 1999 in any format.
January 8, 1999	Unambiguous in any **datestyle** input mode
1/8/1999	January 8 in MDY format. August 1 in DMY format.
1/18/1999	January 18 in MDY format, rejected in other formats.
01/02/03	• January 2, 2003 in MDY format. • February 1, 2003 in DMY format. • February 3, 2001 in YMD format.
1999-Jan-08	January 8 in any mode
Jan-08-1999	January 8 in any mode
08-Jan-1999	January 8 in any mode
99-Jan-08	January 8 in **YMD** mode, else error
08-Jan-99	January 8, except error in YMD format.
Jan-08-99	January 8, except error in YMD format.
19990108	ISO 8601. January 8, 1999 in any format.
990108	ISO 8601. January 8, 1999 in any format.
1999.008	Year and day of year.
J2451187	Julian date.
January 8, 99 BC	Year 99 B.C.

Example:

```
-- Create a table.
gaussdb=# CREATE TABLE date_type_tab(coll date);

-- Insert data.
gaussdb=# INSERT INTO date_type_tab VALUES (date '12-10-2010');

-- View data.
gaussdb=# SELECT * FROM date_type_tab;
        coll
---------------------
 2010-12-10
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- View the date format.
gaussdb=# SHOW datestyle;
 DateStyle
-----------
 ISO, MDY
(1 row)

-- Set the date format.
gaussdb=# SET datestyle='YMD';
SET

-- Insert data.
gaussdb=# INSERT INTO date_type_tab VALUES(date '2010-12-11');

-- View data.
gaussdb=# SELECT * FROM date_type_tab;
        coll
---------------------
 2010-12-10
 2010-12-11
(2 rows)

-- Drop the table.
gaussdb=# DROP TABLE date_type_tab;
```

Time

The time-of-day types are **TIME [(p)] [WITHOUT TIME ZONE]** and **TIME [(p)] [WITH TIME ZONE]**. **TIME** alone is equivalent to **TIME WITHOUT TIME ZONE**.

If a time zone is specified in the input for **TIME WITHOUT TIME ZONE**, it is silently ignored.

For details about the time input types, see **Table 1-14**. For details about time zone input types, see **Table 1-15**.

Table 1-14 Time input types

Example	Description
05:06.8	ISO 8601
4:05:06	ISO 8601
4:05	ISO 8601
040506	ISO 8601
4:05 AM	Same as 04:05. Input hours must be less than or equal to 12.
4:05 PM	Same as 16:05. Input hours must be less than or equal to 12.
04:05:06.789-8	ISO 8601
04:05:06-08:00	ISO 8601
04:05-08:00	ISO 8601
040506-08	ISO 8601

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example	Description
04:05:06 PST	Time zone specified by abbreviation.
2003-04-12 04:05:06 America/New_York	Time zone specified by full name.

Table 1-15 Time zone input types

Example	Description
PST	Abbreviation (for Pacific Standard Time).
America/New_York	Full time zone name.
-8:00	ISO-8601 offset for PST.
-800	ISO-8601 offset for PST.
-8	ISO-8601 offset for PST.

Example:

```
gaussdb=# SELECT time '04:05:06';
   time
----------
 04:05:06
(1 row)

gaussdb=# SELECT time '04:05:06 PST';
   time
----------
 04:05:06
(1 row)

gaussdb=# SELECT time with time zone '04:05:06 PST';
   timetz
-------------
 04:05:06-08
(1 row)
```

Special Values

The special values supported by GaussDB are converted to common date/time values when being read. For details, see **Table 1-16**.

Table 1-16 Special values

Input String	Applicable Type	Description
epoch	date and timestamp	1970-01-01 00:00:00+00 (Unix system time zero).
infinity	timestamp	Infinity, later than any other timestamp.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Input String	Applicable Type	Description
-infinity	timestamp	Infinity, earlier than any other timestamp.
now	date, time, and timestamp	Start time of the current transaction.
today	date and timestamp	Today at midnight.
tomorrow	date and timestamp	Tomorrow at midnight.
yesterday	date and timestamp	Yesterday at midnight.
allballs	time	00:00:00.00 UTC

Example:

```
-- Create a table.
gaussdb=# CREATE TABLE realtime_type_special(col1 varchar(20), col2 date, col3 timestamp, col4 time);

-- Insert data.
gaussdb=# INSERT INTO realtime_type_special VALUES('epoch', 'epoch', 'epoch', NULL);
gaussdb=# INSERT INTO realtime_type_special VALUES('now', 'now', 'now', 'now');
gaussdb=# INSERT INTO realtime_type_special VALUES('today', 'today', 'today', NULL);
gaussdb=# INSERT INTO realtime_type_special VALUES('tomorrow', 'tomorrow', 'tomorrow', NULL);
gaussdb=# INSERT INTO realtime_type_special VALUES('yesterday', 'yesterday', 'yesterday', NULL);

-- View data.
gaussdb=# SELECT * FROM realtime_type_special;
   col1    |        col2         |            col3            |      col4
-----------+---------------------+----------------------------+-----------------
 epoch     | 1970-01-01 00:00:00 | 1970-01-01 00:00:00        |
 now       | 2023-02-27 11:38:13 | 2023-02-27 11:38:13.032815 | 11:38:13.032815
 today     | 2023-02-27 00:00:00 | 2023-02-27 00:00:00        |
 tomorrow  | 2023-02-28 00:00:00 | 2023-02-28 00:00:00        |
 yesterday | 2023-02-26 00:00:00 | 2023-02-26 00:00:00        |
(5 rows)

gaussdb=# SELECT * FROM realtime_type_special WHERE col3 < 'infinity';
   col1    |        col2         |            col3            |      col4
-----------+---------------------+----------------------------+-----------------
 epoch     | 1970-01-01 00:00:00 | 1970-01-01 00:00:00        |
 now       | 2023-02-27 11:38:13 | 2023-02-27 11:38:13.032815 | 11:38:13.032815
 today     | 2023-02-27 00:00:00 | 2023-02-27 00:00:00        |
 tomorrow  | 2023-02-28 00:00:00 | 2023-02-28 00:00:00        |
 yesterday | 2023-02-26 00:00:00 | 2023-02-26 00:00:00        |
(5 rows)

gaussdb=# SELECT * FROM realtime_type_special WHERE col3 > '-infinity';
   col1    |        col2         |            col3            |      col4
-----------+---------------------+----------------------------+-----------------
 epoch     | 1970-01-01 00:00:00 | 1970-01-01 00:00:00        |
 now       | 2023-02-27 11:38:13 | 2023-02-27 11:38:13.032815 | 11:38:13.032815
 today     | 2023-02-27 00:00:00 | 2023-02-27 00:00:00        |
 tomorrow  | 2023-02-28 00:00:00 | 2023-02-28 00:00:00        |
 yesterday | 2023-02-26 00:00:00 | 2023-02-26 00:00:00        |
(5 rows)

gaussdb=# SELECT * FROM realtime_type_special WHERE col3 > 'now';
   col1   |        col2         |        col3         | col4
----------+---------------------+---------------------+------
 tomorrow | 2023-02-28 00:00:00 | 2023-02-28 00:00:00 |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)

gaussdb=# SELECT * FROM realtime_type_special WHERE col3 = 'today';
 col1  |        col2         |        col3         | col4
-------+---------------------+---------------------+------
 today | 2023-02-27 00:00:00 | 2023-02-27 00:00:00 |
(1 row)

gaussdb=# SELECT * FROM realtime_type_special WHERE col3 = 'tomorrow';
   col1   |        col2         |        col3         | col4
----------+---------------------+---------------------+------
 tomorrow | 2023-02-28 00:00:00 | 2023-02-28 00:00:00 |
(1 row)

gaussdb=# SELECT * FROM realtime_type_special WHERE col3 > 'yesterday';
   col1   |        col2         |            col3             |      col4
----------+---------------------+-----------------------------+-----------------
 now      | 2023-02-27 11:38:13 | 2023-02-27 11:38:13.032815 | 11:38:13.032815
 today    | 2023-02-27 00:00:00 | 2023-02-27 00:00:00         |
 tomorrow | 2023-02-28 00:00:00 | 2023-02-28 00:00:00         |
(3 rows)

-- Drop the table.
gaussdb=# DROP TABLE realtime_type_special;
```

Interval Input

The input of **reltime** can be any valid interval in text format. It can be a number (negative numbers and decimals are also allowed) or a specific time, which must be in SQL standard format or ISO-8601 format. In addition, the text input needs to be enclosed with single quotation marks ('').

For details about interval input, see **Table 1-17**.

Table 1-17 Interval input types

Input	Output	Description
60	2 mons	Numbers are used to indicate intervals. The default unit is day. Decimals and negative numbers are allowed. Particularly, a negative interval syntactically means how long before.
31.25	1 mons 1 days 06:00:00	
-365	-12 mons -5 days	
1 years 1 mons 8 days 12:00:00	1 years 1 mons 8 days 12:00:00	Intervals are in POSTGRES format. They can contain both positive and negative numbers and are case-insensitive. Output is a simplified POSTGRES interval converted from the input.
-13 months -10 hours	-1 years -25 days -04:00:00	
-2 YEARS +5 MONTHS 10 DAYS	-1 years -6 mons -25 days -06:00:00	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Input	Output	Description
P-1.1Y10M	-3 mons -5 days -06:00:00	Intervals are in ISO-8601 format. They can contain both positive and negative numbers and are case-insensitive. Output is a simplified POSTGRES interval converted from the input.
-12H	-12:00:00	

Example:

```
-- Create a table.
gaussdb=# CREATE TABLE reltime_type_tab(col1 character(30), col2 reltime);

-- Insert data.
gaussdb=# INSERT INTO reltime_type_tab VALUES ('90', '90');
gaussdb=# INSERT INTO reltime_type_tab VALUES ('-366', '-366');
gaussdb=# INSERT INTO reltime_type_tab VALUES ('1975.25', '1975.25');
gaussdb=# INSERT INTO reltime_type_tab VALUES ('-2 YEARS +5 MONTHS 10 DAYS', '-2 YEARS +5
MONTHS 10 DAYS');
gaussdb=# INSERT INTO reltime_type_tab VALUES ('30 DAYS 12:00:00', '30 DAYS 12:00:00');
gaussdb=# INSERT INTO reltime_type_tab VALUES ('P-1.1Y10M', 'P-1.1Y10M');

-- View data.
gaussdb=# SELECT * FROM reltime_type_tab;
              col1              |                col2
--------------------------------+-----------------------------------
 90                             | 3 mons
 -366                           | -1 years -18:00:00
 1975.25                        | 5 years 4 mons 29 days
 -2 YEARS +5 MONTHS 10 DAYS     | -1 years -6 mons -25 days -06:00:00
 30 DAYS 12:00:00               | 1 mon 12:00:00
 P-1.1Y10M                      | -3 mons -5 days -06:00:00
(6 rows)

-- Drop the table.
gaussdb=# DROP TABLE reltime_type_tab;
```

1.3.7 Geometric Types

Table 1-18 lists the geometric types that can be used in GaussDB. The most fundamental type is the point, which forms the basis for all of the other geometric types.

Table 1-18 Geometric Types

Name	Storage Space	Description	Representation
point	16 bytes	Point on a plane	(x,y)
lseg	32 bytes	Finite line segment	((x1,y1),(x2,y2))
box	32 bytes	Box	((x1,y1),(x2,y2))
path	16 + 16*n* bytes	Closed path (similar to polygon)	((x1,y1),...)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Storage Space	Description	Representation
path	16 + 16*n* bytes	Open path	[(x1,y1),...]
polygon	40 + 16*n* bytes	Polygon (similar to closed path)	((x1,y1),...)
circle	24 bytes	Circle	<(x,y),r> (center point and radius)

A rich set of functions and operators is available in GaussDB to perform various geometric operations, such as scaling, translation, rotation, and determining intersections. For details, see **Geometric Functions and Operators**.

Points

Points are the fundamental two-dimensional building block for geometric types. Values of the point type are specified using either of the following syntaxes:

```
( x , y )
x , y
```

x and **y** are the respective coordinates, as floating-point numbers. The value type of the points is float8.

Points are output using the first syntax.

Example:

```
gaussdb=# SELECT point(1.1, 2.2);
   point
-----------
 (1.1,2.2)
(1 row)
```

Line Segments

Line segments (lseg) are represented by pairs of points. Values of the lseg type are specified using either of the following syntaxes:

```
[ ( x1 , y1 ) , ( x2 , y2 ) ]
( ( x1 , y1 ) , ( x2 , y2 ) )
( x1 , y1 ) , ( x2 , y2 )
x1 , y1   ,   x2 , y2
```

(x1,y1) and **(x2,y2)** are the end points of the line segment. The value type of the points is float8.

Line segments are output using the first syntax.

Example:

```
gaussdb=# SELECT lseg(point(1.1, 2.2), point(3.3, 4.4));
        lseg
-----------------------
 [(1.1,2.2),(3.3,4.4)]
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Boxes

A box is represented by pairs of points that are opposite corners of the box. Values of the box type are specified using either of the following syntaxes:

```
( ( x1 , y1 ) , ( x2 , y2 ) )
( x1 , y1 ) , ( x2 , y2 )
x1 , y1   ,   x2 , y2
```

(x1,y1) and **(x2,y2)** are any two opposite corners of the box. The value type of the points is float8.

Boxes are output using the second syntax.

Any two diagonals can be used. However, the upper right corner endpoint is placed before the lower left corner endpoint for reordering and storage.

Example:

```
gaussdb=# SELECT box(point(1.1, 2.2), point(3.3, 4.4));
        box
---------------------
 (3.3,4.4),(1.1,2.2)
(1 row)
```

Paths

Paths are represented by lists of connected points. Paths can be open or closed. An open path does not connect the start and end points, while a closed path considers the start and end points connected.

Values of the **path** type are specified using any of the following syntaxes:

```
[ ( x1 , y1 ) , ... , ( xn , yn ) ]
( ( x1 , y1 ) , ... , ( xn , yn ) )
( x1 , y1 ) , ... , ( xn , yn )
( x1 , y1   , ... ,   xn , yn )
x1 , y1   , ... ,   xn , yn
```

The points are the end points of the line segments comprising the path. The value type of the points is float8. Square brackets ([]) indicate an open path, while parentheses (()) indicate a closed path. When the outermost parentheses are omitted (as in the third through fifth syntax), the path is assumed to be closed.

Paths are output using the first or second syntax.

Example:

```
gaussdb=# SELECT path(polygon '((0,0),(1,1),(2,0))');
       path
---------------------
 ((0,0),(1,1),(2,0))
(1 row)
```

Polygons

Polygons are represented by lists of points (the vertexes of the polygon). Polygons are very similar to closed paths, but are stored differently and have their own set of support functions.

Values of the polygon type are specified using any of the following syntaxes:

```
( ( x1 , y1 ) , ... , ( xn , yn ) )
( x1 , y1 ) , ... , ( xn , yn )
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
( x1 , y1   , ... ,   xn , yn )
x1 , y1   , ... ,   xn , yn
```

A point indicates the vertex of a polygon. The value type of a point is float8.

Polygons are output using the first syntax.

Example:

```
gaussdb=# SELECT polygon(box '((0,0),(1,1))');
         polygon
--------------------------
 ((0,0),(0,1),(1,1),(1,0))
(1 row)
```

Circle

A circle represented by a center and a radius. Values of the **circle** type are specified using the following syntax:

```
< ( x , y ) , r >
( ( x , y ) , r )
( x , y ) , r
x , y   , r
```

(x,y) is the center point and **r** is the radius of the circle. The value type of the points is float8.

Circles are output using the first syntax.

Example:

```
gaussdb=# SELECT circle(point(0,0),1);
  circle
-----------
 <(0,0),1>
(1 row)
```

1.3.8 Network Address Types

GaussDB provides data types for storing IPv4, IPv6, and MAC addresses, as listed in **Table 1-19**. Compared with pure text types, these types provide input error check and special operations and functions (see **Network Address Functions and Operators**).

Table 1-19 Network Address Types

Name	Storage Space	Description
cidr	7 or 19 bytes	IPv4 or IPv6 networks.
inet	7 or 19 bytes	IPv4 or IPv6 hosts and networks.
macaddr	6 bytes	MAC address.

When sorting inet or classless inter-domain routing (cidr) data types, IPv4 addresses will always sort before IPv6 addresses, including IPv4 addresses encapsulated or mapped to IPv6 addresses, such as ::10.2.3.4 or ::ffff:10.4.3.2.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

cidr

The cidr type holds an IPv4 or IPv6 network address, as described in **Table 1-20**. The format for specifying networks is **address/y** where **address** is the network represented as an IPv4 or IPv6 address, and **y** is the number of bits in the netmask. If **y** is omitted, it is calculated using assumptions from the older classful network numbering system, except it will be at least large enough to include all of the octets written in the input.

Table 1-20 cidr type input examples

cidr Input	cidr Output	abbrev(cidr)
192.168.100.128/25	192.168.100.128/25	192.168.100.128/25
192.168/24	192.168.0.0/24	192.168.0/24
192.168/25	192.168.0.0/25	192.168.0.0/25
192.168.1	192.168.1.0/24	192.168.1/24
192.168	192.168.0.0/24	192.168.0/24
10.1.2	10.1.2.0/24	10.1.2/24
10.1	10.1.0.0/16	10.1/16
10	10.0.0.0/8	10/8
10.1.2.3/32	10.1.2.3/32	10.1.2.3/32
2001:4f8:3:ba::/64	2001:4f8:3:ba::/64	2001:4f8:3:ba::/64
2001:4f8:3:ba:2e0:81ff:fe22:d1f1/128	2001:4f8:3:ba:2e0:81ff:fe22:d1f1/128	2001:4f8:3:ba:2e0:81ff:fe22:d1f1
::ffff:127.0.0.0/120	::ffff:127.0.0.0/120	::ffff:127.0.0.0/120
::ffff:127.0.0.0/128	::ffff:127.0.0.0/128	::ffff:127.0.0.0/128

inet

The inet type stores an IPv4 or IPv6 host address, and optionally its subnet, all in one field. The subnet is represented by the number of network address bits present in the host address (the "netmask"). If the netmask is 32 and the address is an IPv4 address, then the value does not indicate a subnet, only a single host. In IPv6, because the address length is 128 bits, 128 bits specify a unique host address.

The input format for this type is **address/y** where **address** is an IPv4 or IPv6 address and **y** is the number of binary bits in the netmask. If **/y** is omitted, the subnet mask is **32** for an IPv4 address and **128** for an IPv6 address, and the value represents just a single host. On display, the **/y** portion is suppressed if the netmask specifies a single host.

The essential difference between the **inet** and **cidr** data types is that **inet** accepts values with nonzero bits to the right of the netmask, whereas **cidr** does not.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

macaddr

The macaddr type stores MAC addresses, known for example from Ethernet card hardware addresses (although MAC addresses are used for other purposes as well). Input is accepted in the following formats:

```
'08:00:2b:01:02:03'
'08-00-2b-01-02-03'
'08002b:010203'
'08002b-010203'
'0800.2b01.0203'
'08002b010203'
```

These examples would all specify the same address. Data bits **a** to **f** are case-insensitive and are displayed in the first format.

1.3.9 Bit String Types

Bit strings are strings of binary digits (1 and 0). They can be used to store bit masks.

GaussDB supports two bit string types: bit(n) and bit varying(n), where *n* is a positive integer. The maximum value of *n* is **83886080**, which is equivalent to 10 MB.

Data of the bit type must match the length *n*. If the storage length is not *n*, an error is reported. Data of the bit varying type is of variable length up to the maximum length *n*. Longer strings will be rejected. Writing **bit** without a specified length is equivalent to **bit(1)**, while **bit varying** without a specified length means unlimited length.

NOTE

- If you convert a bit string value to bit(*n*), the content that exceeds *n* digits on the right of the bit string will be truncated, or the bit string will be padded with zeros to *n* digits on the right if the bit string value is less than *n* digits. In this case, no error is reported.
- Similarly, if one explicitly casts a bit string value to **bit varying(n)**, the bit string will be truncated on the right if it is more than *n* bits.
- When the ADMS platform driver version 8.1.3-200 or earlier is used, use **::bit varying** to convert the bit type. Otherwise, an error may occur.

```
-- Create a table.
gaussdb=# CREATE TABLE bit_type_t1
(
    BT_COL1 INTEGER,
    BT_COL2 BIT(3),
    BT_COL3 BIT VARYING(5)
) DISTRIBUTE BY REPLICATION;

-- Insert data.
gaussdb=# INSERT INTO bit_type_t1 VALUES(1, B'101', B'00');

-- Specify the type length. An error is reported if an inserted string exceeds this length.
gaussdb=# INSERT INTO bit_type_t1 VALUES(2, B'10', B'101');
ERROR:  bit string length 2 does not match type bit(3)
CONTEXT:  referenced column: bt_col2

-- Specify the type length. Data is converted if it exceeds this length.
gaussdb=# INSERT INTO bit_type_t1 VALUES(2, B'10'::bit(3), B'101');

-- View data.
gaussdb=# SELECT * FROM bit_type_t1;
 bt_col1 | bt_col2 | bt_col3
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
---------+---------+---------
       1 | 101     | 00
       2 | 100     | 101
(2 rows)

-- Drop the table.
gaussdb=# DROP TABLE bit_type_t1;
```

1.3.10 UUID Type

The data type UUID stores universally unique identifiers (UUID) as defined by RFC 4122, ISO/IEC 9834-8:2005, and related standards. This identifier is a 128-bit quantity (16 bytes in size) that is generated by an algorithm chosen to make it very unlikely that the same identifier will be generated by anyone else in the known universe.

Therefore, for distributed systems, these identifiers provide a better uniqueness guarantee than sequence generators, which are only unique within a single database.

A UUID in standard format is composed of 32 lower-case hexadecimal digits, 128 bits in size, and divided into several groups by hyphens, specifically a group of 8 digits, three groups of 4 digits, and a group of 12 digits. An example is as follows:

```
a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11
```

GaussDB also accepts the following alternative forms for input: use of upper-case letters and digits, the standard format surrounded by braces, omitting some or all hyphens, and adding a hyphen after any group of four digits. An example is provided as follows:

```
A0EEBC99-9C0B-4EF8-BB6D-6BB9BD380A11
{a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11}
a0eebc999c0b4ef8bb6d6bb9bd380a11
a0ee-bc99-9c0b-4ef8-bb6d-6bb9-bd38-0a11
```

Output is in the standard format by default.

Examples

```
-- Generate a UUID sequence number.
gaussdb=# SELECT uuid();
                 uuid
--------------------------------------
 846b9d00-172d-c63d-4a03-e8b4700370e0
(1 row)

-- Create a table.
gaussdb=# CREATE TABLE uuid_test(id int, test uuid) DISTRIBUTE BY HASH(test);

-- Use the UUID generated by the system to insert data.
gaussdb=# INSERT INTO uuid_test VALUES(1, uuid()::uuid);

-- Insert data in the example format.
gaussdb=# INSERT INTO uuid_test VALUES(2, 'A0EEBC99-9C0B-4EF8-BB6D-6BB9BD380A11');
gaussdb=# INSERT INTO uuid_test VALUES(3, '{a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11}');
gaussdb=# INSERT INTO uuid_test VALUES(4, 'a0eebc999c0b4ef8bb6d6bb9bd380a11');
gaussdb=# INSERT INTO uuid_test VALUES(5, 'a0ee-bc99-9c0b-4ef8-bb6d-6bb9-bd38-0a11');

-- View data and output the data in the standard format.
gaussdb=# SELECT * FROM uuid_test;
 id |                 test
----+--------------------------------------
  1 | 1b52051c-1731-c63d-4a03-e8b4700370e0
  2 | a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 3 | a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11
 4 | a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11
 5 | a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11
(5 rows)

-- Drop the table.
gaussdb=# DROP TABLE uuid_test;
```

1.3.11 Text Search Types

GaussDB provides two data types to support full-text search. The tsvector type is a document data type optimized by full-text search. The tsquery type indicates the text query type.

tsvector

The tsvector type represents a retrieval unit, usually a textual column within a row of a database table, or a combination of such columns. A tsvector value is a sorted list of distinct lexemes, which are words that have been normalized to merge different variants of the same word. Sorting and deduplication are done automatically during input, as shown below:

```
gaussdb=# SELECT 'a fat cat sat on a mat and ate a fat rat'::tsvector;
                      tsvector
-----------------------------------------------------
 'a' 'and' 'ate' 'cat' 'fat' 'mat' 'on' 'rat' 'sat'
(1 row)
```

To represent lexemes containing spaces or punctuations, enclose them with quotation marks:

```
gaussdb=# SELECT $$the lexeme '    ' contains spaces$$::tsvector;
                 tsvector
-------------------------------------------
 '    ' 'contains' 'lexeme' 'spaces' 'the'
(1 row)
```

Use double dollar signs ($$) to mark entries requiring single quotation marks ('').

```
gaussdb=# SELECT $$the lexeme 'Joe''s' contains a quote$$::tsvector;
                    tsvector
-------------------------------------------------
 'Joe''s' 'a' 'contains' 'lexeme' 'quote' 'the'
(1 row)
```

Optionally, positions can be attached to lexemes:

```
gaussdb=# SELECT 'a:1 fat:2 cat:3 sat:4 on:5 a:6 mat:7 and:8 ate:9 a:10 fat:11 rat:12'::tsvector;
                                  tsvector
-------------------------------------------------------------------------------
 'a':1,6,10 'and':8 'ate':9 'cat':3 'fat':2,11 'mat':7 'on':5 'rat':12 'sat':4
(1 row)
```

A position normally indicates the source word's location in the document. Positional information can be used for ranking. Position values range from 1 to 16383. Values greater than 16383 are automatically set to **16383**. Duplicate positions for the same lexeme are discarded.

Lexemes that have positions can further be labeled with a weight, which can be A, B, C, or D. The default value is **D**, which is not displayed in the output.

```
gaussdb=# SELECT 'a:1A fat:2B,4C cat:5D'::tsvector;
          tsvector
----------------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 'a':1A 'cat':5 'fat':2B,4C
(1 row)
```

Weights are typically used to reflect the document structure, for example, by marking title words differently from body words. Full-text search ranking functions can assign different priorities to different weight markers.

The tsvector type itself does not perform any word normalization. It assumes that the given word has already been properly normalized. The following example is the standard usage of the tsvector type. Example:

```
gaussdb=# SELECT 'The Fat Rats'::tsvector;
     tsvector
--------------------
 'Fat' 'Rats' 'The'
(1 row)
```

For most English text search applications, the preceding words are considered non-normalized, but tsvector does not perform normalization. Therefore, you need to use the to_tsvector function to normalize these words.

```
gaussdb=# SELECT to_tsvector('english', 'The Fat Rats');
   to_tsvector
-----------------
 'fat':2 'rat':3
(1 row)
```

"The" in the preceding example is a stop word. Some words are recognized as stop words due to their high frequency, rendering them ineffective in searches. Consequently, stop words are ignored during the search process, and the to_tsvector function removes them during normalization. However, stop words affect the sequence of words in tsvector (for example, the missing first word in the preceding example), which will affect the ranking result.

tsquery

The tsquery type represents a retrieval condition, stores the terms used for retrieval, and uses the following operators:

- Boolean operators: **&** (AND), **|** (OR), and **!** (NOT).
- Phrase operator (PHRASE): **<->** or **<*n*>**. **<*n*>** indicates that the distance between two words in a document is *n* (leading zeros are allowed). **<->** is equivalent to **<1>**. To parse phrase operators, the GUC parameter **system_function_version** must contain the **parse_phrase_fix** option.

Parentheses can be used to force operator grouping. If there is no parenthesis, **!** (NOT) has the highest priority, and it is followed by **<->** or **<*n*>** (PHRASE), then **&** (AND), and finally **|** (OR).

```
gaussdb=# SELECT 'fat & rat'::tsquery;
    tsquery
----------------
 'fat' & 'rat'
(1 row)

gaussdb=# SELECT 'fat & (rat | cat)'::tsquery;
          tsquery
---------------------------
 'fat' & ( 'rat' | 'cat' )
(1 row)

gaussdb=# SELECT 'fat & rat & ! cat'::tsquery;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
        tsquery
------------------------
 'fat' & 'rat' & !'cat'
(1 row)

gaussdb=# SELECT 'fat <-> cat'::tsquery;
     tsquery
-----------------
 'fat' <-> 'cat'
(1 row)
```

Words in tsquery can be marked with one or more weight letters, which restricts them to match only tsvector lexemes with one of those weights. If the corresponding lexeme of tsvector does not have position information, the weight restriction in tsquery is ignored.

```
gaussdb=# SELECT 'fat:ab & cat'::tsquery;
     tsquery
------------------
 'fat':AB & 'cat'
(1 row)
gaussdb=# SELECT 'fat:1C'::tsvector @@ 'fat:ab'::tsquery AS RESULT;
 result
--------
 f
(1 row)

gaussdb=# SELECT 'fat'::tsvector @@ 'fat:ab'::tsquery AS RESULT;
 result
--------
 t
(1 row)
```

In addition, lexemes in a tsquery can be labeled with asterisks (*) to specify prefix matching:

```
gaussdb=# SELECT 'super:*'::tsquery;
  tsquery
-----------
 'super':*
(1 row)
```

This query will match any word in a tsvector that begins with "super".

Note that to_tsquery performs normalization in the same way as other functions. The following example returns true:

```
gaussdb=# SELECT to_tsvector( 'seriousness' ) @@ to_tsquery( 'series:*' ) AS RESULT;
  result
----------
 t
(1 row)
```

After the "series" is processed, the "seri" is obtained. In this way, the "seriousness" can be matched.

```
gaussdb=# SELECT to_tsquery('series:*');
 to_tsquery
------------
 'seri':*
(1 row)
```

Note that the tsquery type itself does not perform any word normalization, so you need to normalize the words before converting to the tsquery type. The to_tsquery function is used to perform the following normalization:

```
gaussdb=# SELECT to_tsquery('Fat:ab & Cats');
    to_tsquery
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
------------------
 'fat':AB & 'cat'
(1 row)
```

1.3.12 JSON/JSONB Types

JavaScript Object Notation (JSON) data can be a single scalar, an array, or a key-value pair object. The array and object can be called a container:

- Scalar: a single number, Boolean value, string, or **null**.
- Array: defined in a pair of square brackets ([]), in which elements can be any type of JSON data, and are not necessarily of the same type.
- Object: defined in a pair of braces ({}), in which objects are stored in the format of *key:value*. Each key must be a string enclosed by a pair of double quotation marks (""), and its value can be any type of JSON data. In case of duplicate keys, the last key value will be used.

GaussDB provides the JSON and JSONB data types to store JSON data. In the preceding information:

- The JSON data type stores the complete copy of the input text and parses the data when the data is used. During this process, the entered spaces, duplicate keys, and sequence are retained.
- The JSONB data type stores data in binary format. During parsing, JSONB deletes semantic-irrelevant details and duplicate keys, and sorts key values, so that the data does not need to be parsed again during use.

Both data types are JSON and accept the same string as input. The difference between them lies in the efficiency. Because JSON data type stores a complete copy of the input text, the data must be parsed on every execution. In contrast, JSONB data is stored in its parsed binary form and can be processed faster, though this makes it slightly slower to input due to the conversion mechanism. In addition, because the JSONB data form is normalized, it supports more operations, for example, comparing sizes according to a specific rule. In addition, JSONB also supports indexing, which is a significant advantage.

Input Format

An input must be a JSON-compliant string, which is enclosed in single quotation marks ('').

null (null-json): The value can only be **null** in lowercase.

```
gaussdb=# SELECT 'null'::json;   -- suc
 json
------
 null
(1 row)

gaussdb=# SELECT 'NULL'::jsonb;  -- err
ERROR:  invalid input syntax for type json
```

Number (num-json): The value can be a positive or negative integer, decimal fraction, or 0. The scientific notation is supported.

```
gaussdb=# SELECT '1'::json;
 json
------
 1
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)

gaussdb=# SELECT '-1.5'::json;
 json
------
 -1.5
(1 row)

gaussdb=# SELECT '-1.5e-5'::jsonb, '-1.5e+2'::jsonb;
   jsonb   | jsonb
-----------+-------
 -0.000015 | -150
(1 row)

gaussdb=# SELECT '001'::json, '+15'::json, 'NaN'::json;  -- Redundant leading zeros, plus signs (+), NaN, and infinity are not supported.
ERROR:  invalid input syntax for type json
```

Boolean (bool-json): The value can only be **true** or **false** in lowercase.

```
gaussdb=# SELECT 'true'::json;
 json
------
 true
(1 row)

gaussdb=# SELECT 'false'::jsonb;
 jsonb
-------
 false
(1 row)
```

String (str-json): The value must be a string enclosed in double quotation marks ("").

```
gaussdb=# SELECT '"a"'::json;
 json
------
 "a"
(1 row)

gaussdb=# SELECT '"abc"'::jsonb;
 jsonb
-------
 "abc"
(1 row)
```

Array (array-json): Arrays are enclosed in square brackets ([]). Elements in the array can be any valid JSON data, and are unnecessarily of the same type.

```
gaussdb=# SELECT '[1, 2, "foo", null]'::json;
        json
--------------------
 [1, 2, "foo", null]
(1 row)

gaussdb=# SELECT '[]'::json;
 json
------
 []
(1 row)

gaussdb=# SELECT '[1, 2, "foo", null, [[]], {}]'::jsonb;
             jsonb
--------------------------------
 [1, 2, "foo", null, [[]], {}]
(1 row)
```

Object (object-json): The value is enclosed in braces ({}). The key must be a JSON-compliant string, and the value can be any valid JSON string.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT '{}'::json;
 json
------
 {}
(1 row)

gaussdb=# SELECT '{"a": 1, "b": {"a": 2,  "b": null}}'::json;
                json
-------------------------------------
 {"a": 1, "b": {"a": 2,  "b": null}}
(1 row)

gaussdb=# SELECT '{"foo": [true, "bar"], "tags": {"a": 1, "b": null}}'::jsonb;
                        jsonb
-----------------------------------------------------
 {"foo": [true, "bar"], "tags": {"a": 1, "b": null}}
(1 row)
```

NOTICE

- Note that **'null'::json** and **null::json** are different, which are similar to the strings **str="null"** and **str=null**.
- For numbers, when scientific notation is used, JSONB expands them, while JSON stores an exact copy of the input text.

JSONB Advanced Features

- Precautions
  - It cannot be used as a partition key.
  - Foreign tables are not supported.

The main difference between JSON and JSONB lies in the storage mode. JSONB stores the parsed binary format, which reflects the JSON hierarchy and facilitates direct access. Therefore, JSONB has more advanced features than JSON.

- Format normalization
  - After the input object-json string is parsed into JSONB binary format, semantically irrelevant details, such as spaces, are automatically removed.

    ```
    gaussdb=# SELECT '   [1, " a ", {"a"    :1     }]   '::jsonb;
           jsonb
    ----------------------
     [1, " a ", {"a": 1}]
    (1 row)
    ```

  - For object-json, duplicate key-values are deleted and only the last key-value is retained. For example:

    ```
    gaussdb=# SELECT '{"a" : 1, "a" : 2}'::jsonb;
      jsonb
    ----------
     {"a": 2}
    (1 row)
    ```

  - For object-json, key-values will be re-sorted. The collation rule is as follows: 1. Longer key-values are sorted last. 2. If the key-values are of the same length, the key-values with a larger ASCII code are sorted after the key-values with a smaller ASCII code:

    ```
    gaussdb=# SELECT '{"aa" : 1, "b" : 2, "a" : 3}'::jsonb;
              jsonb
    ---------------------------
    ```

```
{"a": 3, "b": 2, "aa": 1}
(1 row)
```

- Size comparison

  Format normalization ensures that only one form of JSONB data exists in the same semantics. Therefore, sizes may be compared according to a specific rule.

  - First, type comparison: object-jsonb > array-jsonb > bool-jsonb > num-jsonb > str-jsonb > null-jsonb
  - Content comparison if the data type is the same:
    - str-jsonb: The default text sorting rule of the database is used for comparison. A positive value indicates greater than, a negative value indicates less than, and **0** indicates equal.
    - num-jsonb: numeric comparison
    - bool-jsonb: **true** > **false**
    - array-jsonb: long elements > short elements. If the lengths are the same, compare each element in sequence.
    - object-jsonb: If the length of a key-value pair is longer than that of a short key-value pair, the keys are compared first, and then the values are compared.

      NOTE

      For comparison within the object-jsonb type, the final result after format sorting is used for comparison. Therefore, the comparison result may not be intuitive compared with the direct input.

- Creating indexes and primary keys
  - B-tree index

    B-tree indexes and primary keys can be created for the JSONB type.
- Inclusion and existence

  Querying whether a JSON contains some elements or whether some elements exist in a JSON is an important capability of JSONB.

```
-- Simple scalar/primitive values contain only the identical value.
gaussdb=# SELECT '"foo"'::jsonb @> '"foo"'::jsonb;
 ?column?
----------
 t
(1 row)

-- The array on the left contains the character string on the right.
gaussdb=# SELECT '[1, "aa", 3]'::jsonb ? 'aa';
 ?column?
----------
 t
(1 row)

-- The array on the left contains all elements of the array on the right, regardless of the sequence and repetition.
gaussdb=# SELECT '[1, 2, 3]'::jsonb @> '[1, 3, 1]'::jsonb;
 ?column?
----------
 t
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- The object-json on the left contains all key-value pairs of object-json on the right.
gaussdb=# SELECT '{"product": "PostgreSQL", "version": 9.4, "jsonb":true}'::jsonb @>
'{"version":9.4}'::jsonb;
 ?column?
----------
 t
(1 row)

-- The array on the left does not contain all elements of the array on the right. The three elements on
the left are 1, 2, and [1,3], but the elements on the right are 1 and 3.
gaussdb=# SELECT '[1, 2, [1, 3]]'::jsonb @> '[1, 3]'::jsonb;
 ?column?
----------
 f
(1 row)

gaussdb=# SELECT '{"foo": {"bar": "baz"}}'::jsonb @> '{"bar": "baz"}'::jsonb;
 ?column?
----------
 f
(1 row)
```

For details about the operators, see **JSON/JSONB Functions and Operators**.

- Functions and operators

  For details about the functions and operators supported by the JSON/JSONB type, see **JSON/JSONB Functions and Operators**.

  NOTE

  GINs can be used to effectively search for keys or key-value pairs that appear in a large number of JSONB documents (datums). Two GIN operator classes (jsonb_ops and jsonb_hash_ops) are provided for different performance and flexibility choices. The default GIN operator class supports @>, <@, ?, ?&, and ?| operator query. The non-default GIN operator class jsonb_hash_ops supports only the @> and <@ operators.

  Internal functions used by GIN indexes and JSON/JSONB aggregate functions:

  - gin_compare_jsonb
  - gin_consistent_jsonb
  - gin_consistent_jsonb_hash
  - gin_extract_jsonb
  - gin_extract_jsonb_hash
  - gin_extract_jsonb_query
  - gin_extract_jsonb_query_hash
  - gin_triconsistent_jsonb
  - gin_triconsistent_jsonb_hash

1.3.13 HLL Type

HyperLogLog (HLL) is an approximation algorithm for efficiently counting the number of distinct values in a data set. It features faster computing and lower space usage. You only need to store HLL data structures, instead of data sets. When new data needs to be collected, the database performs hash calculation on the data and inserts the data into the HLL. Then, you can obtain the final result based on the HLL.

Table 1-21 compares HLL with other algorithms.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-21 Comparison between HLL and other algorithms

Item	Sorting Algorithm	Hash Algorithm	HLL
Time complexity	O(nlogn)	O(n)	O(n)
Space complexity	O(n)	O(n)	log(logn)
Error rate	0	0	≈0.8%
Storage space requirement	Size of original data	Size of original data	The maximum size is 16 KB by default.

HLL has advantages in both the computing speed and storage efficiency. In terms of time complexity, the sorting algorithm needs O(*nlogn*) time for sorting, and the hash algorithm and HLL need O(*n*) time for full table scanning. However, in terms of storage efficiency, the sorting algorithm and hash algorithm need to store raw data before collecting statistics, which consumes a large amount of storage space. The HLL algorithm does not need to store raw data. It only needs to maintain the HLL data structure, saving a lot of storage space. By default, the maximum space of the HLL data structure is about 16 KB.

NOTICE

- In the current default specifications, the maximum number of distinct values that can be calculated is about 1.1e + 15, and the error rate is about 0.8%. If the calculation result exceeds the maximum, the error rate of the calculation result will increase, or the calculation will fail and an error will be reported.
- When using this feature for the first time, you need to evaluate the distinct values of the service, properly select configuration parameters, and perform verification to ensure that the accuracy meets requirements.
  - By default, the maximum number of distinct values that can be calculated is 1.1e+15. If the number of distinct values is **NaN**, you need to adjust **log2m** or use another algorithm to calculate the number of distinct values.
  - Although the hash algorithm has a low probability of hash conflicts, you are advised to select two or three different hash seeds for verification when using the hash algorithm for the first time. If the calculation results of the seeds are similar, you can select any of them as the hash seed.

Table 1-22 describes main HLL data structures.

Table 1-22 Main HLL data structures

Data Type	Description
HLL	The HLL header is a 27-byte column. By default, the data length ranges from 0 KB to 16 KB. The distinct value can be obtained.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

When you create an HLL data type, 0 to 4 input parameters are supported. The parameter meanings and specifications are the same as those of the hll_empty function. The first parameter is **log2m**, indicating the logarithm of the number of buckets, and its value ranges from 10 to 16. The second parameter is **log2explicit**, indicating the threshold in explicit mode, and its value ranges from 0 to 12. The third parameter is **log2sparse**, indicating the threshold of the Sparse mode, and its value ranges from 0 to 14. The fourth parameter is **duplicatecheck**, indicating whether to enable duplicate check, and its value ranges from 0 to 1. When the input parameter is set to **–1**, the default value of the HLL parameter is used. You can run the **\d** or **\d+** command to view the parameters of the HLL type.

NOTE

When the HLL data type is created, the result varies depending on the input parameter behavior:

- When creating an HLL type, do not set the input parameter or set it to **–1**. Use the default value of the corresponding HLL parameter.
- If a valid value is set for the input parameter, the corresponding HLL parameter uses the input value.
- If the input value is invalid, an error is reported when the HLL type is created.

```
-- Create an HLL table without specifying input parameters.
gaussdb=# CREATE TABLE t1 (id integer, set hll);
gaussdb=# \d t1
      Table "public.t1"
 Column |  Type   | Modifiers
--------+---------+-----------
 id     | integer |
 set    | hll     |

-- Create an HLL table, specify the first two input parameters, and use the default values for the last two input parameters.
gaussdb=# CREATE TABLE t2 (id integer, set hll(12,4));
gaussdb=# \d t2
          Table "public.t2"
 Column |      Type      | Modifiers
--------+----------------+-----------
 id     | integer        |
 set    | hll(12,4,12,0) |

-- Create an HLL table, specify the third input parameter, and use default values for other parameters.
gaussdb=# CREATE TABLE t3(id int, set hll(-1,-1,8,-1));
gaussdb=# \d t3
          Table "public.t3"
 Column |      Type      | Modifiers
--------+----------------+-----------
 id     | integer        |
 set    | hll(14,10,8,0) |

-- When a user creates an HLL table and specifies an invalid input parameter, an error is reported.
gaussdb=# CREATE TABLE t4(id int, set hll(5,-1));
ERROR:  log2m = 5 is out of range, it should be in range 10 to 16, or set -1 as default

-- Drop the created HLL table.
gaussdb=# DROP TABLE t1,t2,t3;
DROP TABLE
```

NOTE

When inserting an HLL object to an HLL table, ensure that the parameters of the HLL type are the same as those of the inserted object. Otherwise, an error is reported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create an HLL table.
gaussdb=# CREATE TABLE t1(id integer, set hll(14));

-- Insert an HLL object to a table. The insertion succeeds because parameter types are consistent.
gaussdb=# INSERT INTO t1 VALUES (1, hll_empty(14,-1));

-- Insert an HLL object to a table. The insertion fails because parameter types are inconsistent.
gaussdb=# INSERT INTO t1(id, set) VALUES (1, hll_empty(14,5));
ERROR:  log2explicit does not match: source is 5 and dest is 10

-- Drop the table.
gaussdb=# DROP TABLE t1;
```

The following describes HLL application scenarios:

- Scenario 1: "Hello World"

  The following example shows how to use the HLL data type:

  ```
  -- Create an HLL table.
  gaussdb=# CREATE TABLE helloworld (id integer, set hll);

  -- Insert an empty HLL to the table.
  gaussdb=# INSERT INTO helloworld(id, set) VALUES (1, hll_empty());

  -- Add a hashed integer to the HLL.
  gaussdb=# UPDATE helloworld SET set = hll_add(set, hll_hash_integer(12345)) WHERE id = 1;

  -- Add a hashed string to the HLL.
  gaussdb=# UPDATE helloworld SET set = hll_add(set, hll_hash_text('hello world')) WHERE id = 1;

  -- Obtain the number of distinct values of the HLL.
  gaussdb=# SELECT hll_cardinality(set) FROM helloworld WHERE id = 1;
   hll_cardinality
  -----------------
                2
  (1 row)

  -- Drop the table.
  gaussdb=# DROP TABLE helloworld;
  ```

- Scenario 2: Collect statistics about website visitors.

  The following example shows how an HLL collects statistics on the number of users visiting a website within a period of time:

  ```
  -- Create a raw data table to show that a user has visited the website at a certain time.
  gaussdb=# CREATE TABLE facts (
          date            date,
          user_id         integer
  );

  -- Create a raw data table to show that a user has visited the website at a certain time.
  gaussdb=# INSERT INTO facts VALUES ('2019-02-20', generate_series(1,100));
  gaussdb=# INSERT INTO facts VALUES ('2019-02-21', generate_series(1,200));
  gaussdb=# INSERT INTO facts VALUES ('2019-02-22', generate_series(1,300));
  gaussdb=# INSERT INTO facts VALUES ('2019-02-23', generate_series(1,400));
  gaussdb=# INSERT INTO facts VALUES ('2019-02-24', generate_series(1,500));
  gaussdb=# INSERT INTO facts VALUES ('2019-02-25', generate_series(1,600));
  gaussdb=# INSERT INTO facts VALUES ('2019-02-26', generate_series(1,700));
  gaussdb=# INSERT INTO facts VALUES ('2019-02-27', generate_series(1,800));

  -- Create another table and specify an HLL column:
  gaussdb=# CREATE TABLE daily_uniques (
     date            date UNIQUE,
     users           hll
  );

  -- Group data by date and insert the data into the HLL:
  gaussdb=# INSERT INTO daily_uniques(date, users)
     SELECT date, hll_add_agg(hll_hash_integer(user_id))
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    FROM facts
    GROUP BY 1;

-- Calculate the numbers of users visiting the website every day.
gaussdb=# SELECT date, hll_cardinality(users) FROM daily_uniques ORDER BY date;
    date    | hll_cardinality
------------+------------------
 2019-02-20 |              100
 2019-02-21 | 200.217913059312
 2019-02-22 |  301.76494508014
 2019-02-23 | 400.862858326446
 2019-02-24 | 502.626933349694
 2019-02-25 | 601.922606454213
 2019-02-26 | 696.602316769498
 2019-02-27 | 798.111731634412
(8 rows)

-- Calculate the number of users who had visited the website in the week from February 20, 2019 to
February 26, 2019.
gaussdb=# SELECT hll_cardinality(hll_union_agg(users)) FROM daily_uniques WHERE date >=
'2019-02-20'::date AND date <= '2019-02-26'::date;
 hll_cardinality
------------------
 696.602316769498
(1 row)

-- Calculate the number of users who had visited the website yesterday but have not visited the
website today:
gaussdb=# SELECT date, (#hll_union_agg(users) OVER two_days) - #users AS lost_uniques FROM
daily_uniques WINDOW two_days AS (ORDER BY date ASC ROWS 1 PRECEDING);  -- The default
compatibility details (M-compatibility) are listed as follows:
    date    | lost_uniques
------------+--------------
 2019-02-20 |            0
 2019-02-21 |            0
 2019-02-22 |            0
 2019-02-23 |            0
 2019-02-24 |            0
 2019-02-25 |            0
 2019-02-26 |            0
 2019-02-27 |            0
(8 rows)

-- Drop the table.
gaussdb=# DROP TABLE facts;
gaussdb=# DROP TABLE daily_uniques;
```

- Scenario 3: The data to be inserted does not meet the requirements of the HLL data structure.

  When inserting data into a column of the HLL type, ensure that the data meets the requirements of the HLL data structure. If the data does not meet the requirements after being parsed, an error will be reported. In the following example, **'E\\1234'** to be inserted does not meet the requirements of the HLL data structure after being parsed. As a result, an error is reported.

```
gaussdb=# CREATE TABLE test(id integer, set hll);
gaussdb=# INSERT INTO test VALUES(1, 'E\\1234');
ERROR:  not a hll type, size=6 is not enough
gaussdb=# DROP TABLE test;
```

1.3.14 Range Types

A range type is a data type that represents the range of a value of an element type (called the subtype of a range). For example, the range of timestamp can be used to indicate the reservation time period of a meeting room. In this case, the data type is tsrange (short for range of timestamp), and the timestamp is the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

subtype. The subtype must be ordered, so that the element value can be clearly specified within, before, after a range.

Range types have the following advantages: They can express multiple element values as a single range value, and clearly express concepts such as range overlapping. For example, time and date ranges used for time arrangement, price ranges, and instrument ranges can be expressed.

Built-in Range

The system provides the following built-in range types:

- int4range: integer range.
- int8range: bigint range.
- numrange: numeric range.
- tsrange: range of timestamp without the time zone.
- tstzrange: range of timestamp with the time zone
- daterange: date range.

In addition, you can customize range types. For details, see **CREATE TYPE**.

Example

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE reservation (room int, during tsrange);
gaussdb=# INSERT INTO reservation VALUES (1108, '[2010-01-01 14:30, 2010-01-01 15:30)');

-- Check whether the range contains the specified data.
gaussdb=# SELECT int4range(10, 20) @> 3;
 ?column?
----------
 f
(1 row)

-- Check whether the two ranges overlap.
gaussdb=# SELECT numrange(11.1, 22.2) && numrange(20.0, 30.0);
 ?column?
----------
 t
(1 row)

-- Upper bound extraction.
gaussdb=# SELECT upper(int8range(15, 25));
 upper
-------
    25
(1 row)

-- Calculate the intersection.
gaussdb=# SELECT int4range(10, 20) * int4range(15, 25);
 ?column?
----------
 [15,20)
(1 row)

-- Check whether the range is empty.
gaussdb=# SELECT isempty(numrange(1, 5));
 isempty
---------
 f
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop the table.
gaussdb=# DROP TABLE reservation;
```

See the complete list of operators and functions on a range type in **Range Functions and Operators**.

> **NOTE**
>
> In ORA-compatible mode, after the **mapping_date_to_datea** parameter is enabled, the DATE type is mapped to the DATEA type. The DATEA type does not support being used as a range type, and the daterange function does not support the DATEA type as an input parameter.

Including and Excluding Bounds

Each non-empty range has two bounds, a lower bound and an upper bound. All values between the upper and lower bounds are included in the range. An inclusion bound means that the bound value itself is included in the range, while an exclusion bound means that the bound value is not included in the range.

In the textual form of a range, the inclusion lower bound is expressed as "[", and the exclusion lower bound is expressed as "("; the inclusion upper bound is expressed as "]", and the exclusion upper bound is expressed as ")". For details, see **Range Input/Output**.

The lower_inc and upper_inc functions detect the upper and lower bounds of a range value, respectively.

Infinite (Unbounded) Range

When the lower bound of a range is unbounded, it means that all values less than the upper bound are included in the range. Similarly, when the upper bound of a range is unbounded, all values greater than the lower bound are included in the range. When both the upper and lower bounds are unbounded, all values of the element type are considered within the range. When the upper bound and the lower bound of the range are not set, that is, the upper bound is positive infinity and the lower bound is negative infinity, the range is an infinite (unbounded) range.

Element types with the **infinity** values can be used as explicit bound values. For example, the range of timestamp [today,infinity) indicates that the special timestamp value **infinity** is not included, which is similar to [today,). The range of timestamp [today,infinity] indicates that the special timestamp value infinity is included, which is similar to [today,].

The lower_inf and upper_inf functions detect the infinite upper and lower bounds of a range, respectively.

Range Input/Output

The input of a range value must follow one of the following formats:

```
(lower-bound, upper-bound)
(lower-bound, upper-bound]
[lower-bound, upper-bound)
[lower-bound, upper-bound]
Empty
```

The output of a range value must follow one of the following formats:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
[lower-bound, upper-bound)
Empty
```

Parentheses or square brackets are used to indicate whether the upper and lower bounds are included or excluded. **empty** specifies an empty range (a range that does not contain vertices).

The value of *lower-bound* can be a valid input string of the subtype or null, indicating that there is no lower bound. Similarly, *upper-bound* can be a valid input string of the subtype or null, indicating that there is no upper bound.

The double quotation marks ("") are required for quotation if a bound value contains parentheses (), square brackets [], commas (,), double quotation marks (""), or backslashes (\). Otherwise, those characters will be considered as part of the range syntax. If the quoted bound value contains double quotation marks ("") or backslashes (\), add a backslash (\) before them for escape. (Two consecutive double quotation marks in the quoted bound value indicate one double quotation mark, which is the same as the single quotation mark escape logic in SQL strings.) In addition, all characters that may be parsed as range syntax must be escaped. An empty string boundary value must be written as **""**. If no any value is written, it indicates an infinite bound.

Spaces are allowed before and after a range value, but any space between parentheses or square brackets is used as part of the upper or lower bound value (whether the space has semantics depends on the element type).

Examples:

```
-- 3 is included, 7 is not included, and all values between 3 and 7 are included.
gaussdb=# SELECT '[3,7)'::int4range;
 int4range
-----------
 [3,7)
(1 row)
-- 3 and 7 are not included, but all values between 3 and 7 are included.
gaussdb=# SELECT '(3,7)'::int4range;
 int4range
-----------
 [4,7)
(1 row)
-- Only a single value 4 is included.
gaussdb=# SELECT '[4,4]'::int4range;
 int4range
-----------
 [4,5)
(1 row)
-- Any value is excluded (and will be normalized to empty).
gaussdb=# SELECT '[4,4)'::int4range;
 int4range
-----------
Empty
(1 row)
```

Constructing Range

Each range type has a constructor function with the same name. Using constructor functions is often more convenient than writing a range literal constant because it avoids extra references to bound values. Constructor functions receive two or three parameters.

- Two parameters: Construct a range in the standard format (the lower bound is included and the upper bound is excluded).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Three parameters: Construct a range according to the boundary type specified by the third parameter. The third parameter must be one of the following strings: (), (], [), or [].

Examples:

```
-- The complete format is: lower bound, upper bound, and textual parameters indicating the inclusion/exclusion of bounds.
gaussdb=# SELECT numrange(1.0, 14.0, '(]');
  numrange
------------
 (1.0,14.0]
(1 row)
-- If the third parameter is ignored, it is '[)' by default.
gaussdb=# SELECT numrange(1.0, 14.0);
  numrange
------------
 [1.0,14.0)
(1 row)
-- For discrete range types (such as int8range), although the third parameter is specified as '(]', the value will be converted to the standard format when being displayed.
gaussdb=# SELECT int8range(1, 14, '(]');
 int8range
-----------
 [2,15)
(1 row)
-- If NULL is used as the bound value, the corresponding bound is infinite.
gaussdb=# SELECT numrange(NULL, 2.2);
 numrange
----------
 (,2.2)
(1 row)
```

Discrete Range

A range element type (such as integer or date) has a clear definition of "step". In these types, if there is no valid value between two elements, they can be called adjacent elements. This is different from a continuous range, in which other element values can be identified between two given values. For example, the range of the numeric type is continuous, and the range of the timestamp type is also continuous. (Although timestamp has limited precision and can be considered discrete in theory, it can also be considered continuous because its step is not concerned.)

Another feature of the discrete range type is that each element value in the range has a next value and a previous value. Based on this feature, you can switch between inclusion and exclusion attributes of a range bound by replacing it with the original given next or previous element value. For example, in an integer range type, [4,8] and (3,9) represent the same set of values, but not for numeric ranges.

A discrete range type should have a regularization function that knows the specified step size of the element type. The regularization function is used to convert different range expressions with the same value set to the same format, in particular consistent with the inclusion or exclusion bounds. If you do not specify a regularization function, ranges with different formats will always be considered as unequal, even if they actually express the same set of values.

The built-in range types int4range, int8range, and daterange use the standard format that includes the lower bound and excludes the upper bound, that is, [). However, user-defined range types can use other conventions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Index

In addition, B-tree indexes can be created on table columns of the range type. For such indexes, only the equality operation is meaningful. Although the B-tree sorting order is defined for range values when the < and > operators are used, the sorting order is random and has limited value in actual applications. The B-tree support for range types is primarily designed to allow sorting within a query, rather than creating an index.

1.3.15 OID Types

GaussDB uses object identifiers (OIDs) as primary keys in system catalogs. You are advised not to use the OID column as the primary key in user-created tables. The OID type is implemented using a 4-byte unsigned integer. The system does not automatically add the OID column to user-created tables. **Table 1-23** describes the OID types.

Table 1-23 OID types

Name	Reference	Description	Examples
OID	-	OID represented in digits.	564182
CID	-	32-bit command identifier. It is the data type of the system columns **cmin** and **cmax**.	-
XID	-	64-bit transaction identifier. It is the data type of the system columns **xmin** and **xmax**.	-
TID	-	Row identifier. It is the data type of the system column **ctid**. A row ID is a pair (block number, tuple index within block) that identifies the physical location of the row within its table.	-
REGCONFIG	pg_ts_config	Text search configuration.	english
REGDICTIONARY	pg_ts_dict	Text search dictionary.	simple
REGOPER	pg_operator	Operator name.	-
REGOPERATOR	pg_operator	Operator with argument types	*(integer,integer) or -(NONE,integer)
REGPROC	pg_proc	Function name.	sum

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Reference	Description	Examples
REGPROCEDURE	pg_proc	Function with parameter types.	sum(int4)
REGCLASS	pg_class	Relation name.	pg_type
REGTYPE	pg_type	Data type name.	integer

The **OID** type is used for a column in the database system catalog.

Example:

```
gaussdb=# SELECT oid FROM pg_class WHERE relname = 'pg_type';
 oid
------
 1247
(1 row)
```

The alias type for **OID** is **REGCLASS** which allows simplified search for **OID** values.

Example:

```
gaussdb=# SELECT attrelid,attname,atttypid,attstattarget FROM pg_attribute WHERE attrelid =
'pg_type'::REGCLASS;
 attrelid |  attname   | atttypid | attstattarget
----------+------------+----------+---------------
     1247 | xc_node_id     |       23 |             0
     1247 | tableoid       |       26 |             0
     1247 | cmax           |       29 |             0
     1247 | xmax           |       28 |             0
     1247 | cmin           |       29 |             0
     1247 | xmin           |       28 |             0
     1247 | oid            |       26 |             0
     1247 | ctid           |       27 |             0
     1247 | typname        |       19 |            -1
     1247 | typnamespace   |       26 |            -1
     1247 | typowner       |       26 |            -1
     1247 | typlen         |       21 |            -1
     1247 | typbyval       |       16 |            -1
     1247 | typtype        |       18 |            -1
     1247 | typcategory    |       18 |            -1
     1247 | typispreferred |       16 |            -1
     1247 | typisdefined   |       16 |            -1
     1247 | typdelim       |       18 |            -1
     1247 | typrelid       |       26 |            -1
     1247 | typelem        |       26 |            -1
     1247 | typarray       |       26 |            -1
     1247 | typinput       |       24 |            -1
     1247 | typoutput      |       24 |            -1
     1247 | typreceive     |       24 |            -1
     1247 | typsend        |       24 |            -1
     1247 | typmodin       |       24 |            -1
     1247 | typmodout      |       24 |            -1
     1247 | typanalyze     |       24 |            -1
     1247 | typalign       |       18 |            -1
     1247 | typstorage     |       18 |            -1
     1247 | typnotnull     |       16 |            -1
     1247 | typbasetype    |       26 |            -1
     1247 | typtypmod      |       23 |            -1
     1247 | typndims       |       23 |            -1
     1247 | typcollation   |       26 |            -1
     1247 | typdefaultbin  |      194 |            -1
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
   1247 | typdefault    |       25 |           -1
   1247 | typacl        |     1034 |           -1
   1247 | typelemmod    |       23 |           -1
(39 rows)
```

1.3.16 Pseudo-Types

The data types of GaussDB contain a number of special-purpose types that are collectively called pseudo-types, as listed in **Table 1-24**. A pseudo-type cannot be used as a column data type, but it can be used to declare a function's argument or result type.

Pseudo-types are useful when a function not only simply accepts or returns a certain SQL data type. **Table 1-24** lists all pseudo-types.

Table 1-24 Pseudo-types

Name	Description
any	Indicates that a function accepts any input data type.
anyelement	Indicates that a function accepts any data type.
anyarray	Indicates that a function accepts any array data type.
anynonarray	Indicates that a function accepts any non-array data type.
anyenum	Indicates that a function accepts any enum data type.
anyrange	Indicates that a function accepts any range data type.
cstring	Indicates that a function accepts or returns a null-terminated C string.
internal	Indicates that a function accepts or returns a server-internal data type.
language_handler	Indicates that a procedural language call handler is declared to return **language_handler**.
fdw_handler	Indicates that a foreign-data wrapper handler is declared to return **fdw_handler**.
record	Identifies a function returning an unspecified row type.
trigger	Indicates that a trigger function is declared to return **trigger**.
void	Indicates that a function returns no value.
opaque	Indicates an obsolete type name that formerly served all the above purposes.

Functions coded in C language (whether built in or dynamically loaded) can declare any pseudo-type data as an input or output argument. When a pseudo-type is used as an argument type, the user needs to ensure that the function can run properly.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Functions coded in procedural languages can use pseudo-types only as allowed by their implementation languages. At present the procedural languages all forbid use of a pseudo-type as an argument type, and allow only void or record as a return type. Some also support polymorphic functions using the anyelement, anyarray, anynonarray, anyenum, and anyrange types.

Each location (parameter or return value) declared as the anyelement type is allowed to have any specific actual data type, but they must all be of the same actual type in any given query.

The internal pseudo-type is used to declare functions that are meant only to be called internally by the database system, and not by direct calling in an SQL query. If a function has at least one internal-type argument, then it cannot be called from SQL. You are advised not to create any function that is declared to return the internal type unless it has at least one internal argument.

Example:

```
-- Create a table.
gaussdb=# CREATE TABLE t1 (a int);

-- Insert two data records.
gaussdb=# INSERT INTO t1 values(1),(2);

-- Create the showall() function.
gaussdb=# CREATE OR REPLACE FUNCTION showall() RETURNS SETOF record
AS $$ SELECT count(*) from t1; $$
LANGUAGE SQL;

-- Call the showall() function.
gaussdb=# SELECT showall();
 showall
---------
 (2)
(1 row)

-- Drop the function.
gaussdb=# DROP FUNCTION showall();

-- Drop the table.
gaussdb=# DROP TABLE t1;
```

1.3.17 Data Types Used by the Ledger Database

The ledger database uses the hash16 data type to store row-level hash digests or table-level hash digests, and uses the hash32 data type to store global hash digests or history table verification hash values.

Table 1-25 Hash types of the ledger database

Name	Description	Storage Space	Range
HASH16	Stored as an unsigned 64-bit integer.	8 bytes	0 to +18446744073709551615

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Storage Space	Range
HASH32	Stored as an unsigned integer array of 16 elements.	16 bytes	Value range of an unsigned integer array of 16 elements.

The hash16 data type is used to store row-level or table-level hash digests in the ledger database. After obtaining the hash sequence of a 16-character hexadecimal string, the system calls the hash16in function to convert the sequence into an unsigned 64-bit integer and stores the integer in a hash16 variable. For example:

```
Hexadecimal string: e697da2eaa3a775b; 64-bit unsigned integer: 16615989244166043483
Hexadecimal string: ffffffffffffffff; 64-bit unsigned integer: 18446744073709551615
```

Example:

```
gaussdb=# CREATE TABLE hash16test(id int, data hash16);
gaussdb=# INSERT INTO hash16test VALUES (1, 'e697da2eaa3a775b');
gaussdb=# SELECT * FROM hash16test;
 id |              data
----+----------------------------------
  1 | e697da2eaa3a775b
(1 row)
gaussdb=# DROP TABLE hash16test;
```

The hash32 data type is used to store the global hash digest or history table verification hash in the ledger database. After obtaining the hash sequence of a 32-character hexadecimal string, the system calls the hash32in function to convert the sequence to an array containing 16 unsigned integer elements. For example:

```
Hexadecimal string: 685847ed1fe38e18f6b0e2b18c00edee
Hash32 array: [104,88,71,237,31,227,142,24,246,176,226,177,140,0,237,238]
```

Example:

```
gaussdb=# CREATE TABLE hash32test(id int, data hash32);
gaussdb=# INSERT INTO hash32test VALUES (1, '685847ed1fe38e18f6b0e2b18c00edee');
gaussdb=# SELECT * FROM hash32test;
 id |              data
----+----------------------------------
  1 | 685847ed1fe38e18f6b0e2b18c00edee
(1 row)
gaussdb=# DROP TABLE hash32test;
```

1.3.18 XML Types

The XML data type can store well-formed documents as defined by the XML standard, as well as content fragments, which are defined by referencing broader "DOCUMENT NODE" XQuery and XPath data models, allowing content fragments to contain multiple top-level elements or character nodes. The expression XMLVALUE IS DOCUMENT can be used to determine whether a specific XML value is a complete document or a document fragment. The internal format of XML is the same as that of the TEXT data type. Its advantage over storing XML data in a TEXT field is that, XML data supports standard XML operation functions based on LIBXML2 and XML standardization check. For details about functions related to XML types, see **Functions of the XML Type**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The XML parser converts an XML document into an XML DOM object. The document object model (DOM) defines standard methods for accessing and manipulating documents. XML DOM defines standard methods for accessing and manipulating XML documents. XML DOM considers XML documents as a tree structure. All elements can be accessed through the DOM tree, and their content can be modified or deleted. New elements can be created. An element, the text of an element, and the attributes of an element are considered as nodes.

The XML bottom layer uses the same data structure as the TEXT type for storage. The maximum size is 1 GB.

Example:

```
gaussdb=# CREATE TABLE xmltest ( id int, data xml );
gaussdb=# INSERT INTO xmltest VALUES (1, 'one');
gaussdb=# INSERT INTO xmltest VALUES (2, XMLCONCAT(('<?xml version="1.0" encoding="UTF8"
standalone="no"?><bar>foo</bar>'), ('<?xml version="1.0" encoding="UTF8" standalone="no"?><bar>foo</
bar>')));
gaussdb=# SELECT * FROM xmltest ORDER BY 1;
 id |                          data
----+-------------------------------------------------------------------
  1 | one
  2 | <?xml version="1.0" standalone="no"?><bar>foo</bar><bar>foo</bar>
(2 rows)
gaussdb=# DROP TABLE xmltest;
```

The XML type does not support the following operations:

- Logical expressions AND, OR, and NOT.
- Input parameter of a system function that is used as a non-XML operation function.
- Used as a distribution key, partition key, subpartition key, primary key, or unique constraint.
- Implicit conversion related to XML, including the conversion between strings and the XML data type.
- Array expression, row expression, subquery expression, TABLE OF, and TABLE OF INDEX.
- Used to create an ordinary index, unique index, global index, local index, or partial index using XML columns.
- Comparison expressions >, <, >=, <=, =, <>, !=, ^=, <=>, BETWEEN AND, IS DISTINCT FROM, and IS NOT DISTINCT FROM.
- Condition expressions DECODE, NULLIF, GREATEST, and LEAST.
- Parameters of the DISTINCT, GROUP BY, and ORDER BY expressions.
- Aggregate functions, including sum, max, min, avg, list_agg, corr, covar_pop, covar_samp, stddev, stddev_pop, stddev_samp, var_pop, var_samp, variance, bit_and, bit_or, bool_and, bool_or, every, regr_avgx, regr_avgy, regr_count, regr_intercept, regr_r2, regr_slope, regr_sxx, regr_sxy, regr_syy, rank, and spread.
- ODBC-related APIs with binding parameters.

The XML type supports the following operations:

- Physical backup and restoration.
- NULL value determinations IS NULL and IS NOT NULL.
- Condition expressions CASE and COALESCE.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Global temporary tables and local temporary tables.
- Forcible type conversion.
- Expression indexes.
- Input XML values that comply with the XML standard.
- gs_dump and gs_restore.
- Parallel query, supporting Astore and Ustore.
- Input parameters, output parameters, customized variables, and return values of a user-defined function.
- Input parameters, output parameters, customized variables, and return values of a stored procedure. Stored procedures in autonomous transactions are supported.
- Character processing functions quote_literal(string text) (explicitly converted to the character type) and quote_nullable(string text) (explicitly converted to the character type).
- Aggregate functions count, array_agg, and checksum (explicitly converted to the character type), and string_agg (explicitly converted to the character type).
- If addition, deletion, modification, and query of user-defined composite types involve XML types that are similar to XML columns in ordinary tables, the composite types must be inserted and modified based on the XML syntax.
- JDBC and ODBC operations on XML data types are supported. The SELECT, UPDATE, INSERT, and DELETE operations can be performed on an XML column. You can enter an XML value using the SQL syntax and use the getSQLXML method of the ResultSet class to obtain the XML value. JDBC-related APIs with binding parameters are supported. For example, you can use the setSQLXML method in the PreparedStatement API and the getSQLXML(int columnIndex) method in the ResultSet execution result set API.

  In the calling process, use the java.sql.SQLXML API class to construct an XML object, set the specified object type to Oid.XML, and send the type ID and XML value to the server. After obtaining the result returned from the server, call ResultSet.getString. Then, use the java.sql.SQLXML API class to construct the obtained character string to an XML object. In this case, the system checks whether the content complies with the XML standard again. Therefore, you can also use ResultSet.getString to directly obtain the string object.

1.3.19 XMLType

The XMLType type stores well-formed "documents" that comply with the XML standard. Data is stored in character strings. Its advantage over storing XML data in a TEXT field is that, XMLType data supports standard XML operation functions based on LIBXML2 and XML standardization check. For details about functions related to XMLType, see **Functions of the XMLType Type**.

The XML parser converts an XML document into an XML DOM object. The document object model (DOM) defines standard methods for accessing and manipulating documents. XML DOM defines standard methods for accessing and manipulating XML documents. XML DOM considers XML documents as a tree structure. All elements can be accessed through the DOM tree, and their content can be modified or deleted. New elements can be created. An element, the text of

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

an element, and the attributes of an element are considered as nodes. The maximum size is 1 GB.

Example:

```
gaussdb=# CREATE TABLE xmltypetest(id int, data xmltype);
gaussdb=# INSERT INTO xmltypetest VALUES (1, '<ss/>');
gaussdb=# INSERT INTO xmltypetest VALUES (2, '<xx/>');
gaussdb=# SELECT * FROM xmltypetest ORDER BY 1;
 id | data
----+-------
  1 | <ss/>
  2 | <xx/>
(2 rows)
gaussdb=# DROP TABLE xmltypetest;
```

The XMLType type does not support the following operations:

- Logical expressions AND, OR, and NOT.
- Input parameter of a system function that is used as a non-XMLType operation function.
- Used as a distribution key, partition key, subpartition key, primary key, or unique constraint.
- Implicit conversion related to XMLType, including the conversion between strings and the XMLType data type.
- Array expression, row expression, subquery expression, TABLE OF, and TABLE OF INDEX.
- Used to create an ordinary index, unique index, global index, local index, or partial index using XMLType columns.
- Comparison expressions >, <, >=, <=, =, <>, !=, ^=, <=>, BETWEEN AND, IS DISTINCT FROM, and IS NOT DISTINCT FROM.
- Condition expressions DECODE, NULLIF, GREATEST, and LEAST.
- Parameters of the DISTINCT, GROUP BY, and ORDER BY expressions.
- Aggregate functions, including sum, max, min, avg, list_agg, corr, covar_pop, covar_samp, stddev, stddev_pop, stddev_samp, var_pop, var_samp, variance, bit_and, bit_or, bool_and, bool_or, every, regr_avgx, regr_avgy, regr_count, regr_intercept, regr_r2, regr_slope, regr_sxx, regr_sxy, regr_syy, rank, and spread.
- ODBC-related APIs with binding parameters.

The XMLType type supports the following operations:

- Physical backup and restoration.
- NULL value determinations IS NULL and IS NOT NULL.
- Condition expressions CASE and COALESCE.
- Global temporary tables and local temporary tables.
- Forcible type conversion.
- Expression indexes.
- Input XMLType values that comply with the XML standard.
- gs_dump and gs_restore.
- Parallel query, supporting Astore and Ustore.
- Input parameters, output parameters, customized variables, and return values of a user-defined function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Input parameters, output parameters, customized variables, and return values of a stored procedure. Stored procedures in autonomous transactions are supported.
- Character processing functions quote_literal(string text) (explicitly converted to the character type) and quote_nullable(string text) (explicitly converted to the character type).
- Aggregate functions count, array_agg, and checksum (explicitly converted to the character type), and string_agg (explicitly converted to the character type).
- If addition, deletion, modification, and query of user-defined composite types involve XMLType types that are similar to XMLType columns in ordinary tables, the composite types must be inserted and modified based on the XMLType syntax.
- You can perform the SELECT, UPDATE, INSERT, or DELETE operation on the column and use the SQL syntax to enter the XMLType value.

You can create a schema named **xmltype**. In the schema, you can create functions, but cannot use schema.func() to call functions defined in the schema.

When the XMLType type is output, the encoding attribute in the XML text declaration is changed to the current database encoding.

1.3.20 ACLItem Type

The ACLItem data type is used to store object permission information. The internal implementation is based on the int type, and the supported format is "*user1*=*privs*/*user2*".

The aclitem[] data type is an array consisting of multiple ACLItem elements. The supported format is "**{***user1*=*privs1*/*user3*, *user2*=*privs2*/*user3***}**".

user1, *user2*, and *user3* indicate the existing users or roles in the database, and *privs* indicates the permissions supported by the database. For details, see **Table 7-105**.

Example:

```
-- Create a user.
gaussdb=# CREATE USER user1 WITH PASSWORD '********';
gaussdb=# CREATE USER user2 WITH PASSWORD '********';
gaussdb=# CREATE USER omm WITH PASSWORD '********';

-- Create a data table table_acl that contains three columns of the int, aclitem, and aclitem[] types.
gaussdb=# CREATE TABLE table_acl (id int,priv aclitem,privs aclitem[]);

-- Insert a data record whose content is (1,'user1=arw/omm','{omm=d/user2,omm=w/omm}') into the
table_acl table.
gaussdb=# INSERT INTO table_acl VALUES (1,'user1=arw/omm','{omm=d/user2,omm=w/omm}');

-- Insert a data record whose content is (2,'user1=aw/omm','{omm=d/user2}') into the table_acl table.
gaussdb=# INSERT INTO table_acl VALUES (2,'user1=aw/omm','{omm=d/user2}');
gaussdb=# SELECT * FROM  table_acl;
 id |     priv      |          privs
----+---------------+-------------------------
  1 | user1=arw/omm | {omm=d/user2,omm=w/omm}
  2 | user1=aw/omm  | {omm=d/user2}
(2 rows)

-- Drop the table and users.
gaussdb=# DROP USER user1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# DROP USER user2;
gaussdb=# DROP USER omm;
gaussdb=# DROP TABLE table_acl;
```

1.3.21 Array Types

Array types can be used to store several elements of the same type. For details about how to use array types and related functions, see **Arrays**.

Array Type Definition

Generally, an array type is defined by adding square brackets ([]) to the end of the data type name of an array element.

Example 1: Create a table named **sal_emp**. The table has the following columns: **name** of the text type, indicating employee names; **pay_by_quarter** of the integer element type, indicating an array of quarterly salaries; **phone_numbers** of the varchar(11) element type, indicating an array of mobile numbers.

```
gaussdb=# CREATE TABLE sal_emp (
  name            text,
  pay_by_quarter  integer[],
  phone_numbers   varchar(11)[]
);
gaussdb=# DROP TABLE sal_emp;
```

Example 2: Define an array type in other ways. For details about the definition method and behavior, see the comments in the example.

```
gaussdb=# CREATE TABLE sal_emp (
  name             text,
  pay_by_quarter1  integer[][],      -- Two-dimensional array of the int type.
  pay_by_quarter2  integer[3],       -- One-dimensional array of the int type. The size is 3.
  pay_by_quarter3  integer[3][3],    -- Two-dimensional array of the int type. The size of each dimension is 3.
  pay_by_quarter4  integer ARRAY,    -- One-dimensional array of the int type.
  pay_by_quarter5  integer ARRAY[3]  -- One-dimensional array of the int type. The size is 3.
);
gaussdb=# DROP TABLE sal_emp;
```

NOTE

- The number of dimensions can be specified in the definition of an array type. However, this definition is only a part of the type declaration and does not restrict the number of dimensions of the actually stored array.
- The size can be specified in the definition of an array type. However, this definition is only a part of the type declaration and does not restrict the size of the actually stored array.
- An array can have up to six dimensions.
- The restrictions on the number of array elements are as follows:
  - The maximum number of elements is 134217727.
  - The maximum storage space of all elements cannot exceed 1 GB minus 1 byte, that is, 1073741823 bytes.

Array Constructor

An array constructor is an expression used to build array values. A simple array constructor consists of the keyword ARRAY and an expression list of array element values which are separated by commas (,) and enclosed by square brackets ([]). For example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT ARRAY[1, 2, 3 + 4];
  array
---------
 {1,2,7}
(1 row)
```

By default, the element type of an array is the common type of member expressions and follows the same type inference rules as the UNION or CASE structure (for details, see **UNION, CASE, and Related Constructs**). You can construct an array to the desired data type through explicit type conversion. The following is an example:

```
gaussdb=# SELECT ARRAY[1, 2, 3]::varchar[];
  array
---------
 {1,2,3}
(1 row)

gaussdb=# SELECT ARRAY['a', 'b', 'c']::varchar[];
  array
---------
 {a,b,c}
(1 row)
```

In addition to the preset basic types, the record type and table type can also be defined as array types. The following is an example:

```
gaussdb=# CREATE TYPE rec IS (c1 int, c2 int);
gaussdb=# SELECT ARRAY[(1, 1), (2, 2)]::rec[];
       array
-------------------
 {"(1,1)","(2,2)"}
(1 row)

gaussdb=# CREATE TABLE tab (c1 int, c2 int);
gaussdb=# SELECT ARRAY[(1, 1), (2, 2)]::tab[];
       array
-------------------
 {"(1,1)","(2,2)"}
(1 row)

gaussdb=# DROP TYPE rec;
gaussdb=# DROP TABLE tab;
```

An array must have a specific type. Therefore, when constructing an empty array, you must specify the required type. The following is an example:

```
gaussdb=# SELECT ARRAY[]::int[];
 array
-------
 {}
(1 row)
```

You can also construct an array from the result of a subquery. In this case, the array constructor consists of the keyword ARRAY and a subquery enclosed in parentheses. The subquery must return only one separate column. The generated one-dimensional array generates an element for each row of the result in the subquery. The element type matches the output column of the subquery. For example:

```
gaussdb=# SELECT ARRAY(SELECT generate_series(1, 6));
     array
---------------
 {1,2,3,4,5,6}
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Multidimensional array values can be created by nesting array constructors. The ARRAY keyword in the inner constructor can be omitted. For example, the following two examples will produce the same result:

```
gaussdb=# SELECT ARRAY[ARRAY[1,2], ARRAY[3,4]];
     array
---------------
 {{1,2},{3,4}}
(1 row)

gaussdb=# SELECT ARRAY[[1,2], [3,4]];
     array
---------------
 {{1,2},{3,4}}
(1 row)
```

NOTE

- Inner constructors at the same layer must generate subarrays of the same dimension.
- Any type conversion applied to the outer ARRAY constructor is automatically applied to all inner constructors.

String Inputs of the Array Type

To write an array value as a literal constant (constant input), enclose the element values with braces and separate them with commas. The general format of an array constant is as follows:

```
'{ val1 delim val2 delim ... }'
```

In the preceding format, **delim** indicates the delimiter of the element type and is recorded in the **typdelim** column in the pg_type catalog. Each **val** can be a constant of the array element type or a subarray. For example:

```
gaussdb=# SELECT '{1, 2, 3}'::int[] AS RESULT;
 result
---------
 {1,2,3}
(1 row)

gaussdb=# SELECT '{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}'::int[] AS RESULT;
         result
-----------------------------
 {{1,2,3},{4,5,6},{7,8,9}}
(1 row)
```

Double quotation marks can be used around any element value, and double quotation marks must be used when the element value contains special characters such as commas or braces.

The following example indicates that there is an array of the varchar type and there are seven varchar elements in total. The elements are as follows:

- Character string containing a space.
- Character string whose value is **"NULL"**.
- Character string whose value is **NULL**.
- Character string containing a backslash (\).
- Character string containing a left brace ({).
- Character string containing a right brace (}).
- Character string containing a comma (,).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT '{" ", "NULL", null, "\\", "{", "}", ","}'::varchar[] AS RESULT;
              result
-----------------------------------
 {" ","NULL",NULL,"\\","{","}",","}
(1 row)
```

NOTE

- For array string constant inputs, if the array element value is an empty string, contains braces, delimiters, double quotation marks, backslashes, or spaces, or matches the keyword NULL, these elements are enclosed in double quotation marks. An additional backslash is input if the element value contains double quotation marks or backslashes.
- The keyword NULL is case-insensitive.
- The input spaces not enclosed in double quotation marks are automatically ignored.
- You are advised to use the array constructor instead of string constants to create arrays.

String Outputs of the Array Type

The output of an array value consists of the output of the array element type plus some modifiers indicating the array structure. These modifiers consist of braces ({}) around array values plus delimiters between adjacent items. In a multidimensional array, each dimension has its own level of braces and contains delimiters between adjacent braces at the same level.

The data of the array type contains special characters (in the following description). The following is an example of character string output:

```
gaussdb=# SELECT ARRAY['{', '}', 'hello, world', '"', '\', '  ', NULL] AS RESULT;
                  array
-----------------------------------------------
 {"{","}","hello, world","\"","\\","  ",NULL}
(1 row)
```

NOTE

For array string constant outputs, if the array element value is an empty string or contains braces, delimiters, double quotation marks, backslashes, or spaces, or the element is **NULL**, these element outputs are automatically enclosed in double quotation marks. An additional backslash is output if the element value contains double quotation marks or backslashes.

Use of Array Types

The following is an example of using the array type:

```
-- Create a table with array columns and insert some data.
gaussdb=# CREATE TABLE orders (
  name  varchar,
  items varchar[]
);
gaussdb=# INSERT INTO orders VALUES('a', ARRAY['Apple', 'Orange', 'Pear']);
gaussdb=# INSERT INTO orders VALUES('b', ARRAY['Mineral water', 'Coke', 'Sprite']);
gaussdb=# INSERT INTO orders VALUES('c', ARRAY['Mouse', 'Keyboard', 'Earphone']);
gaussdb=# INSERT INTO orders VALUES('d', '{Cabbage, Potato, Eggplant}');

-- Query data.
gaussdb=# SELECT * FROM orders ORDER BY name;
 name |       items
------+--------------------
 a    | {Apple,Orange,Pear}
 b    | {Mineral water,Coke,Sprite}
 c    | {Mouse,Keyboard,Earphone}
 d    | {Cabbage,Potato,Eggplant}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(4 rows)

-- Access array elements.
gaussdb=# SELECT items[1] FROM orders ORDER BY name;
 items
--------
 Apple
 Mineral water
 Mouse
 Cabbage
(4 rows)

-- If the accessed element exceeds the range or the accessed index is NULL, NULL is returned.
gaussdb=# SELECT items[4] FROM orders ORDER BY name;
 items
-------




(4 rows)

gaussdb=# SELECT items[null] FROM orders ORDER BY name;
 items
-------




(4 rows)

-- Access a subarray.
gaussdb=# SELECT items[1:2] FROM orders ORDER BY name;
     items
---------------
 {Apple,Orange}
 {Mineral water,Coke}
 {Mouse,Keyboard}
 {Cabbage,Potato}
(4 rows)

-- Update the entire array.
gaussdb=# UPDATE orders SET items = ARRAY['Banana', 'Watermelon', 'Strawberry'] WHERE name = 'a';
gaussdb=# SELECT items FROM orders WHERE name = 'a';
      items
------------------
 {Banana,Watermelon,Strawberry}
(1 row)

-- Update the elements of an array.
gaussdb=# UPDATE orders SET items[1] = 'Mango' WHERE name = 'a';
gaussdb=# SELECT items FROM orders WHERE name = 'a';
      items
------------------
 {Mango,Watermelon,Strawberry}
(1 row)

-- Update the element fragment of an array.
gaussdb=# UPDATE orders SET items[1:2] = ARRAY['Computer','Phone'] WHERE name = 'c';
gaussdb=# SELECT items FROM orders WHERE name = 'c';
      items
------------------
 {Computer,Mobile phone,Earphone}
(1 row)

-- Add an array element. All unassigned elements between the last element of the original array and the
new element are set to NULL.
gaussdb=# UPDATE orders SET items[4] = 'Display' WHERE name = 'c';
gaussdb=# SELECT items FROM orders WHERE name = 'c';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
          items
-------------------------
 {Computer,Mobile phone,Earphone,Display}
(1 row)

gaussdb=# UPDATE orders SET items[6] = 'Display 2' WHERE name = 'c';
gaussdb=# SELECT items FROM orders WHERE name = 'c';
                items
--------------------------------------
 {Computer,Mobile phone,Earphone,Display,NULL,Display 2}
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE orders;
```

1.3.22 Data Types Supported by the Vectorized Executor

Table 1-26 lists the data types supported by the vectorized executor.

Table 1-26 Data types supported by the vectorized executor

Category	Data Type	Length
Numeric Types	tinyint [unsigned]	1
	smallint [unsigned]	2
	mediumint [unsigned]	3
	integer [unsigned]	4
	bigint [unsigned]	8
	int16 [signed]	16
	decimal	-1
	numeric	-1
	real	4
	double precision	8
	smallserial	2
	serial	4
	bigserial	8
	largeserial	-1
Monetary Types	money	8
Boolean Types	Boolean	1
Character Types	character varying(n), varchar(n)	-1
	character(n), char(n),nchar(n)	n
	character, char	1

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Data Type	Length
	text	-1
	varchar2(n)	-1
	nvarchar2(n)	-1
	clob	-1
Binary Types	blob	-1
	raw	-1
	bytea	-1
	byteawithoutorderwithe-qualcol	-1
	byteawithoutordercol	-1
Bit String Types	bit(n)	n
	bit varying(n)	-1
Date/Time Types	timestamp with time zone	8
	timestamp without time zone	8
	date	4
	time without time zone	8
	time with time zone	12
	smalldatetime	8
	interval	16
	reltime	4
	abstime	4
	datea	8
Network Address Types	cidr	7 or 19
	inet	7 or 19
Text Retrieval Types	tsvector	-1
	tsquery	-1
JSON Types	json	-1
	jsonb	-1
HyperLoglog Types	hll	-1

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Data Type	Length
Range Types	int4range	-1
	int8range	-1
	numrange	-1
	tsrange	-1
	tstzrange	-1
	daterange	-1
Object Identifier Types	OID	4
	CID	4
	XID	8
	TID	8
	regconfig	4
	regdictionary	4
	regoper	4
	regoperator	4
	regproc	4
	regprocedure	4
	regclass	4
	regtype	4
Pseudo Types	anyelement	4
	anynonarray	4
	anyrange	-1
	cstring	-2
	record	-1
	void	4
XML Types	xml	-1
XMLTYPE Types	xmltype	-1
Hash Types	hash16	8
Array Types	tinyint[]	-1
	smallint[]	-1
	mediumint[]	-1

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Data Type	Length
	integer[]	-1
	bigint[]	-1
	int16[]	-1
	decimal[]	-1
	numeric[]	-1
	real[]	-1
	double precision[]	-1
	money[]	-1
	boolean[]	-1
	varchar[]	-1
	char(n)[]	-1
	char[]	-1
	text[]	-1
	varchar2(n)[]	-1
	nvarchar2(n)[]	-1
	clob[]	-1
	blob[]	-1
	raw[]	-1
	bytea[]	-1
	byteawithoutorderwithe-qualcol[]	-1
	byteawithoutordercol[]	-1
	bit(n)[]	-1
	bit varying(n)[]	-1
	timestamp with time zone[]	-1
	timestamp without time zone[]	-1
	date[]	-1
	time without time zone[]	-1
	time with time zone[]	-1

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Data Type	Length
	smalldatetime[]	-1
	interval[]	-1
	reltime[]	-1
	abstime[]	-1
	datea[]	-1
	cidr[]	-1
	inet[]	-1
	tsvector[]	-1
	tsquery[]	-1
	json[]	-1
	jsonb[]	-1
	hll[]	-1
	int4range[]	-1
	int8range[]	-1
	numrange[]	-1
	tsrange[]	-1
	tstzrange[]	-1
	daterange[]	-1
	OID[]	-1
	CID[]	-1
	XID[]	-1
	TID[]	-1
	regconfig[]	-1
	regdictionary[]	-1
	regoper[]	-1
	regoperator[]	-1
	regproc[]	-1
	regprocedure[]	-1
	regclass[]	-1
	regtype[]	-1

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Data Type	Length
	cstring[]	-1
	record[]	-1
	xml[]	-1
	xmltype[]	-1
	hash16[]	-1
	hash32[]	-1
	aclitem[]	-1
	name[]	-1
	macaddr[]	-1
	UUID[]	-1
	point[]	-1
	lseg[]	-1
	box[]	-1
	path[]	-1
	polygon[]	-1
	circle[]	-1
Vector Types	floatvector	-1
	boolvector	-1
Other Types	enum	An enum type is a list of several labels. Each label value is a non-empty string, and the string length cannot exceed 63 bytes.

NOTE

Currently, the enum type can be created only in the following mode. For details, see **CREATE TYPE**.

```
CREATE TYPE name AS ENUM (['label' [, ...]]);
```

1.3.23 ROWID Type

The ROWID data type is used to identify a data row in a table and contains key information required for locating the row. In this way, data rows in the table can be quickly located.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The ROWID data type consists of 29 Base64-encoded characters. **Table 1-27** describes the meaning of each character string.

Table 1-27 Columns of the ROWID type

Column	Description	Character Length	Value Range (Base64 Encoding)
reserve_flag	Reserved flag.	6	**0** (**AAAAAA**) in the current version.
node_slice	Shard ID of a node.	3	The value ranges from 0 (AAA) to 65535 (P//). The actual maximum value is the number of cluster shards.
bucket_id	Bucket ID of the hash bucket table.	3	The value ranges from 0 (AAA) to 32767 (H//). The actual maximum value is the maximum bucket ID of the table. For a non-hash bucket table, the value of this column is fixed at **65535** (**P//**).
table_oid	Table OID.	6	The value ranges from 0 (AAAAAA) to 4294967295 (D/////).
row_no	Row ID.	11	The value ranges from 0 (AAAAAAAAAAA) to 9223372036854775807 (H//////////).

NOTE

The ROWID type does not support the following operations:

- The ROWID type cannot be used as the partitioned key or distribution key of a table.
- The ROWID type does not support HTAP.
- The ROWID type is supported only in Oracle-compatible databases.
- The ROWID type does not support data masking policies.
- The ROWID type cannot be used as the subtype of a user-defined range type. For details, see **Range Types**.
- The ROWID type supports only the JDBC driver.

Examples

```
gaussdb=# CREATE DATABASE ora_test DBCOMPATIBILITY='ORA';
CREATE DATABASE
gaussdb=# \c ora_test
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "ora_test" as user "$user".

-- Create a table.
ora_test=# CREATE TABLE rowid_test(id INT, rwd ROWID) DISTRIBUTE BY HASH(id);
CREATE TABLE

-- Insert data whose value is NULL.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ora_test=# INSERT INTO rowid_test VALUES(1, NULL);
INSERT 0 1

-- Insert data that meets the ROWID format.
ora_test=# INSERT INTO rowid_test VALUES(2, 'AAAAAAAAAP//BBBBBAAAAAAAAAAAC');
INSERT 0 1
ora_test=# INSERT INTO rowid_test VALUES(3, 'AAAAAAAAAP//BBBd+AAAAAAAA901/');
INSERT 0 1
ora_test=# INSERT INTO rowid_test VALUES(4, 'AAAAAAAAAP//A12i+AAAAAAAAItn/');
INSERT 0 1
ora_test=# INSERT INTO rowid_test VALUES(5, 'AAAAAAAAAP//CC+uJAAAAAAGASIi9');
INSERT 0 1

-- Insert data that does not meet the ROWID format.
ora_test=# INSERT INTO rowid_test VALUES(6, 'AAAAAAAAAP//BBBd+AAAAAAAA@@@@');
ERROR:  Invalid rowid.
LINE 1: INSERT INTO rowid_test VALUES(6, 'AAAAAAAAAP//BBBd+AAAAAAAA@...
                                         ^
CONTEXT:  referenced column: rwd

-- Insert data that does not meet the ROWID value range.
ora_test=# INSERT INTO rowid_test VALUES(6, 'AAAAAAAAAP/////d+AAAAAAAA901/');
ERROR:  Invalid rowid.
LINE 1: INSERT INTO rowid_test VALUES(6, 'AAAAAAAAAP/////d+AAAAAAAA9...
                                         ^
CONTEXT:  referenced column: rwd

-- View data.
ora_test=# SELECT * FROM rowid_test;
 id |              rwd
----+-------------------------------
  1 |
  2 | AAAAAAAAAP//BBBBBAAAAAAAAAAAC
  3 | AAAAAAAAAP//BBBd+AAAAAAAA901/
  4 | AAAAAAAAAP//A12i+AAAAAAAAItn/
  5 | AAAAAAAAAP//CC+uJAAAAAAGASIi9
(5 rows)

-- Drop the table.
ora_test=# DROP TABLE IF EXISTS rowid_test;
DROP TABLE
ora_test=# \c postgres
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "postgres" as user "$user".
gaussdb=# DROP DATABASE ora_test;
DROP DATABASE
```

Operators

The ROWID type supports comparison operators (=, <>, <, >, <=, and >=). If a comparison operation is performed between the ROWID type and a data type that cannot be implicitly converted to the ROWID type, the error message "Inconsistent datatypes" is displayed. For details about comparison operators, see **Comparison Operators**.

The comparison rules of the ROWID type are as follows: The values of the **node_slice**, **table_oid**, **bucket_id**, and **row_no** columns are compared in sequence.

Examples

```
gaussdb=# CREATE DATABASE ora_test DBCOMPATIBILITY='ORA';
CREATE DATABASE
gaussdb=# \c ora_test
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "ora_test" as user "$user".

-- Compare data of the ROWID type.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ora_test=# CREATE OR REPLACE FUNCTION func1 RETURN ROWID AS
ora_test$# DECLARE
ora_test$#     a ROWID;
ora_test$#     b ROWID;
ora_test$# BEGIN
ora_test$#     a := 'AAAAAAAAAP//ABPH//AAAAAAAACAB';
ora_test$#     b := 'AAAAAAAAAP//ABPH//AAAAAAAACAA';
ora_test$#     DBE_OUTPUT.PRINT_LINE((a = b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <> b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a > b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a < b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <= b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a >= b)::text);
ora_test$#     RETURN a;
ora_test$# END;
ora_test$# /
CREATE FUNCTION
ora_test=# SELECT func1();
false
true
true
false
false
true
            func1
------------------------------
 AAAAAAAAAP//ABPH//AAAAAAAACAB
(1 row)

-- Compare data of the ROWID type with data of other types that can be implicitly converted.
ora_test=# CREATE OR REPLACE FUNCTION func1 RETURN ROWID AS
ora_test$# DECLARE
ora_test$#     a ROWID;
ora_test$#     b TEXT;
ora_test$#     c VARCHAR(30);
ora_test$#     d VARCHAR2;
ora_test$#     e CHAR(29);
ora_test$#     F NVARCHAR2;
ora_test$# BEGIN
ora_test$#     -- Prepare data.
ora_test$#     a:='AAAAAAAAAP//ABPH//AAAAAAAACAB';
ora_test$#     b:='AAAAAAAAAP//ABPH//AAAAAAAACAA';
ora_test$#     c:='AAAAAAAAAP//ABPH//AAAAAAAACAA';
ora_test$#     d:='AAAAAAAAAP//ABPH//AAAAAAAACAA';
ora_test$#     e:='AAAAAAAAAP//ABPH//AAAAAAAACAA';
ora_test$#     F:='AAAAAAAAAP//ABPH//AAAAAAAACAA';
ora_test$#
ora_test$#     -- Compare with text.
ora_test$#     DBE_OUTPUT.PRINT_LINE('text:');
ora_test$#     DBE_OUTPUT.PRINT_LINE((a = b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <> b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a > b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a < b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <= b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a >= b)::text);
ora_test$#
ora_test$#     -- Compare with varchar.
ora_test$#     DBE_OUTPUT.PRINT_LINE('varchar:');
ora_test$#     DBE_OUTPUT.PRINT_LINE((a = c)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <> c)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a > c)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a < c)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <= c)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a >= c)::text);
ora_test$#
ora_test$#     -- Compare with varchar2.
ora_test$#     DBE_OUTPUT.PRINT_LINE('varchar2:');
ora_test$#     DBE_OUTPUT.PRINT_LINE((a = d)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <> d)::text);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ora_test$#     DBE_OUTPUT.PRINT_LINE((a > d)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a < d)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <= d)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a >= d)::text);
ora_test$#     -- Compare with char.
ora_test$#     DBE_OUTPUT.PRINT_LINE('char:');
ora_test$#     DBE_OUTPUT.PRINT_LINE((a = e)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <> e)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a > e)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a < e)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <= e)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a >= e)::text);
ora_test$#
ora_test$#     -- Compare with nvarchar2.
ora_test$#     DBE_OUTPUT.PRINT_LINE('nvarchar2:');
ora_test$#     DBE_OUTPUT.PRINT_LINE((a = f)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <> f)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a > f)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a < f)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <= f)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a >= f)::text);
ora_test$#
ora_test$#     RETURN a;
ora_test$# END;
ora_test$# /
CREATE FUNCTION
ora_test=# SELECT func1();
text:
false
true
true
false
false
true
varchar:
false
true
true
false
false
true
varchar2:
false
true
true
false
false
true
char:
false
true
true
false
false
true
nvarchar2:
false
true
true
false
false
true
            func1
-------------------------------
 AAAAAAAAAP//ABPH//AAAAAAAAACAB
(1 row)

-- Compare data of the ROWID type with data of other types that cannot be implicitly converted.
ora_test=# CREATE OR REPLACE FUNCTION func1 RETURN ROWID AS
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ora_test$# DECLARE
ora_test$#     a ROWID;
ora_test$#     b INT;
ora_test$# BEGIN
ora_test$#     -- Prepare data.
ora_test$#     a := 'AAAAAAAAAP//ABPH//AAAAAAAACAB';
ora_test$#     b := 1;
ora_test$#
ora_test$#     -- Compare with int.
ora_test$#     DBE_OUTPUT.PRINT_LINE((a = b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <> b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a > b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a < b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a <= b)::text);
ora_test$#     DBE_OUTPUT.PRINT_LINE((a >= b)::text);
ora_test$#     RETURN a;
ora_testb$# END;
ora_test$# /
CREATE FUNCTION

ora_test=# SELECT func1();
ERROR:  Inconsistent datatypes: expected rowid got integer
CONTEXT:  SQL statement "CALL dbe_output.print_line((a = b)::text)"
PL/pgSQL function func1() line 11 at PERFORM
referenced column: func1

ora_test=# DROP FUNCTION IF EXISTS func1();
DROP FUNCTION
ora_test=# \c postgres
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "postgres" as user "$user".
gaussdb=# DROP DATABASE ora_test;
DROP DATABASE
```

Indexes

Ordinary columns of the ROWID type support B-tree and UB-tree indexes. For details, see **CREATE INDEX**. When ordinary columns of the ROWID type are used for JOIN operations, the HASH JOIN operator is supported. For details, see **Join Queries**.

Examples

```
gaussdb=# CREATE DATABASE ora_test DBCOMPATIBILITY='ORA';
CREATE DATABASE
gaussdb=# \c ora_test
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "ora_test" as user "$user".

ora_test=# CREATE TABLE t1(b ROWID, c INT) WITH (storage_type=ustore);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'c' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
ora_test=# CREATE TABLE t2(b ROWID, c INT) WITH (storage_type=astore);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'c' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
ora_test=# INSERT INTO t1 VALUES('AAAAAAAAAP//BBBBBAAAAAAAAAAAC', generate_series(1,100));
INSERT 0 100
ora_test=# INSERT INTO t2 VALUES('AAAAAAAAAP//BBBBBAAAAACdAAAAC', generate_series(1,100));
INSERT 0 100
ora_test=# --B-tree index.
ora_test=# CREATE INDEX t1_rowid_index ON t1 USING BTREE (b);
NOTICE:  Index type btree is converted to index type ubtree.
CREATE INDEX
ora_test=# CREATE INDEX t2_rowid_index ON t2 USING BTREE (b rowid_ops);
CREATE INDEX
ora_test=# \d t1
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
     Table "public.t1"
 Column |  Type   | Modifiers
--------+---------+-----------
 b      | rowid   |
 c      | integer |
Indexes:
    "t1_rowid_index" ubtree (b) WITH (storage_type=USTORE) TABLESPACE pg_default

ora_test=# \d t2
      Table "public.t2"
 Column |  Type   | Modifiers
--------+---------+-----------
 b      | rowid   |
 c      | integer |
Indexes:
    "t2_rowid_index" btree (b) TABLESPACE pg_default

ora_test=# DROP INDEX t1_rowid_index;
DROP INDEX
ora_test=# DROP INDEX t2_rowid_index;
DROP INDEX
ora_test=# CREATE INDEX t1_rowid_index ON t1 USING UBTREE (b);
CREATE INDEX
ora_test=# CREATE INDEX t2_rowid_index on t2 USING UBTREE (b rowid_ops);
NOTICE:  Index type ubtree is converted to index type btree.
CREATE INDEX
ora_test=# \d t1
      Table "public.t1"
 Column |  Type   | Modifiers
--------+---------+-----------
 b      | rowid   |
 c      | integer |
Indexes:
    "t1_rowid_index" ubtree (b) WITH (storage_type=USTORE) TABLESPACE pg_default

ora_test=# \d t2
      Table "public.t2"
 Column |  Type   | Modifiers
--------+---------+-----------
 b      | rowid   |
 c      | integer |
Indexes:
    "t2_rowid_index" btree (b) TABLESPACE pg_default

ora_test=# DROP INDEX t1_rowid_index;
DROP INDEX
ora_test=# DROP INDEX t2_rowid_index;
DROP INDEX
ora_test=# ANALYZE t1;
ANALYZE
ora_test=# ANALYZE t2;
ANALYZE
-- HASH JOIN operator
ora_test=# EXPLAIN VERBOSE SELECT * FROM t1 JOIN t2 ON t1.b = t2.b;
                                QUERY PLAN
-------------------------------------------------------------------------------------
 Streaming (type: GATHER)  (cost=11.69..14.48 rows=2 width=56)
   Output: t1.b, t1.c, t2.b, t2.c
   Node/s: All datanodes
   ->  Hash Join  (cost=11.62..14.38 rows=2 distinct=[200, 1] width=56)
         Output: t1.b, t1.c, t2.b, t2.c, (Expression Flatten Optimized)
         Hash Cond: (t1.b = t2.b), (Expression Flatten Optimized)
         ->  Seq Scan on public.t1  (cost=0.00..2.50 rows=100 width=28)
               Output: t1.b, t1.c
               Distribute Key: t1.c
         ->  Hash  (cost=10.37..10.37 rows=200 width=28)
               Output: t2.b, t2.c
               ->  Streaming(type: BROADCAST)  (cost=0.00..10.37 rows=200 width=28)
                     Output: t2.b, t2.c
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
                  Spawn on: All datanodes
                  Consumer Nodes: All datanodes
                  ->  Seq Scan on public.t2  (cost=0.00..2.50 rows=100 width=28)
                        Output: t2.b, t2.c
                        Distribute Key: t2.c
(18 rows)
ora_test=# DROP TABLE IF EXISTS t1;
DROP TABLE
ora_test=# DROP TABLE IF EXISTS t2;
DROP TABLE
ora_test=# \c postgres
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "postgres" as user "$user".
gaussdb=# DROP DATABASE ora_test;
DROP DATABASE
```

Type Conversion

Implicit type conversion is allowed between the ROWID type and the string type.

Examples

```
gaussdb=# CREATE DATABASE ora_test DBCOMPATIBILITY='ORA';
CREATE DATABASE
gaussdb=# \c ora_test
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "ora_test" as user "$user".

ora_test=# CREATE TABLE t1(b ROWID, c int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'c' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
ora_test=# INSERT INTO t1 VALUES('AAAAAAAAAP//BBBBBAAAAAAAAAAAAC'::BPCHAR(29));
INSERT 0 1
ora_test=# INSERT INTO t1 VALUES('AAAAAAAAAP//BBBBBAAAAAAAAAAAAC'::VARCHAR(29));
INSERT 0 1
ora_test=# INSERT INTO t1 VALUES('AAAAAAAAAP//BBBBBAAAAAAAAAAAAC'::NVARCHAR2(29));
INSERT 0 1
ora_test=# INSERT INTO t1 VALUES('AAAAAAAAAP//BBBBBAAAAAAAAAAAAC'::TEXT);
INSERT 0 1
ora_test=# CREATE TABLE t2(b BPCHAR(29));
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'b' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
ora_test=# INSERT INTO t2 VALUES('AAAAAAAAAP//BBBBBAAAAAAAAAAAAC'::ROWID);
INSERT 0 1
ora_test=# CREATE TABLE t3(b VARCHAR(29));
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'b' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
ora_test=# INSERT INTO t3 VALUES('AAAAAAAAAP//BBBBBAAAAAAAAAAAAC'::ROWID);
INSERT 0 1
ora_test=# CREATE TABLE t4(b NVARCHAR2(29));
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'b' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
ora_test=# INSERT INTO t4 VALUES('AAAAAAAAAP//BBBBBAAAAAAAAAAAAC'::ROWID);
INSERT 0 1
ora_test=# CREATE TABLE t5(b TEXT);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'b' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
ora_test=# INSERT INTO t5 VALUES('AAAAAAAAAP//BBBBBAAAAAAAAAAAAC'::ROWID);
INSERT 0 1
gaussdb=# DROP TABLE IF EXISTS t1;
DROP TABLE
gaussdb=# DROP TABLE IF EXISTS t2;
DROP TABLE
gaussdb=# DROP TABLE IF EXISTS t3;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DROP TABLE
gaussdb=# DROP TABLE IF EXISTS t4;
DROP TABLE
gaussdb=# DROP TABLE IF EXISTS t5;
DROP TABLE
ora_test=# \c postgres
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "postgres" as user "$user".
gaussdb=# DROP DATABASE ora_test;
DROP DATABASE
```

1.3.24 Vector Data Types

Vector data types include floatvector and boolvector.

- The floatvector data type indicates that multidimensional data contains data of the float type, for example, [1.0,3.0,11.0,110.0,62.0,22.0,4.0].

  NOTE

  - The floatvector member supports only single precision. The single-precision range is –3.402E+38 to +3.402E+38, round to six decimal places.
  - The floatvector does not support NULL, NaN, or Inf as elements. If a vector contains NULL values, the database reports an error.
  - The floatvector cannot be NULL. When NULL values are inserted, updated, or converted as vector data, the database reports an error.
  - In some scenarios (when the execution plan is a streaming plan), the numeric precision of floatvector members is controlled by the GUC parameter **extra_float_digits**. This may cause inconsistent calculation results for the same data. Setting **extra_float_digits** to **3** can avoid this problem.

- The boolvector data type indicates that the data contained in multi-dimensional data is of the Boolean type, for example, [0,0,0,0,0,1,0,0,1,0,0,0].

  NOTE

  - Members of the boolvector type can express Boolean data using **t(T)**/**f(F)**, **y(Y)**/**n(N)**, **1**/**0**, **true**/**false**, **yes**/**no**, or **on**/**off**.
  - The boolvector type does not support NULL, NaN, or Inf as elements. If a vector contains NULL values, the database reports an error.
  - The boolvector cannot be NULL. When NULL values are inserted, updated, or converted as vector data, the database reports an error.

Usage of Vector Types

The following is an example of using vector types:

```
-- Create a table that contains vector types and set data dimensions. Dimensions must be specified for vector types during table creation.
gaussdb=# CREATE TABLE t1(id int unique, repr floatvector(4));
gaussdb=# CREATE TABLE t2(id int unique, repr boolvector(3));

-- Insert data.
gaussdb=# INSERT INTO t1 VALUES(0, '[30, 12, 12, 25]');
gaussdb=# INSERT INTO t2 VALUES(1, '[1, 0, 1]');

-- Drop tables.
gaussdb=# DROP TABLE t1;
gaussdb=# DROP TABLE t2;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.4 Character Sets and Collations

A character set provides character encoding rules, and a collation provides character sorting rules. This section describes the character sets and collations in MySQL-compatible GaussDB (that is, **sql_compatibility** is set to **'MYSQL'**). The following character sets, collation rules, and syntax are supported only in MySQL-compatible mode:

For details about the character sets supported by GaussDB, see "ENCODING" in **CREATE DATABASE**. For details about the supported collations, see the PG_COLLATION system catalog.

Some character sets have default collations in MySQL-compatible mode. For details, see **Table 1-398**.

The description of character sets and collations is as follows:

- Each character set has one or more collations and has only one default collation.
- Each collation has only one associated character set.
- The sorting results of the same data may vary according to collations.
- In GaussDB, utf8mb4 and UTF-8 are the same character set.
- When **sql_compatibility** is set to **'MYSQL'**, the binary and SQL_ASCII character sets are the same.
- You are advised to select the same character set for table columns and **server_encoding** to avoid performance loss caused by transcoding.

GaussDB supports the following functions:

- Multiple character sets can be used to store character strings.
- Collations can be used to compare character strings.
- Database-level, schema-level, table-level, and column-level character sets and collations are supported.

  NOTE

  Character strings with different character sets and collations cannot be used in the same server, database, table, or SQL statement.

1.4.1 Character Sets and Collations of the Client Connection

When data containing character set attributes is transmitted between the server and client, encoding conversion is automatically performed. After receiving an SQL statement from a client, the server converts the character set of the SQL statement from the client character set client_encoding to the database character set server_encoding. Before the query result data is sent to the client, the data is encoded and converted to the client_encoding character set of the client.

System parameters:

- **server_encoding**

  Character set specified during database creation. For details, see **CREATE DATABASE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **client_encoding**

  Character set of the client, which can be modified by using the SET NAMES statement. For details, see **SET**. For details about the parameters, see **client_encoding**.

- **character_set_connection**

  Default character set applied when no character set is specified for character string constants in SQL statements. For details about this parameter, see **character_set_connection**.

- **collation_connection**

  Default collation applied when no collation is specified for character string constants in SQL statements. For details about this parameter, see **collation_connection**.

- **character_set_results**

  Character set of the returned result. For details about this parameter, see **character_set_results**.

  NOTE

  - For an expression that converts a non-character type object to a character type, the result character set and collation are **character_set_connection** and **collation_connection**.
  - By default, the character set and collation number of a bound parameter of the character type are the values of **character_set_connection** and **collation_connection**, respectively. Any value entered for a bound parameter is considered belonging to the preceding character set.
  - During character data conversion, the character encoding is verified. If the character encoding does not meet the requirements, an exception is thrown and an error message is displayed.

1.4.2 Database-level Character Sets and Collations

When creating a database, you can specify the character set sequence of the database. For details about how to create a database, see **CREATE DATABASE**.

```
CREATE DATABASE [IF NOT EXISTS] database_name
        [ ENCODING [=] encoding ] |
        [ LC_COLLATE [=] lc_collate ] |
        [ LC_CTYPE [=] lc_ctype ] |;
```

Parameters:

- **database_name**

  Specifies the database name.

  Value range: a string. It must comply with the identifier naming convention.

- **ENCODING [=] encoding**

  Specifies the character encoding used by the new database. The value can be a string (for example, **SQL_ASCII**) or an integer.

- **LC_COLLATE [=] 'lc_collate'**

  Specifies the character set used by the new database. For example, set this parameter by using **lc_collate = 'zh_CN.gbk'**.

  This parameter affects the sorting behavior of strings (including the sorting result using ORDER BY and the sorting order using indexes on text columns). The default is to use the order rule of the template database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: character sets supported by the OS.

- **LC_CTYPE [=] 'lc_ctype'**

  Specifies the character type used by the new database. For example, set this parameter by using **lc_ctype = 'zh_CN.gbk'**. The use of this parameter affects the character type, such as uppercase letters, lowercase letters, and digits. By default, the character type of the template database is used.

  Value range: character types supported by the OS.

  NOTE

  - The database-level character set and collation syntax can be used in all schemas. For details about the syntax, see **CREATE DATABASE**.
  - The **LC_COLLATE** and **LC_CTYPE** parameters do not support the collations specific to MySQL-compatible mode. The parameter value range depends on the character sets supported by the local environment. You can run the **locale -a** command to view the parameter value range.

1.4.3 Schema-level Character Sets and Collations

Create a schema and specify the default character set and collation. For details about how to create a schema, see **CREATE SCHEMA**.

```
CREATE SCHEMA schema_name
 [ [DEFAULT] CHARACTER SET | CHARSET [ = ] default_charset ]
 [ [DEFAULT] COLLATE [ = ] default_collation ];
```

Modify the default character set and collation attributes of a schema. For details about how to modify a schema, see **ALTER SCHEMA**.

```
ALTER SCHEMA schema_name
    [ [DEFAULT] CHARACTER SET | CHARSET [ = ] default_charset ] [ [DEFAULT] COLLATE [ = ]
default_collation ];
```

Parameters:

- **schema_name**

  Specifies the schema name.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **default_charset**

  Specifies the default character set of a schema. If you specify this parameter separately, the default collation of the schema will be set to the default collation of the specified character set.

- **default_collation**

  Specifies the default collation of a schema. If you specify this parameter separately, the default character set of the schema will be set to the character set corresponding to the specified collation.

GaussDB selects a character set and collation of a schema in the following ways:

- If both **default_charset** and **default_collation** are specified, character set specified in **default_charset** and the collation specified in **default_collation** will apply. In addition, **default_charset** and **default_collation** must match. Otherwise, an error is reported.
- If only **default_charset** is specified, the character set **default_charset** and its default collation are used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If only **default_collation** is specified, the collation **default_collation** and its corresponding character set are used.
- If neither **default_charset** nor **default_collation** is specified, the schema has no default character set or default collation.

NOTE

- If only **default_charset** is specified, it can only be specified to a character set with a default collation. Specifying a character set without a default collation will trigger an error.
- Only the collations in MySQL-compatible mode (**sql_compatibility** is set to **'MYSQL'**) support **default_collation**. If another collation is specified, an error is reported.
- The character set of the new schema must be the same as **server_encoding** of the database.

Examples:

```
-- Set only the character set. The collation is the default collation of the character set.
gaussdb=# CREATE SCHEMA test CHARSET utf8;

-- Set only the collation. The character set is the character set associated with the collation.
gaussdb=# CREATE SCHEMA test COLLATE utf8_bin;

-- Set both the character set and collation. The character set and collation must correspond to each other.
gaussdb=# CREATE SCHEMA test CHARSET utf8 COLLATE utf8_bin;

-- Change the default character set of test to utf8mb4 and the default collation to utf8mb4_bin.
gaussdb=# ALTER SCHEMA test CHARSET utf8mb4 COLLATE utf8mb4_bin;
```

1.4.4 Table-level Character Sets and Collations

Set the default character set and collation of a table. For details about how to create a table, see **CREATE TABLE**.

```
CREATE TABLE table_name (column_list)
    [ [DEFAULT] CHARACTER SET | CHARSET [ = ] default_charset ]
    [ [DEFAULT] COLLATE [ = ] default_collation ]
```

Modify the default character set and default collation of a table. The modification does not affect the existing columns in the table. For details about how to modify a table, see **ALTER TABLE**.

```
ALTER TABLE table_name
    [ [DEFAULT] CHARACTER SET | CHARSET [ = ] default_charset ]
    [ [DEFAULT] COLLATE [ = ] default_collation ]
```

NOTE

Currently, you cannot modify the default character set and default collation of a table to the specified values, set the character set and collation of all columns with character type to the specified value, and convert the data in the column to new character set encoding. The syntax is as follows:

```
ALTER TABLE table_name
    CONVERT TO CHARACTER SET | CHARSET charset [ COLLATE collation ]
```

Parameters

- **table_name**

  Specifies the name of the table.

- **default_charset**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the default character set of a table. If you specify this parameter separately, the default collation of the table will be set to the default collation of the specified character set.

- **default_collation**

  Specifies the default collation of a table. If you specify this parameter separately, the default character set of the table will be set to the character set corresponding to the specified collation.

GaussDB selects a character set and collation of a table in the following ways:

- If both **default_charset** and **default_collation** are specified, character set specified in **default_charset** and the collation specified in **default_collation** will apply. In addition, **default_charset** and **default_collation** must match. Otherwise, an error is reported.
- If only **default_charset** is specified, the character set **default_charset** and its default collation are used.
- If only **default_collation** is specified, the collation **default_collation** and its corresponding character set are used.
- If neither **default_charset** nor **default_collation** is specified, the default character set and default collation of the schema where the table is located will apply.

NOTE

- If only **default_charset** is specified, it can only be specified to a character set with a default collation. Specifying a character set without a default collation will trigger an error.
- Only the collations in MySQL-compatible mode (**sql_compatibility** is set to **'MYSQL'**) support **default_collation**. If another collation is specified, an error is reported.
- If the default collation of a table is binary, the text types whose collation is not specified in the table are converted to the corresponding binary type, and the collation is set to binary.
- Currently, the default character set of a table must be server_encoding of the database.
- Currently, the data of columns of the character type in the table cannot be converted into encoding.

Example:

```
-- Set only the character set. The collation is the default collation of the character set.
gaussdb=# CREATE TABLE test(c1 text) CHARSET utf8;

-- Set only the collation. The character set is the character set associated with the collation.
gaussdb=# CREATE TABLE test(c1 text) COLLATE utf8_bin;

-- Set both the character set and collation. The character set and collation must correspond to each other.
gaussdb=# CREATE TABLE test(c1 text)  CHARSET utf8 COLLATE utf8_bin;

-- Change the default character set of the table to utf8mb4 and the default collation to utf8mb4_bin.
gaussdb=# ALTER TABLE test CHARSET utf8mb4 COLLATE utf8mb4_bin;

-- Drop the table.
gaussdb=# DROP TABLE test;
```

1.4.5 Column-Level Character Sets and Collations

You can set the column-level character set and collation for each column of the string type (CHAR, VARCHAR, or TEXT).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
CREATE TABLE table_name (
    column_name data_type
 [ CHARACTER SET | CHARSET charset ]
    [ COLLATE collation ]
);
```

Parameters:

- **table_name**

  Specifies the name of the table.

- **data_type**

  Specifies the data type of a column. The value can only be character set and collation of the string type.

- **CHARACTER SET | CHARSET charset**

  Specifies the character set of a column. If this parameter is specified separately, the collation of the column is set to the default collation of the specified character set.

- **COLLATE collation**

  The COLLATE clause specifies the collation of a column (the data type of the column must support collation). If no collation is specified, the default collation is used.

GaussDB selects a character set and collation of a table column in the following ways:

- If both **charset** and **collation** are specified, **charset** and **collation** are used. **charset** and **collation** must correspond to each other. Otherwise, an error is reported.
- If only **charset** is specified, the character set **charset** and its default collation are used.
- If only **collation** is specified, collation specified in **collation** and its corresponding character set will apply.
- If neither **charset** nor **collation** is specified, the default character set and collation of the table are used.

NOTE

- If only **charset** is specified, it can only be specified to a character set with a default collation. Specifying a character set without a default collation will trigger an error.
- Only the collations in MySQL-compatible mode support **collation**. If another collation is specified, an error is reported.
- If a table column is of the text type and the specified collation is binary, the text type is converted to the corresponding binary type and the collation is the specified binary collation.
- The character set of the partition key of the partitioned table must be the same as that of the database.
- Currently, the default character set of a table column must be server_encoding of the database.

Examples:

```
-- Set only the character set. The collation is the default collation of the character set.
gaussdb=# CREATE TABLE test(c1 text CHARSET utf8);

-- Set only the collation. The character set is the character set associated with the collation.
gaussdb=# CREATE TABLE test(c1 text COLLATE utf8_bin);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Set both the character set and collation. The character set and collation must correspond to each other.
gaussdb=# CREATE TABLE test(c1 text CHARSET utf8 COLLATE utf8_bin);
```

1.4.6 Character Sets and Collations of Expressions of the String Type

Each expression of the string type contains character sets and collation attributes.

In the MySQL-compatible database (that is, **sql_compatibility** is set to **'MYSQL'**), if **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s2'**, the default character sets and collations of string constants are determined by the system parameters **character_set_connection** and **collation_connection**. If they are not set, the default character set is the same as **server_encoding** of the database, and the default collation is **default**.

Character set syntax:

GaussDB does not support the character set of string constants specified by the following syntax:

```
[_charset_name]'string'
```

Collation syntax:

You can also specify the collation for expressions of other character string types.

```
EXPRESSION [COLLATE collation_name]
```

Parameter:

COLLATE collation_name

Collation name, which is the collation attribute of the string.

- The data type of the expression must be a data type that supports collations.
- The specified collation must be supported by the character set of the expression.

Example:

```
-- Use the COLLATE statement to specify the collation.
gaussdb=# SELECT 'a' COLLATE utf8mb4_general_ci = 'A';
```

1.4.7 Rules for Combining Character Sets and Collations

In the MySQL-compatible database (that is, **sql_compatibility** is set to **'MYSQL'**), if **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s2'**, the system processes expressions of different character sets and collations based on the priority to determine the collation and character set of the expression used for string comparison.

Collation priority

The priorities of different expressions in descending order are as follows:

- The COLLATE syntax has the highest priority.
- Expressions with collation conflicts (for example, two character strings with different collations) exist.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Columns of data types that support collation, user-defined variables, stored procedure parameters, and CASE expressions.
- Specific system functions (such as version() and opengauss_version() function expressions).
- String constants and bound parameters.
- NULL expressions.
- If a data type of an expression does not support collation, the expression has the lowest priority.

When the collations of two expressions are different, the collation of the expression with the highest priority is used.

Example:

```
gaussdb=# CREATE TABLE t_utf8(c1 varchar(16) CHARACTER SET utf8mb4 COLLATE utf8mb4_bin);
gaussdb=# INSERT INTO t_utf8 VALUES('STRING');

-- If the utf8mb4_bin collation is used for comparison, the result is false.
gaussdb=# SELECT c1 = 'string' AS result FROM t_utf8;
 result
--------
 f
(1 row)

-- If the utf8mb4_general_ci collation is used for comparison, the result is true.
gaussdb=# SELECT c1 = 'string' COLLATE utf8mb4_general_ci AS result FROM t_utf8;
 result
--------
 t
(1 row)

-- Define the collation of the bound parameter $1 as collation_connection.
gaussdb=# PREPARE test_collation(text) AS SELECT c1 = $1 AS result FROM t_utf8;

-- The collation of the bound parameter is at the same level as that of the string constant. Even if the input expression contains an explicit collation, the collation of c1 is still used for comparison.
gaussdb=# EXECUTE test_collation('string' COLLATE utf8mb4_general_ci);
 result
--------
 f
(1 row)

-- The CASE expression is at the same level as the c1 column. Even if the expression contains an explicit collation, the collation of the c1 column is still used for comparison. The two collations are different. "same level" is displayed.
gaussdb=# SELECT CASE 'string' COLLATE utf8mb4_general_ci WHEN c1 THEN 'different level' ELSE 'same level' END AS result FROM t_utf8;
   result
------------
 same level
(1 row)

-- The IN subquery has the same level as the c1 column. Even if the expression contains an explicit collation, the collation of c1 is still used for comparison. The two collations are different.
gaussdb=# SELECT c1 FROM t_utf8 WHERE c1 in (SELECT 'string' COLLATE utf8mb4_general_ci);
 c1
----
(0 rows)
```

If the collations of two expressions with the same priority are different, the following processing method is used:

- If the two character sets are the same, the collation suffixed with **_bin** is preferred.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If the two character sets are the same, the default collation is not preferred.
- If the preceding conditions are not met, the two expressions are marked as a collation conflict, and the collations are marked as invalid.
  - If a conflict occurs because the COLLATE syntax specifies different collations for the same character set, an exception is displayed.
  - If the two conflicting collations are supported in MySQL-compatible mode (**sql_compatibility** is set to **'MYSQL'**), an exception occurs.
  - If the conflicting character set is different from server_encoding of the database, an exception occurs.
  - If an invalid collation is used for sorting operations (such as > and <), an exception occurs.
  - During equality comparison of character strings, if the collation is invalid, the character strings are compared as binary values to determine whether they are equal.

Examples:

```
gaussdb=# CREATE TABLE t_utf8mb4_charset(
    c_utf8_bin varchar(16) character set utf8mb4 collate utf8mb4_bin,
    c_utf8_uni varchar(16) character set utf8mb4 collate utf8mb4_unicode_ci,
    c_utf8_gen varchar(16) character set utf8mb4 collate utf8mb4_general_ci);
gaussdb=# INSERT INTO t_utf8mb4_charset VALUES('STRING', 'String', 'string');

-- The utf8mb4_bin collation is preferentially used for comparison. The result is false.
gaussdb=# SELECT c_utf8_bin = c_utf8_uni FROM t_utf8mb4_charset;

-- Collation conflict. Binary comparison is performed, and the result is false.
gaussdb=# SELECT c_utf8_uni = c_utf8_gen FROM t_utf8mb4_charset;

-- Conflict of the explicitly specified collation. An exception is reported.
gaussdb=# SELECT c_utf8_uni COLLATE utf8mb4_unicode_ci = c_utf8_gen COLLATE utf8mb4_general_ci
FROM t_utf8mb4_charset;
```

NOTE

- Only the character sets of objects and expressions of the string type (excluding char, name, and clob) can be different from those of the database.
- Data types such as ARRAY, XML, JSON, and TSVECTOR contain text data. The character sets in the text data of the objects and expressions of these data types must be the same as the character sets of the database.
- According to the rules for combining character sets and collations, the character sets corresponding to C, POSIX, and DEFAULT collations are server_encoding.

1.5 Constant and macro

Table 1-28 lists the constants and macros that can be used in GaussDB.

Table 1-28 Constant and macro

Parameter	Description	Example
CURRENT_CATALOG	Current database.	testdb=# SELECT CURRENT_CATALOG; current_database ---------- testdb (1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Example
CURRENT_ROLE	Current user.	gaussdb=# SELECT CURRENT_ROLE; current_user -------------- omm (1 row)
CURRENT_SCHEMA	Current database schema.	gaussdb=# SELECT CURRENT_SCHEMA; current_schema ---------------- public (1 row)
CURRENT_USER	Current user.	gaussdb=# SELECT CURRENT_USER; current_user -------------- omm (1 row)
LOCALTIMESTAMP	Current session time (without time zone).	gaussdb=# SELECT LOCALTIMESTAMP; timestamp ---------------------------- 2015-10-10 15:37:30.968538 (1 row)
NULL	NULL value.	-
SESSION_USER	Current system user.	gaussdb=# SELECT SESSION_USER; session_user -------------- omm (1 row)
SYSDATE	Current system date.	gaussdb=# SELECT SYSDATE; sysdate --------------------- 2015-10-10 15:48:53 (1 row)
USER	Specifies the current user, also called **CURRENT_USER**.	gaussdb=# SELECT USER; current_user -------------- omm (1 row)

1.6 Functions and Operators

An operator can process one or more operands and can be placed before, after, or between operands. Results are returned after the processing.

A function is a set of encapsulated service logic for implementing specific functions. It may or may not have parameters. A function has a return type, and once executed, the result is returned.

Users can modify system functions. However, after the modification, the meaning of the functions may change, which results in disorder in system control. Therefore, users are not allowed to manually modify system functions.

Functions are classified into system functions and syntax functions. Some functions are both system functions and syntax functions. For details, see **Table 1-29**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- System functions: They are registered in the pg_proc system catalog by using the registration mechanism. You can query the pg_proc system catalog to obtain the metadata of system functions.
- Syntax functions: They are directly defined by SQL syntax rules and have dedicated syntax formats. For details about the syntax functions, see **Table 1-29**.

Table 1-29 Syntax functions

Name	ORA-compatible Mode	MySQL-compatible Mode	Other Compatibility Mode
adddate	N/A	Syntax and system functions	Syntax and system functions
cast	Syntax function	Syntax function	Syntax function
coalesce	Syntax function	Syntax function	Syntax function
collation for	Syntax function	Syntax function	Syntax function
convert	System function	Syntax and system functions	Syntax and system functions
current_catalog	Syntax function	Syntax function	Syntax function
current_date	Syntax function	Syntax function	Syntax function
current_role	Syntax function	Syntax function	Syntax function
current_schema	Syntax and system functions	Syntax and system functions	Syntax and system functions
current_time	Syntax function	Syntax function	Syntax function
current_timestamp	Syntax and system functions	Syntax function	Syntax function
current_user	Syntax and system functions	Syntax and system functions	Syntax and system functions
curtime	N/A	System function	N/A
date_add	N/A	Syntax and system functions	Syntax and system functions
date_sub	N/A	Syntax and system functions	Syntax and system functions
db_jsobj	N/A	Syntax function	Syntax function
dbtimezone	Syntax and system functions	N/A	N/A
default	N/A	Syntax function	N/A
extract	Syntax function	Syntax function	Syntax function

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	ORA-compatible Mode	MySQL-compatible Mode	Other Compatibility Mode
get_format	N/A	Syntax and system functions	Syntax and system functions
greatest	Syntax function	Syntax function	Syntax function
ifnull	N/A	Syntax function	N/A
last_day	System function	Syntax and system functions	Syntax and system functions
least	Syntax function	Syntax function	Syntax function
lnnvl	Syntax function	N/A	N/A
localtime	Syntax and system functions	Syntax and system functions	Syntax and system functions
localtimestamp	Syntax function	Syntax function	Syntax function
now	Syntax and system functions	Syntax and system functions	Syntax and system functions
nullif	Syntax function	Syntax function	Syntax function
nvl	Syntax function	Syntax function	Syntax function
nvl2	Syntax function	N/A	N/A
overlay	Syntax and system functions	Syntax and system functions	Syntax and system functions
position	Syntax and system functions	Syntax and system functions	Syntax and system functions
regexp_like	Syntax and system functions	Syntax and system functions	Syntax and system functions
rownum	Syntax function	Syntax function	Syntax function
sessiontimezone	Syntax function	N/A	N/A
session_user	Syntax and system functions	Syntax and system functions	Syntax and system functions
subdate	N/A	Syntax and system functions	Syntax and system functions
substr	Syntax and system functions	Syntax and system functions	Syntax and system functions
substring	Syntax and system functions	Syntax and system functions	Syntax and system functions
sysdate	Syntax and system functions	Syntax and system functions	Syntax and system functions

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	ORA-compatible Mode	MySQL-compatible Mode	Other Compatibility Mode
timestampadd	N/A	Syntax function	Syntax function
timestampdiff	N/A	Syntax function	N/A
treat	Syntax function	Syntax function	Syntax function
trim	Syntax function	Syntax function	Syntax function
user	Syntax function	Syntax function	Syntax function
utc_date	N/A	System function	N/A
utc_time	N/A	System function	N/A
utc_timestamp	N/A	System function	N/A
xmlcast	Syntax function	N/A	N/A
xmlconcat	Syntax function	Syntax function	Syntax function
xmlelement	Syntax function	Syntax function	Syntax function
xmlexists	Syntax and system functions	Syntax and system functions	Syntax and system functions
xmlforest	Syntax function	Syntax function	Syntax function
xmlparse	Syntax function	Syntax function	Syntax function
xmlpi	Syntax function	Syntax function	Syntax function
xmlquery	Syntax function	N/A	N/A
xmlroot	Syntax function	Syntax function	Syntax function
xmlserialize	Syntax function	Syntax function	Syntax function
year	N/A	System function	N/A

NOTE

When the GUC parameter **behavior_compat_options** includes **'enable_funcname_with_argsname'**, the projection alias displays the complete function.

1.6.1 Logical Operators

Common logical operators include AND, OR, and NOT. The operation result has three values: **TRUE**, **FALSE**, and **NULL**. NULL indicates unknown. Their priorities are NOT > AND > OR.

Table 1-30 lists the calculation rules, where a and b represent logical expressions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-30 Operation rules

a	b	a AND b Result	a OR b Result	NOT a Result
TRUE	TRUE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE	FALSE
TRUE	NULL	NULL	TRUE	FALSE
FALSE	FALSE	FALSE	FALSE	TRUE
FALSE	NULL	FALSE	NULL	TRUE
NULL	NULL	NULL	NULL	NULL

NOTE

- The operators AND and OR are commutative, that is, you can switch the left and right operand without affecting the result.
- Operations on XML data are not supported.

1.6.2 Comparison Operators

Comparison operators are available for the most data types and return Boolean values.

All comparison operators are binary operators. Only data types that are the same or can be implicitly converted can be compared using comparison operators.

Table 1-31 describes comparison operators provided by GaussDB.

Table 1-31 Comparison operations

Operator	Description
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
=	Equal to
<>, !=, or ^=	Not equal to

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- For comparison operators <=, <>, >=, and ^=, if there is a space between two symbols, it does not affect normal operations. For !=, if there is a space between the exclamation mark (!) and an equal sign (=), the exclamation mark will be identified as factorial, which may cause the result to be inconsistent with the expected result.
- Comparison operators are available for all relevant data types. All comparison operators are binary operators that returned values of Boolean type. The calculation priority of the inequality sign is higher than that of the equality sign. If the entered data type is different and cannot be implicitly converted, the comparison fails. For example, an expression such as **1 < 2 < 3** is invalid because the less-than sign (<) cannot be used to compare Boolean values and 3.
- Each comparison operator has a corresponding function in the PG_PROC system catalog. If the value of the **proleakproof** attribute of the corresponding function is **f**, the function is not used to prevent data leakage. If a user only has the permission on a system view, but does not have the permission on the corresponding table, the query plan may not be optimal when the user queries the system view.
- This operator does not support data of the XML type.
- If the two data types to be compared are different or cannot be implicitly converted to each other, they are first implicitly converted to a third convertible data type before comparison. In this case, the comparison result may be meaningless.

  Example:

```
gaussdb=# EXPLAIN verbose SELECT to_datea('2010-10-10','yyyy-mm-dd') > 1.5;
                                 QUERY PLAN
-----------------------------------------------------------------------------------
 Result  (cost=0.00..0.02 rows=1 width=0)
   Output: ((to_datea('2010-10-10'::text, 'yyyy-mm-dd'::text))::text > '1.5'::text)
(2 rows)
```

1.6.3 Character Processing Functions and Operators

Character processing functions and operators provided by GaussDB are for concatenating strings with each other, concatenating strings with non-strings, and matching the patterns of strings. Except length-related functions, other character processing functions and operators do not support CLOBs larger than 1 GB as parameters.

NOTE

If invalid characters are found in input parameters during character comparison, **false** will be returned by default.

- bin(number)

  Description: Returns a binary string of a given number.

  Parameters: See **Table 1-32**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-32 bin parameters

Parameter	Type	Description
number	• Integer types: tinyint, smallint, mediumint, int, and bigint. • Unsigned integer types: tinyint unsigned, smallint unsigned, int unsigned, and bigint unsigned. • Character and text types: char, varchar, tinytext, text, mediumtext, and longtext. Only numeric integer strings are supported, and the integer range is within the bigint range. • Floating-point types: float, real, and double. • Fixed-point types: numeric, decimal, and dec. • Boolean type.	Given number.

Return type: text.

Example:

```
gaussdb=# SELECT bin(5);
  bin
--------
  101
(1 row)
```

NOTE

- The function takes effect only when **sql_compatibility** is set to **'MYSQL'**.
- If the value of the input parameter **number** is a decimal, round it down.
- If the absolute value of the input parameter **number** exceeds the maximum value of the bigint unsigned type, the parameter is converted to the upper limit.

- bit_length(string)

  Description: Returns the number of bits in the input string.

  Return type: int.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT bit_length('world');
 bit_length
------------
         40
(1 row)
```

- btrim(string text [, characters text])

  Description: Removes the longest string containing only characters specified by **characters** (a space by default) from the start and end of **string**.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT btrim('sring' , 'ing');
   btrim
  -------
   sr
  (1 row)
  ```

- char_length(string) or character_length(string)

  Description: Specifies the number of characters in a string.

  Return type: int.

  Example:

  ```
  gaussdb=# SELECT char_length('hello');
   char_length
  -------------
             5
  (1 row)
  ```

- dump(expr[, return_fmt [, start_position [, length]]])

  Description: Returns the data type code, byte length, and internal representation of the input expression.

  Parameters: **return_fmt** specifies the number system of the internal representation, **start_position** specifies the byte from which the internal representation starts, and **length** specifies the length of the data to be read.

  Return type: text.

  NOTE

  This function takes effect in ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- elt(pos, str1, str2,...)

  Description: Returns a string at a specified position. This function takes multiple string parameters and returns the corresponding string based on the position parameter **pos**.

  Parameters: See **Table 1-33**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-33 elt parameters

Parameter	Type	Description
pos	• Integer types: tinyint, smallint, mediumint, int, and bigint. • Unsigned integer types: tinyint unsigned, smallint unsigned, and int unsigned. • Character and text types: char, varchar, tinytext, text, mediumtext, and longtext. Only numeric integer strings are supported, and the integer range is within the bigint range. • Floating-point types: float, real, and double. • Fixed-point types: numeric, decimal, and dec. • Boolean type.	Specified position of the parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
str1, str2,...	• Integer types: tinyint, smallint, mediumint, int, and bigint. • Unsigned integer types: tinyint unsigned, smallint unsigned, int unsigned, and bigint unsigned. • Character and text types: char, varchar, tinytext, text, mediumtext, and longtext. • Floating-point types: float, real, and double. • Fixed-point types: numeric, decimal, and dec. • Boolean type. • Large object types: tinyblob, blob, mediumblob, and longblob. • Date types: datetime, timestamp, date, and time.	List of character strings.

Return type: text.

Example:

```
gaussdb=# SELECT elt(3, 'a', 'b', 'c');
 elt
-------
   c
(1 row)
```

NOTE

- The function takes effect only when **sql_compatibility** is set to **'MYSQL'**.
- If the input parameter **pos** is less than 1 or exceeds the number of parameters, **NULL** is returned.

- field(str, str1, str2, str3,...)

  Description: Returns the position number of **str** in the parameter list (for example, **str1**, **str2**, **str3**) using the field function. The position number starts from 1 and increases in ascending order. If no match is found or the value of

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

the **str** parameter is **NULL**, **0** is returned. This function uses different comparison rules based on the types of input parameters.

- If all input parameters are of the numeric type, the comparison is performed based on the numeric rule.
- If all input parameters are of the non-numeric type, the comparison is performed based on the character string rule.
- If the input parameters contain both digits and non-digits, the comparison is performed based on the double type.

Parameters: See **Table 1-34**.

Table 1-34 field parameters

Parameter	Type	Description
str, str1, str2, str3,...	• Integer types: tinyint, smallint, mediumint, int, bigint, tinyint unsigned, smallint unsigned, int unsigned, and bigint unsigned. • Boolean types: true and false. • Floating-point types: float(p), float, real, double, float(m, d), double(m, d), and real(m, d). • Fixed-point types: numeric, decimal, and dec. • Text types: tinytext, text, mediumtext, and longtext. • Character string types: char and varchar. • Large object types: tinyblob, blob, mediumblob, and longblob. • Date types: datetime, timestamp, date, and time.	List of character strings.

Return type: bigint

Example:

```
gaussdb=# SELECT field( 'abc','1',1,'abc','abcd' );
 field
-------
     3
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The field function takes effect only when **sql_compatibility** is set to **'MYSQL'**.
- Since the version whose **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s1'**, the sql_mode parameter **pad_char_to_full_length** specifies whether to add spaces at the end of the char type, which affects the field comparison result. For details, see **Table 1-8**.
- Since the version whose **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s1'**, the behavior of the character, binary, numeral, and date and time types is M-compatible. This affects the result of field comparison. For details, see **Data Types**. To ensure consistent precision, it is advisable to use the NUMERIC type rather than floating-point (under the numeric type). The use of floating-point may lead to varying precision compared to M-compatible mode, due to differences in connection parameter settings.
- Since the version whose **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s2'**, characters can be escaped and constant character strings can obtain collations. The collations affect the field comparison result. For details, see the SET NAMES syntax in **SET**. For details about the rules for combining different collations of the character types, see **Rules for Combining Character Sets and Collations**.

- insert(str1, pos, len, str2)

  Description: Returns the processing result of *str1*. The substring starts from *pos* and the number of *len* characters in the character string is replaced with *str2*. If any input parameter is **NULL**, the return value is **NULL**.

  Parameters: See **Table 1-35**.

Table 1-35 insert parameters

Parameter	Type	Description
pos, len	• Integer types: tinyint, smallint, mediumint, int, and bigint. • Unsigned integer types: tinyint unsigned, smallint unsigned, int unsigned, and bigint unsigned. • Character and text types: char, varchar, tinytext, text, mediumtext, and longtext. Only numeric integer strings are supported. • Floating-point types: float, real, and double. • Fixed-point types: numeric, decimal, and dec.	**pos** indicates the parameter in the specified position, and **len** indicates the replacement length.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
str1, str2	• NULL. • Integer types: tinyint, smallint, mediumint, int, bigint, tinyint unsigned, smallint unsigned, int unsigned, and bigint unsigned. However, the integer value exceeds the range from 81 negative nines to 81 positive nines, and the return value is different from that of MySQL. This difference between GaussDB and MySQL is inherent and does not result from this function. • Floating-point and fixed-point types: float, real, and double; numeric, decimal, and dec. When using scientific notation, GaussDB displays trailing zeros, unlike MySQL, which does not. This difference between GaussDB and MySQL is inherent and does not result from this function. • Character string types: char and varchar. • Text types: tinytext, text, mediumtext, and longtext. • Large object types: tinyblob, blob, mediumblob, and longblob.	String.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
	• Date types: datetime, timestamp, date, and time.	

Return value type: The return value of the function is of the text type (both **s1** and **s2** are of the text type) or bytea type (any of **s1** or **s2** is of the bytea type).

Example:

```
gaussdb=# SELECT INSERT('abcdef',2,3,'gg');
 insert
--------
 aggef
(1 row)
```

NOTE

- The insert function takes effect only when **sql_compatibility** is set to **'MYSQL'**.
- The value range of the input parameter of the Int64 type is limited. If the value exceeds the value range of –9223372036854775808 to 9223372036854775807, an error is reported. MySQL has no restriction on the range of input parameters of the numeric type. If an exception occurs, an alarm is generated and the upper or lower limit is used. Input parameters of the character string are restricted. The maximum length of the input parameter of the text type is 2^30 – 5 bytes, and the maximum length of the input parameter of the bytea type is 2^30 – 512 bytes.

- instr(text, text, int, int)

  Description: Returns the position of the *n*th matched occurrence of the second parameter in the first parameter, starting from the position specified by the third parameter (*n* is the value of the fourth parameter). The third parameter indicates the start position for matching, and the fourth parameter indicates the number of matches.

  Return type: int.

  Example:

  ```
  gaussdb=# SELECT instr( 'abcdabcdabcd', 'bcd', 2, 2 );
   instr
  -------
      6
  (1 row)
  ```

- instrb(text, text, int, int)

  Description: Returns the position of the *n*th matched occurrence of the second parameter in the first parameter, starting from the position specified by the third parameter (*n* is the value of the fourth parameter). The third parameter indicates the start position for matching, and the fourth parameter indicates the number of matches. Different from the instr function, instrb is calculated in bytes and is not affected by the character set in use.

  Return type: int.

  Example:

  ```
  gaussdb=# SELECT instrb( 'abcdabcdabcd', 'bcd', 2, 2 );
   instrb
  -------
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
   6
(1 row)
```

NOTE

- This function takes effect in ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s1**.
- If the values of *int1* and *int2* are decimals, the values are truncated instead of being rounded off.

- lengthb(text/bpchar)

  Description: Obtains the number of bytes of a specified string.

  Return type: int.

  Example:

```
gaussdb=# SELECT lengthb('hello');
 lengthb
---------
       5
(1 row)
```

- lengthc(text/bpchar)

  Description: Returns the length of a specified string, calculated according to Unicode character encoding rules.

  Return type: int.

  Example:

```
gaussdb=# SELECT lengthc('hello');
 lengthc
---------
       5
(1 row)
```

- left(str text, n int)

  Description: Returns the first *n* characters in a string. When *n* is negative, all but the last */n/* characters are returned.

  Return type: text.

  Example:

```
gaussdb=# SELECT left('abcde', 2);
 left
------
 ab
(1 row)
```

- length(string bytea, encoding name)

  Description: Specifies the number of characters in **string** in the given encoding format. In this encoding format, **string** must be valid.

  Return type: int.

  Example:

```
gaussdb=# SELECT length('jose', 'UTF8');
 length
--------
      4
(1 row)
```

NOTE

If the length of the bytea type is queried and UTF-8 encoding is specified, the maximum length can only be **536870888**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- locate(substr, str[, pos])

  Description: Returns the position where *substr* appears for the first time in *str*, starting from *pos* (**1** by default). If *substr* cannot be found in *str*, this function returns **0**.

  Parameters: See **Table 1-36**.

  Table 1-36 locate parameters

Parameter	Type	Description
substr, str	• Integer types: tinyint, smallint, mediumint, int, bigint, tinyint unsigned, smallint unsigned, int unsigned, and bigint unsigned. • Floating-point types: float and double. • Arbitrary precision type: numeric. • Character types: char, varchar, and text. • Binary types: bytea and blob. • Date/Time types: date, time, datetime, and timestamp.	• **substr** (required): substring to be searched for. • **str** (required): character string to be searched for.
pos	• Integer types: tinyint, smallint, mediumint, int, tinyint unsigned, smallint unsigned, int unsigned, and bigint unsigned. • Floating-point types: float and double. • Arbitrary precision type: numeric. • Character types: char, varchar, and text. • Boolean types: true and false.	**pos** (optional): start position of the search.

  Return type: int.

  Example:

```
gaussdb=# SELECT locate( 'b','abcabc');
 locate
--------
      2
(1 row)

gaussdb=# SELECT locate( 'b','abcabc',3);
 locate
--------
      5
(1 row)
```

  NOTE

  The locate function takes effect only when **sql_compatibility** is set to **'MYSQL'**.

- lpad_s(string text, length int [, fill text])

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Fills up a string to *length* by appending the characters *fill* (a space by default). If the string is already longer than *length*, an error will be reported.

Return type: text.

NOTE

In ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s1**, this function operates as follows:

- The **length** parameter indicates the display length for character strings. The display length for individual characters is consistent with that in ORA-compatible databases.
- During the function execution, if the remaining length is 1 and the next character is of the full-width type (2 bytes), a space character is added to the left of the string.
- If the value of **length** is a decimal, the value is rounded down instead of being rounded off.
- The **string** and **fill** parameters do not comply with the encoding specifications.

In other cases:

- The **length** parameter indicates the total length of characters in a character string. The length of a single character is fixed to 1.
- If the value of **length** is a decimal, the value is rounded off.
- The **string** and **fill** parameters do not comply with the encoding specifications.

- lpad(string text, length int [, fill text])

  Description: Fills up a string to *length* by appending the characters *fill* (a space by default). If the string is already longer than *length*, it will be truncated from the end.

  Return type: text.

  NOTE

  In ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s1**, this function operates as follows:

  - The **length** parameter indicates the display length for character strings. The display length for individual characters is consistent with that in ORA-compatible databases.
  - During the function execution, if the remaining length is 1 and the next character is of the full-width type (2 bytes), a space character is added to the left of the string.
  - If the value of **length** is a decimal, the value is rounded down instead of being rounded off.
  - The **string** and **fill** parameters do not comply with the encoding specifications.

  In other cases:

  - The **length** parameter indicates the total length of characters in a character string. The length of a single character is fixed to 1.
  - If the value of **length** is a decimal, the value is rounded off.
  - The **string** and **fill** parameters do not comply with the encoding specifications.

- make_set(bits, str1, str2,...)

  Description: **str1** corresponds to bit 0 of the bits input parameter in the bit format, **str2** corresponds to bit 1, and so on. If the lower bit of the corresponding bit form is **1**, the corresponding **str** parameter is appended to the result and returned with a comma as the separator.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: See **Table 1-37**.

Table 1-37 make_set parameters

Parameter	Type	Description
bits	The input parameter **bits** supports the following types: • NULL. • Integer types: tinyint, smallint, mediumint, int, bigint, tinyint unsigned, smallint unsigned, int unsigned, and bigint unsigned. • Floating-point and fixed-point types: float, real, and double; numeric, decimal, and dec. • Bit type: bit. GaussDB does not support bits. In the code design, when the input parameter **1** is in hexadecimal format (for example, **x'FFF'**) and the length is an odd number, MySQL reports an error, but GaussDB processes the error in a different way. • Character string types: char and varchar. Only integer strings (within the range of bigint) are supported. For strings containing decimals, the returned results are different because the rounding rule is different from that of MySQL. The character string consisting of digits and letters is not supported. • Text types: tinytext, text, mediumtext, and longtext. Only numeric integer strings (within the range of bigint) are supported. If the integer range is exceeded, the returned result is different from that in MySQL. The returned results of character strings with decimal places are different from those of MySQL due to different rounding rules. The text type combined by digits and letters is not supported. • Large object types: tinyblob, blob, mediumblob, and longblob.	Specifies whether to include the subsequent **str** parameters based on each bit of the binary value corresponding to this parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
str1, str2, ...	The input parameter **str** supports the following types: • NULL. • Integer types: tinyint, smallint, mediumint, int, bigint, tinyint unsigned, smallint unsigned, int unsigned, and bigint unsigned. However, the integer value exceeds the range from 81 negative nines to 81 positive nines, and the return value is different from that of MySQL. This difference between GaussDB and MySQL is inherent and does not result from this function. • Floating-point and fixed-point types: float, real, and double for floating-point types; numeric, decimal, and dec for fixed-point types. However, when the value is expressed in scientific notation, the trailing zeros in GaussDB are displayed, and those in MySQL are not displayed. The value is printed in scientific notation. This is an inherent display difference and is not caused by this function. • Character string types: char and varchar. • Text types: tinytext, text, mediumtext, and longtext. • Large object types: tinyblob, blob, mediumblob, and longblob. • Date types: datetime, timestamp, date, and time.	Specifies whether to append some strings of **str1**, **str2**, and so forth to the return result in sequence based on the binary bits of the **bits** parameter.

Return type: text.

Example:

```
gaussdb=# SELECT make_set(7,'a','b','c','d');
 make_set
----------
 a,b,c
(1 row)
gaussdb=# SELECT
make_set(b'10101010101011010101010101101010101010110101010101011010101010101','1','2','3','4','
5','6','7','8','9','10','11','12','13','14','15','16','17','18','19','20','21','22','23','24','25','26','27','28','29','30','31','3
2','33','34','35','36','37','38','39','40','41','42','43','44','45','46','47','48','49','50','51','52','53','54','55','56','57','
58','59','60','61','62','63','64','65','66','67','68','69');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
                                                  make_set
-----------------------------------------------------------------------------------------------------
 2,4,6,8,10,12,13,15,17,19,21,23,25,26,28,30,32,34,36,38,39,41,43,45,47,49,51,52,54,56,58,60,62,64
(1 row)
```

NOTE

- The make_set function takes effect only when **sql_compatibility** is set to **'MYSQL'**.
- When the **bits** parameter is an integer, the maximum range is int128, which is smaller than the MySQL range.
- When the **bits** parameter is of the date type (datetime, timestamp, date, or time), it is not supported because the conversion from the date type to the integer type is different from that in MySQL, which is not caused by this function.
- GaussDB and MySQL are inherently different in the bit and Boolean types, causing different returned results, which is not caused by this function. When the **bits** input parameter is of the Boolean type, the **str** input parameter of the bit and Boolean types is not supported.

- notlike(bytea, bytea), notlike(name, text), and notlike(text, text)

  Description: This function is an overloaded function and is used to compare whether two input parameters are of different data types.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT notlike(1,2);
   notlike
--------------
          t
(1 row)
gaussdb=# SELECT notlike(1,1);
   notlike
--------------
          f
(1 row)
```

- octet_length(string)

  Description: Specifies the number of bytes in a string.

  Return type: int.

  Example:

```
gaussdb=# SELECT octet_length('jose');
 octet_length
--------------
          4
(1 row)
```

- overlay(string placing string FROM int [for int])

  Description: Replaces substrings. **FROM int** indicates the start position of the replacement in the first string. **for int** indicates the number of characters replaced in the first string.

  Return type: text.

  Example:

```
gaussdb=# SELECT overlay('hello' placing 'world' FROM 2 for 3 );
 overlay
---------
 hworldo
(1 row)
```

- position(substring in string)

  Description: Specifies the position of a substring. Parameters are case-sensitive.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: int. If the character string does not exist, **0** is returned.

Example:

```
gaussdb=# SELECT position('ing' in 'string');
 position
----------
        4
(1 row)
```

- pg_client_encoding()

  Description: Specifies the current client-side encoding name.

  Return type: name

  Example:

  ```
  gaussdb=# SELECT pg_client_encoding();
   pg_client_encoding
  --------------------
   UTF8
  (1 row)
  ```

- quote(str)

  Description: Adds a single quotation mark (') at both ends of the string specified by **str**, and escapes any special characters within the string, such as backslashes (\\), single quotation marks (\'), ASCII NUL (\0), and Control+Z (\Z). If the parameter is **NULL**, the return value is **NULL** without single quotation marks (').

  Parameters: See **Table 1-38**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-38 quote parameters

Parameter	Type	Description
str	• NULL. • Integer types: tinyint, smallint, mediumint, int, bigint, tinyint unsigned, smallint unsigned, int unsigned, and bigint unsigned. However, the integer value exceeds the range from 81 negative nines to 81 positive nines, and the return value is different from that of MySQL. This difference between GaussDB and MySQL is inherent and does not result from this function. • Floating-point and fixed-point types: float, real, and double for floating-point types; numeric, decimal, and dec for fixed-point types. However, when the value is expressed in scientific notation, the trailing zeros in GaussDB are displayed, and those in MySQL are not displayed. The value is printed in scientific notation. This is an inherent display difference and is not caused by this function. • Character string types: char and varchar.	Adds a single quotation mark (') at both ends of the string specified by **str**, and escapes any special characters within the string, such as backslashes (\\), single quotation marks (\'), ASCII NUL (\0), and Control+Z (\Z).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
	• Text types: tinytext, text, mediumtext, and longtext. • Large object types: tinyblob, blob, mediumblob, and longblob. • Date types: datetime, timestamp, date, and time.	

Return type: text.

Example:

```
gaussdb=# SELECT quote('hello\' world');
    quote
--------------
 'hello\' world'
(1 row)
```

NOTE

- The function takes effect only when **sql_compatibility** is set to **'MYSQL'**.
- Enable GUC parameters.
  ```
  SET standard_conforming_strings=off;
  SET escape_string_warning=off;
  SET backslash_quote=on;
  ```
- If **str** contains \Z, \r, \%, or _, GaussDB does not escape these characters, unlike MySQL-compatible databases which do escape them. In addition, differences are also observed for slashes followed by certain numbers, such as "\563". These differences are caused by the differences in escape character processing between GaussDB and MySQL, and are not caused by this function.
- The output format of \b in the **str** character string is different from that in MySQL. This difference between GaussDB and MySQL is inherent and does not result from this function.
- If **str** contains "\0", GaussDB cannot recognize it due to the limitations of the UTF-8 character set, leading to input failure. This difference between GaussDB and MySQL is inherent and does not result from this function.
- GaussDB does not support the bit or Boolean type for **str** due to its divergence from MySQL-compatible databases.
- GaussDB supports a maximum data transmission of 1 GB. The input parameter **str** can have a maximum length of 536,870,908, and the returned result string can be up to 1 GB in size.

- quote_ident(string text)

  Description: Returns the given string suitably quoted to be used as an identifier in an SQL statement string (quotation marks are used as required). Quotation marks are added only if necessary (that is, if the string contains non-identifier characters or would be case-folded). The quotation marks embedded in the return value must be written twice.

  Parameter: The input parameter is a string, which will be used as an SQL identifier.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: text.

Example:

```
gaussdb=# SELECT quote_ident('hello world');
 quote_ident
--------------
 "hello world"
(1 row)
```

- quote_literal(string text)

  Description: Returns the given string suitably quoted to be used as text in an SQL statement string (quotation marks are used as required). It supports XML data that is explicitly converted to the character type.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT quote_literal('hello');
   quote_literal
  ---------------
   'hello'
  (1 row)
  ```

  If a command similar to the following exists, the text will be escaped.

  ```
  gaussdb=# SELECT quote_literal(E'O\'hello');
   quote_literal
  ---------------
   'O''hello'
  (1 row)
  ```

  If a command similar to the following exists, the backslash will be properly doubled.

  ```
  gaussdb=# SELECT quote_literal('O\hello');
   quote_literal
  ---------------
   E'O\\hello'
  (1 row)
  ```

  If the parameter is **NULL**, **NULL** is returned. If the parameter may be null, you are advised to use the quote_nullable function.

  ```
  gaussdb=# SELECT quote_literal(NULL);
   quote_literal
  ---------------

  (1 row)
  ```

- quote_literal(value anyelement)

  Description: Converts the given value to text and then quotes it as a literal.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT quote_literal(42.5);
   quote_literal
  ---------------
   '42.5'
  (1 row)
  ```

  If a command similar to the following exists, the given value will be escaped.

  ```
  gaussdb=# SELECT quote_literal(E'O\'42.5');
   quote_literal
  ---------------
   '0''42.5'
  (1 row)
  ```

  If a command similar to the following exists, the backslash will be properly doubled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT quote_literal('O\42.5');
 quote_literal
---------------
 E'O\\42.5'
(1 row)
```

- quote_nullable(string text)

  Description: Returns the given string suitably quoted to be used as a string in an SQL statement string (quotation marks are used as required). It supports XML data that is explicitly converted to the character type.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT quote_nullable('hello');
   quote_nullable
  ----------------
   'hello'
  (1 row)
  ```

  If a command similar to the following exists, the text will be escaped.

  ```
  gaussdb=# SELECT quote_nullable(E'O\'hello');
   quote_nullable
  ----------------
   'O''hello'
  (1 row)
  ```

  If a command similar to the following exists, the backslash will be properly doubled.

  ```
  gaussdb=# SELECT quote_nullable('O\hello');
   quote_nullable
  ----------------
   E'O\\hello'
  (1 row)
  ```

  If the parameter is **NULL**, **NULL** is returned.

  ```
  gaussdb=# SELECT quote_nullable(NULL);
   quote_nullable
  ----------------
   NULL
  (1 row)
  ```

- quote_nullable(value anyelement)

  Description: Converts the given value to text and then quotes it as a literal. The input parameter type is anyelement.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT quote_nullable(42.5);
   quote_nullable
  ----------------
   '42.5'
  (1 row)
  ```

  If a command similar to the following exists, the given value will be escaped.

  ```
  gaussdb=# SELECT quote_nullable(E'O\'42.5');
   quote_nullable
  ----------------
   'O''42.5'
  (1 row)
  ```

  If a command similar to the following exists, the backslash will be properly doubled.

  ```
  gaussdb=# SELECT quote_nullable('O\42.5');
   quote_nullable
  ----------------
  ```

```
 E'O\\42.5'
(1 row)
```

If the parameter is **NULL**, **NULL** is returned.

```
gaussdb=# SELECT quote_nullable(NULL);
 quote_nullable
----------------
 NULL
(1 row)
```

- space(count)

  Description: Returns a string consisting of a specified number of spaces.

  Parameters: See **Table 1-39**.

  Table 1-39 space parameters

Parameter	Type	Description
count	• Integer types: tinyint, smallint, mediumint, int, and bigint. • Unsigned integer types: tinyint unsigned, smallint unsigned, and int unsigned. • Character and text types: char, varchar, tinytext, text, mediumtext, and longtext. Only integer strings are supported, and the integer range is within the bigint range. • Floating-point types: float, real, and double. • Fixed-point types: numeric, decimal, and dec. • Boolean type.	Number of spaces.

  Return type: text.

  Example:

```
gaussdb=# SELECT space(5);
 space
--------

(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The space function takes effect only when **sql_compatibility** is set to **'MYSQL'**.
- If the input parameter **count** is less than **1073741819** and greater than **0**, a string of *count* spaces is returned. Otherwise, an empty string is returned.
- If the input parameter is of the floating-point or fixed-point type and the parameter value is not an integer, the return value is rounded down.
- If the input parameter is of the Boolean type, the value **true** is converted to **1** and the value **false** is converted to **0**.

- substring_inner(string [from int] [for int])

  Description: Extracts a substring. **from int** indicates the start position of the truncation. **for int** indicates the number of bytes truncated.

  Return type: text.

  Example:

```
gaussdb=# SELECT substring_inner('adcde', 2,3);
 substring_inner
-----------------
 dcd
(1 row)
```

- substring(string [from int] [for int])

  Description: Extracts a substring. **from int** indicates the start position of the truncation. **for int** indicates the number of bytes truncated.

  Return type: text.

  Example:

```
gaussdb=# SELECT substring('Thomas' from 2 for 3);
 substring
-----------
 hom
(1 row)
```

  NOTE

  This function is designed for MySQL-compatible databases. When the GUC parameters **b_format_version** is **'5.7'** and **b_format_dev_version** is **'s1'**, **[from int]** can be a negative number, indicating the character position counted backwards from the end of the string. If these parameters are not set, the result will be empty when **[from int]** is negative.

- substring(string, pos, len)

  Description: Extracts substring.

  Parameters: See **Table 1-40**.

Table 1-40 substring(string, pos, len) parameters

Parameter	Type	Description	Value Range
string	text	Character string to be truncated.	-
pos	Int	Start position of the character string to be truncated.	The absolute value is less than the string length.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description	Value Range
len	Int	Length of the character string to be truncated.	The value is greater than 0.

Return type: text.

Example:

```
gaussdb=# SELECT substring('substrteststring', -5, 5);
 substring
-----------
 tring
(1 row)
```

NOTE

This function is designed for MySQL-compatible databases. When the GUC parameters **b_format_version** is **'5.7'** and **b_format_dev_version** is **'s1'**, **pos** can be a negative number, indicating the character position counted backwards from the end of the string. If these parameters are not set, the result will be empty when **pos** is negative.

- substring_index(str, split, index)

  Description: Returns the left or right part of the *n*th (specified by *index*) *str* string separated by *split*. If *index* is a positive number, the left part is returned. If *index* is a negative number, the right part is returned.

  Parameters: See **Table 1-41**.

  Table 1-41 substring_index parameters

Parameter	Type	Description
str	text	Source string to be processed.
split	text	Separator to be used.
index	Int	Position index of the separator (starting from 1). A positive number indicates the left side, and a negative number indicates the right side.

  Return type: text.

  Example:

```
gaussdb=# SELECT substring_index('Test1splitTest2splitTest3splitTest4', 'split', 2);
 substring_index
-----------------
 Test1splitTest2
(1 row)

gaussdb=# SELECT substring_index('Test1splitTest2splitTest3splitTest4', 'split', -2);
 substring_index
-----------------
 Test3splitTest4
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- substring(string FROM pattern)

  Description: Extracts substrings matching the POSIX regular expression. It returns null if there is no match, otherwise the portion of the text that matched the pattern.

  Return type: text.

  Example:

```
gaussdb=# SELECT substring('Thomas' FROM '...$');
 substring
-----------
 mas
(1 row)
gaussdb=# SELECT substring('foobar' FROM 'o(.)b');
 result
--------
 o
(1 row)
gaussdb=# SELECT substring('foobar' FROM '(o(.)b)');
 result
--------
 oob
(1 row)
```

  NOTE

  - If the POSIX regular expression contains any parentheses, the portion of the text that matched the first parenthesized sub-expression (the one whose left parenthesis comes first) is returned. If you want to use parentheses without triggering this behavior, you can put a pair of additional parentheses around the whole expression.
  - When **sql_compatibility** is set to **'MYSQL'**, and the GUC parameters **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s1'**, this function is used in the same way as **substr(str FROM pos)**. The output result and behavior of this function are the same as those of the **substr(str FROM pos)** function.

- substring(string from pattern for escape)

  Description: Extracts substrings matching the SQL regular expression. The declared schema must match the entire data string; otherwise, the function fails and returns a null value. To indicate the part of the pattern that should be returned on success, the pattern must contain two occurrences of the escape character followed by a double quotation mark ("). The text matching the portion of the pattern between these marks is returned.

  Return type: text.

  Example:

```
gaussdb=# SELECT substring('Thomas' from '%#"o_a#"_' for '#');
 substring
-----------
 oma
(1 row)
```

  NOTE

  When **sql_compatibility** is **'MYSQL'**, and the GUC parameters **b_format_version** is **'5.7'** and **b_format_dev_version** is **'s1'**, this function operates similarly to **substr(str FROM pos FOR len)**. Its output and behavior are identical to those of **substr(str FROM pos FOR len)**.

- rawcat(raw, raw)

  Description: Indicates the string concatenation function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: raw

Example:

```
gaussdb=# SELECT rawcat('ab','cd');
 rawcat
--------
 ABCD
(1 row)
```

- regexp_substr(string text, pattern text [, position int [, occurrence int [, flags text]]])

  Description: Extracts substrings from a regular expression. Its function is similar to **substr**. When a regular expression contains multiple parallel brackets, it also needs to be processed.

  Parameters:

  - **string**: source character string used for matching.
  - **pattern**: regular expression pattern string used for matching.
  - **position**: start character of the source string used for matching. This parameter is optional. The default value is **1**.
  - **occurrence**: sequence number of the matched substring to be extracted. This parameter is optional. The default value is **1**.
  - **flags**: contains zero or multiple single-letter flags that change the matching behavior of the function. This parameter is optional. **Table 1-42** describes the options supported by **flags** and their meanings.

    Table 1-42 Options supported by flags

Option	Description
'b'	Indicates the BRE matching without extension.
'c'	Indicates the case-sensitive matching.
'e'	Indicates the ERE matching with extension.
'i'	Indicates the case-insensitive matching.
'm'	Indicates the multi-line matching. If **flags** contains **'m'**, use the multi-line matching. Otherwise, use the single-line matching.
'n'	The meaning of **'n'** varies depending on the GUC parameter **behavior_compat_options** and the compatibility mode of the current database. When **sql_compatibility** is set to **'ORA'** or **'MYSQL'** and the GUC parameter **behavior_compat_options** includes **aformat_regexp_match**, the **'n'** option means that "." matches with a newline character (**'\n'**). If **'n'** is not specified in **flags**, "." will not match with a newline character. In other cases, the **'n'** option has the same meaning as the **'m'** option.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Option	Description
'p'	Indicates partial linefeed-sensitive matching, which is similar to the linefeed-sensitive matching (**'m'** or **'n'**) and affects "." and square bracket expression, but does not affect ^ and $.
'q'	Indicates common character matching.
's'	Indicates the single-line matching. The meaning is opposite to those of **'m'** and **'n'**.
't'	Indicates the compact matching. The whitespace characters match themselves.
'w'	Indicates the inverse partial newline-sensitive matching, opposite to **'p'**.
'x'	Indicates the loose matching. The whitespace characters are ignored.

Return type: text.

Example:

```
gaussdb=# SELECT regexp_substr('str','[ac]');
 regexp_substr
---------------


(1 row)

gaussdb=# SELECT regexp_substr('foobarbaz', 'b(..)', 3, 2) AS RESULT;
 result
--------
 baz
(1 row)
```

- regexp_count(string text, pattern text [, position int [, flags text]])

  Description: obtains the number of substrings used for matching.

  Parameters:

  - **string**: source character string used for matching.
  - **pattern**: regular expression pattern string used for matching.
  - **position**: start character of the source string used for matching. This parameter is optional. The default value is **1**.
  - **flags**: contains zero or multiple single-letter flags that change the matching behavior of the function. This parameter is optional. **Table 1-42** describes the options supported by **flags** and their meanings.

  NOTE

  When the function is in an ORA-compatible database, the value of **a_format_version** is **10c**, and the value of **a_format_dev_version** is **s1**, the **pattern** parameter ending with a backslash (\) is valid.

  Return type: int.

  Example:

  ```
  gaussdb=# SELECT regexp_count('foobarbaz','b(..)', 5) AS RESULT;
  result
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------
1
(1 row)
```

- regexp_instr(string text, pattern text [, position int [, occurrence int [, return_opt int [, flags text]]]])

  Description: obtains the position (starting from 1) of the substring that meets the matching condition. If no substring is matched, **0** is returned.

  Parameters:

  - **string**: source character string used for matching.
  - **pattern**: regular expression pattern string used for matching.
  - **position**: start character of the source string used for matching. This parameter is optional. The default value is **1**.
  - **occurrence**: sequence number of the matched substring to be obtained. This parameter is optional. The default value is **1**.
  - **return_opt**: specifies whether to return the position of the first or last character of the matched substring. This parameter is optional. If the value is **0**, the position of the first character (starting from 1) of the matched substring is returned. If the value is greater than 0, the position of the next character of the end character of the matched substring is returned. The default value is **0**.
  - **flags**: contains zero or multiple single-letter flags that change the matching behavior of the function. This parameter is optional. **Table 1-42** describes the options supported by **flags** and their meanings.

  Return type: int.

  Example:

```
gaussdb=# SELECT regexp_instr('foobarbaz','b(..)', 1, 1, 0) AS RESULT;
result
--------
4
(1 row)

gaussdb=# SELECT regexp_instr('foobarbaz','b(..)', 1, 2, 0) AS RESULT;
result
--------
7
(1 row)
```

- regexp_matches(string text, pattern text [, flags text])

  Description: Returns all captured substrings resulting from matching a POSIX regular expression against **string**. If there is no match to the pattern, the function returns no rows.

  - If the pattern contains no parenthesized sub-expressions, then each row returned is a single-element text array containing the substring matching the whole pattern.
  - If the pattern contains parenthesized sub-expressions, the function returns a text array whose *n*th element is the substring matching the *n*th parenthesized sub-expression of the pattern.

  Parameter: The optional parameter **flags** contains zero or multiple single-letter flags that change the function's behavior. Flag **i** specifies case-insensitive matching, while flag **g** specifies replacement of each matching substring rather than only the first one.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

If the last parameter is provided but the parameter value is an empty string (''') and the SQL compatibility mode is set to **ORA**, the returned result is an empty set. This is because the ORA compatible mode treats the empty string (''') as **NULL**. To resolve this problem, you can:

- Change the database SQL compatibility mode to **TD**.
- Do not provide the last parameter or do not set the last parameter to an empty string.

Return type: SETOF text[]

Example:

```
gaussdb=# SELECT regexp_matches('foobarbequebaz', '(bar)(beque)');
 regexp_matches
----------------
 {bar,beque}
(1 row)
gaussdb=# SELECT regexp_matches('foobarbequebaz', 'barbeque');
 regexp_matches
----------------
 {barbeque}
(1 row)
gaussdb=# SELECT regexp_matches('foobarbequebazilbarfbonk', '(b[^b]+)(b[^b]+)', 'g');
 regexp_matches
----------------
 {bar,beque}
 {bazil,barf}
(2 rows)
```

- regexp_match(string text, pattern text [, flags text])

  Description: Returns a string array of matching substrings within the first match of a POSIX regular expression pattern. If there is no match to the pattern, the function returns an empty row.

  – If the pattern contains no parenthesized sub-expressions, then each row returned is a single-element text array containing the substring matching the whole pattern.

  – If the pattern contains parenthesized sub-expressions, the function returns a text array whose *n*th element is the substring matching the *n*th parenthesized sub-expression of the pattern.

  Parameter: The **flags** parameter is optional. If **flags** is not specified, tight syntax matching is used by default. The **flags** parameter contains zero or multiple single-letter flags that change the behavior of a function. The following table lists the supported flags.

Table 1-43 flags parameters

Option	Description
'b'	Converts the POSIX regular expression to a BRE.
'c'	Indicates the case-sensitive matching.
'e'	Converts the POSIX regular expression to an ERE.
'i'	Indicates the case-insensitive matching.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Option	Description
'n'	Indicates newline-sensitive matching. A character string is regarded as multiple lines. Caret (^) and dollar sign ($) match the beginning and end of each line. Periods (.) do not match linefeed.
'm'	Indicates newline-sensitive matching. A character string is regarded as multiple lines. Caret (^) and dollar sign ($) match the beginning and end of each line. Periods (.) do not match linefeed. The matching mode of *n* is the same.
'p'	Indicates partial newline-sensitive matching. Periods (.) do not match linefeed. Caret (^) can only match the beginning of the first line and dollar sign ($) can only match the end of the last line.
'q'	Represents the regular expression is a text string, all of which are common characters.
's'	Indicates non-newline-sensitive matching. That is, a character string is regarded as a single line, and caret (^) and dollar sign ($) match the beginning and end of the line, respectively. In this mode, periods (.) in the regular expression can match any character, including the linefeed character.
't'	Indicates the tight syntax matching.
'w'	Indicates inverse partial newline-sensitive matching. In the regular expression, caret (^) matches the beginning of each line, and dollar sign ($) matches only the end of the last line. Period (.) can match any character, including the linefeed character.
'x'	Indicates the extended syntax matching, ignoring whitespace characters and comments in regular expressions. The following three scenarios are excluded: • The whitespace characters or comments starting with the number sign (#) that are prefixed with slashes (\) will be retained. • Whitespace characters or number signs (#) in square brackets will be retained. • The whitespace and comment characters such as (? : are not allowed in multi-character symbols.

NOTICE

If the parameter contains an empty string ('') and the compatibility mode of the database is set to **ORA**, the returned result is **NULL**. This is because the empty string is processed as **NULL** in ORA-compatible mode.

Return type: SETOF text[]

Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT regexp_match('foobarbequebaz', '(bar)(beque)');
 regexp_match
--------------
 {bar,beque}
(1 row)
gaussdb=# SELECT (regexp_match('foobarbequebaz', 'bar.*que'))[1];
 regexp_match
--------------
 barbeque
(1 row)
gaussdb=# SELECT regexp_match('Learning #PostgreSQL', 'R', 'c');
 regexp_match
--------------

(1 row)
gaussdb=# SELECT regexp_match('hello world', 'h e l l o', 'x');
 regexp_match
--------------
 {hello}
(1 row)
```

- regexp_split_to_table(string text, pattern text [, flags text])

  Description: Splits **string** using a POSIX regular expression as the delimiter. If there is no match to the pattern, the function returns the string. When there is at least one match, each match returns the text between the end position of the previous match (or the start position of the string) and the start position of the current match. When there are no more matches, it returns the text from the end of the last match to the end of the string.

  Parameter: The **flags** parameter contains zero or multiple single-letter flags that change the function's behavior. Flag **i** specifies case-insensitive matching. By default, all matching substrings are replaced, not just the first one.

  Return type: SETOF text

  Example:

  ```
  gaussdb=# SELECT regexp_split_to_table('hello world', E'\\s+');
   regexp_split_to_table
  -----------------------
   hello
   world
  (2 rows)
  ```

- regexp_split_to_array(string text, pattern text [, flags text])

  Description: Splits **string** using a POSIX regular expression as the delimiter. This function is similar to the regexp_split_to_table function. However, this function returns the split result as a text array.

  Return type: text[]

  Example:

  ```
  gaussdb=# SELECT regexp_split_to_array('hello world', E'\\s+');
   regexp_split_to_array
  -----------------------
   {hello,world}
  (1 row)
  ```

- repeat(string text, number int)

  Description: Repeats **string** the specified number of times.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT repeat('Pg', 4);
    repeat
  ----------
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 PgPgPgPg
(1 row)
```

NOTE

The maximum size of memory allocated at a time cannot exceed 1 GB due to the memory allocation mechanism of the database. Therefore, the maximum value of **number** cannot exceed (1 GB – **x**)/**lengthb** (**string**) – 1. **x** indicates the length of the header information, which is usually greater than 4 bytes. The value varies among different scenarios.

- replace(string text, from text, to text)

  Description: Replaces all occurrences in **string** of substring specified by **from** with substring specified by **to**.

  Return type: text.

  Example:

```
gaussdb=# SELECT replace('abcdefabcdef', 'cd', 'XXX');
    replace
----------------
 abXXXefabXXXef
(1 row)
```

- replace(string, substring)

  Description: Deletes all substrings in a string.

  Parameters: **string** and **substring**, both are of the text type.

  Return type: text.

  Example:

```
gaussdb=# SELECT replace('abcdefabcdef', 'cd');
    replace
----------------
 abefabef
(1 row)
```

- reverse(str)

  Description: Returns a reversed string (by character).

  Return type: text.

  Example:

```
gaussdb=# SELECT reverse('abcde');
 reverse
---------
 edcba
(1 row)
```

- right(str text, n int)

  Description: Returns the last *n* characters in a string. When *n* is negative, all but the first */n/* characters are returned.

  Return type: text.

  Example:

```
gaussdb=# SELECT right('abcde', 2);
 right
-------
 de
(1 row)

gaussdb=# SELECT right('abcde', -2);
 right
-------
 cde
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- rpad(string text, length int [, fill text])

  Description: Pads a string to a specified length by filling the character specified by **fill** (blank by default). If the string is already longer than *length*, it will be truncated from the end.

  Return type: text.

  NOTE

  In ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s1**, this function operates as follows:

  - The **length** parameter indicates the display length for character strings. The display length for individual characters is consistent with that in the Oracle Database.
  - During the function execution, if the remaining length is 1 and the next character is of the full-width type (2 bytes), a space character is added to the right of the string.
  - If the value of *length* is a decimal, the value is truncated instead of being rounded off.
  - The **string** and **fill** parameters do not comply with the encoding specifications.
  - If the length of the **string** parameter exceeds 1 GB after padding, an error is reported.

  In other cases:

  - The **length** parameter indicates the total length of characters in a character string. The length of a single character is fixed to 1.
  - If the length of the **string** parameter exceeds 1 GB after padding, an error is reported.
  - If the value of **length** is a decimal, the value is rounded off.
  - The **string** and **fill** parameters do not comply with the encoding specifications.

- substrb(text, int, int)

  Description: Extracts substring. The first **int** parameter indicates the start position of the extraction. It can be a positive or negative number. A positive number indicates that the extraction starts from the beginning of the string, and a negative number indicates that the extraction starts from the end of the string. The second **int** parameter indicates the number of characters to be extracted.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT substrb('string',2,3);
   substrb
  ---------
   tri
  (1 row)
  ```

- substrb(text, int)

  Description: Extracts a substring. **int** indicates the start position of the extraction.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT substrb('string',2);
   substrb
  ---------
   tring
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- substr(text, int)/substr(str FROM pos)

  Description: **str** is the target string, and **pos** is the position in the string. The string of **str** from the **pos** position is output to the end of the character.

  Return type: text.

  Example:

```
gaussdb=# SELECT substr('stringtest' FROM 4);
 substr
---------
 ingtest
(1 row)

gaussdb=# SELECT substr('stringtest', 4);
 substr
---------
 ingtest
(1 row)
```

- substr(str FROM pos for len)

  Description: Extracts a substring. The first **int** indicates the start position of the subtraction. The second **int** indicates the number of characters extracted. The value of **pos** can be a negative number. If the value is a negative number, the value is extracted from back to front.

  Return type: text.

  Example:

```
gaussdb=# SELECT substr('teststring' FROM 5 for 2);
 substr
--------
 st
(1 row)
```

- substr(bytea, from, count)

  Description: Extracts a substring from **bytea**. **from** specifies the position where the extraction starts. **count** specifies the length of the extracted substring.

  Return type: text.

  Example:

```
gaussdb=# SELECT substr('string',2,3);
 substr
--------
 tri
(1 row)
```

- string || string

  Description: Concatenates strings.

  Return type: text.

  Example:

```
gaussdb=# SELECT 'MPP'||'DB' AS RESULT;
 result
--------
 MPPDB
(1 row)
```

- string || non-string or non-string || string

  Description: Concatenates strings and non-strings.

  Return type: text.

  Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT 'Value: '||42 AS RESULT;
  result
-----------
 Value: 42
(1 row)
```

- split_part(string text, delimiter text, field int)

  Description: Splits *string* based on *delimiter* and returns the *field*th substring (counting from text of the first appeared delimiter).

  Return type: text.

  Example:

```
gaussdb=# SELECT split_part('abc~@~def~@~ghi', '~@~', 2);
 split_part
------------
 def
(1 row)
```

- strpos(string, substring)

  Description: Specifies the position of a substring. This function is the same as position(substring in string) except that it reverses the parameter order.

  Return type: int.

  Example:

```
gaussdb=# SELECT strpos('source', 'rc');
 strpos
--------
      4
(1 row)
```

- to_hex(number int or bigint)

  Description: Converts a number to a hexadecimal expression.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_hex(2147483647);
  to_hex
----------
 7fffffff
(1 row)
```

- translate(string text, from text, to text)

  Description: Replaces any characters in string that match characters of *from* with the corresponding characters of *to*. If the length of **from** is greater than that of **to**, the characters in **string** that match the characters in **from** (excluding the characters in **to**) will be deleted.

  Return type: text.

  Example:

```
gaussdb=# SELECT translate('12345', '143', 'ax');
 translate
-----------
 a2x5
(1 row)
```

- length(string)

  Description: Obtains the number of characters in a string. When **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, if the string is of the string or text type, the number of bytes of the string is returned.

  Return type: integer

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- lengthb(string)

  Description: Obtains the number of bytes in a string. The value depends on character sets. For instance, the number of returned bytes varies between GBK and UTF-8.

  Return type: integer

  Example:

  ```
  gaussdb=# SELECT lengthb('ABCDEF');
   lengthb
  ---------
         7
  (1 row)
  ```

- substr(string, from)

  Description: Extracts substrings from **string**. **from** indicates the start position of the extraction.

  - If **from** starts at 0, **1** will be used.
  - If the value of *from* is positive, all characters from *from* to the end are extracted.
  - If the value of *from* is negative, the last *n* characters in the string are extracted, and *n* indicates the absolute value of *from*.

  Return type: text.

  Example:

  - If the value of **from** is positive:

    ```
    gaussdb=# SELECT substr('ABCDEF',2);
     substr
    --------
     BCDEF
    (1 row)
    ```

  - If the value of **from** is negative:

    ```
    gaussdb=# SELECT substr('ABCDEF',-2);
     substr
    --------
     EF
    (1 row)
    ```

- substr(string, from, count)

  Description: Extracts substrings from **string**. **count** indicates the length of the extracted substring, and **from** indicates the start position of the extraction.

  - If **from** starts at 0, **1** will be used.
  - If *from* is a positive number, *count* characters will be extracted, starting from *from*.
  - If *from* is a negative number, *count* characters will be extracted, starting from the *n*th character from the end, where *n* is the absolute value of *from*.
  - If **count** is smaller than 1, **null** will be returned.

  Return type: text.

  Example:

  - If the value of **from** is positive:

    ```
    gaussdb=# SELECT substr('ABCDEF',2,2);
     substr
    --------
     BC
    (1 row)
    ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - If the value of **from** is negative:

```
gaussdb=# SELECT substr('ABCDEF',-3,2);
 substr
--------
 DE
(1 row)
```

- substrb(string, from)

  Description: Same as the functionality of the substr(string,from) function, but the calculation unit is byte.

  Return type: text.

  Example:

```
gaussdb=# SELECT substrb('ABCDEF',-2);
 substrb
---------
 EF
(1 row)
```

- substrb(string, from, count)

  Description: Same as the functionality of the substr(string, from, count) function, but the calculation unit is byte.

  Return type: text.

  Example:

```
gaussdb=# SELECT substrb('ABCDEF',2,2);
 substrb
---------
 BC
(1 row)
```

- to_single_byte(char)

  Description: Converts all multi-byte characters in a string to single-byte characters.

  Return type: text.

  NOTE

  This function takes effect in ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- to_multi_byte(char)

  Description: Converts all single-byte characters in a string to multi-byte characters.

  Return type: text.

  NOTE

  This function takes effect in ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- trim([leading |trailing |both] [characters] FROM string)

  Description: Removes the longest string consisting only of characters in **characters** (a space by default) from the start, end, or both sides of **string**.

  Return type: text.

  Example:

```
gaussdb=# SELECT trim(BOTH 'x' FROM 'xTomxx');
 btrim
-------
 Tom
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT trim(LEADING 'x' FROM 'xTomxx');
 ltrim
-------
 Tomxx
(1 row)
gaussdb=# SELECT trim(TRAILING 'x' FROM 'xTomxx');
 rtrim
-------
 xTom
(1 row)
```

- rtrim(string [, characters])

  Description: Removes the longest string consisting only of characters in **characters** (a space by default) from the end of **string**.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT rtrim('TRIMxxxx','x');
   rtrim
  -------
   TRIM
  (1 row)
  ```

- ltrim(string [, characters])

  Description: Removes the longest string consisting only of characters in **characters** (a space by default) from the start of **string**.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT ltrim('xxxxTRIM','x');
   ltrim
  -------
   TRIM
  (1 row)
  ```

- upper(string)

  Description: Converts a character string to uppercase.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT upper('tom');
   upper
  -------
   TOM
  (1 row)
  ```

- lower(string)

  Description: Converts a character string to lowercase.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT lower('TOM');
   lower
  -------
   tom
  (1 row)
  ```

- nls_upper(string [, nlsparam])

  Description: Converts a character string to uppercase. You can specify a sorting rule to process special uppercase conversions in some languages. The format of **nlsparam** is **'nls_sort=sort_name'**, where **sort_name** is replaced by the specific sorting rule name. When the **nlsparam** parameter is not set, this function is equivalent to **upper**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: text.

Example:

```
gaussdb=# SELECT nls_upper('große');
 nls_upper
-----------
 GROßE
(1 row)
gaussdb=# SELECT nls_upper('große', 'nls_sort = XGerman');
 nls_upper
-----------
 GROSSE
(1 row)
```

NOTE

This function can be used when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s2** in an ORA-compatible database.

- nls_lower(string [, nlsparam])

  Description: Converts a character string to lowercase letters. You can specify a sorting rule to process special lowercase conversion rules in some languages. The format of **nlsparam** is **'nls_sort=sort_name'**, where **sort_name** is replaced by the specific sorting rule name. When the **nlsparam** parameter is not set, this function is equivalent to **lower**.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT nls_lower('INDIVISIBILITY');
    nls_lower
  ----------------
   indivisibility
  (1 row)
  gaussdb=# SELECT nls_lower('INDIVISIBILITY', 'nls_sort = XTurkish');
    nls_lower
  ----------------
   ındıvısıbılıty
  (1 row)
  ```

  NOTE

  This function can be used when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s2** in an ORA-compatible database.

- instr(string, substring[,position, occurrence])

  Description: Queries and returns the value of the substring position that occurs the *occurrence* (**1** by default) times from the *position* (**1** by default) in the string.

  – When **position** is **0**, the function will return **0**.
  – When **position** is a negative number, the search starts backwards from the *n*th character counting from the end of the string. *n* represents the absolute value of **position**.

  In this function, the calculation unit is character. One Chinese character is one character.

  Return type: integer

  Example:

  ```
  gaussdb=# SELECT instr('corporate floor','or', 3);
   instr
  -------
       5
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT instr('corporate floor','or',-3,2);
 instr
-------
     2
(1 row)
```

- initcap(string)

  Description: Capitalizes the first letter of each word in a string.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT initcap('hi THOMAS');
    initcap
  -----------
   Hi Thomas
  (1 row)
  ```

  NOTE

  When **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s2**, if a case-insensitive character, such as Chinese, is followed by a case-sensitive character, only the first letter of the case-sensitive character is capitalized. Therefore, you are advised to set **a_format_version** to **10c** and **a_format_dev_version** to **s2**.

- ascii(string)

  Description: Indicates the ASCII code of the first character in the string.

  Return type: integer

  Example:

  ```
  gaussdb=# SELECT ascii('xyz');
   ascii
  -------
    120
  (1 row)
  ```

- ascii2(string)

  Description: Returns the decimal code of the first character of the input string in the database character set.

  Return type: bigint

  NOTE

  This function takes effect in ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- asciistr(string)

  Description: Converts non-ASCII characters in the input string to *XXXX*, where *XXXX* indicates the UTF-16 code unit.

  Return type: text.

  NOTE

  This function takes effect in ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- unistr(string)

  Description: Converts the coding sequence in a string into the corresponding character. Other characters remain unchanged.

  Return type: text.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This function takes effect in ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s2**.
- The backslash (\) must be followed by four hexadecimal characters to indicate the coding sequence, or another backslash (\) indicates that a single backslash (\) is entered.
- If the input parameter is of the time type, the time type is implicitly converted to the character string type.

- vsize(expr)

  Description: Returns the number of bytes of the input expression.

  Return type: int.

  NOTE

  This function takes effect in ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- replace(string varchar, search_string varchar, replacement_string varchar)

  Description: Replaces all substrings specified by **search_string** in the string with substrings specified by **replacement_string**.

  Return type: text.

  Example:

```
gaussdb=# SELECT replace('jack and jue','j','bl');
    replace
----------------
 black and blue
(1 row)
```

- concat(str1, str2)

  Description: Connects str1 and str2 and returns the string. This function calls the output function of the data type and the return value is uncertain. As a result, the optimizer cannot calculate the result in advance when generating a plan. If there are performance requirements, you are advised to use the operator ||.

  NOTE

  - If **sql_compatibility** is set to **'MYSQL'** and **str1** or **str2** is set to **NULL**, the returned result is **NULL**.
  - The return value of this function is of the variable-length type. When this function is compared with table data, the character string length is lost in the combination result. As a result, the comparison results are different.

  Return type: text.

  Example:

```
gaussdb=# SELECT concat('Hello', ' World!');
    concat
--------------
 Hello World!
(1 row)
gaussdb=# SELECT concat('Hello', NULL);
 concat
--------

(1 row)
gaussdb=# CREATE TABLE test_space(c char(10));
CREATE TABLE
gaussdb=# INSERT INTO test_space values('a');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
INSERT 0 1
-- After spaces are padded, the character string is still a fixed-length character string. It is expected that the result can be found.
gaussdb=# SELECT * FROM test_space WHERE c = 'a ';
    c
------------
 a
(1 row)
-- The combination result is a variable-length character string. The comparison fails and the result cannot be found.
gaussdb=# SELECT * FROM test_space WHERE c = 'a' || ' ';
 c
---
(0 rows)
```

- chr(integer)

  Description: For the UTF-8 character set, the input is encoded as Unicode and a UTF-8 character is returned. For other character sets, an ASCII character is returned.

  Return type: text.

  NOTE

  - When **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1** in an ORA-compatible database, the value is truncated instead of being rounded off if the value of *integer* is a decimal.
  - For details about how to set and modify the character set and collation, see **Character Sets and Collations**.

  Example:

```
gaussdb=# SELECT chr(65);
 chr
-----
 A
(1 row)

-- In the case of UTF-8 character set
gaussdb=# SELECT chr(19968);
 chr
-----
 一
(1 row)
```

- chr(cvalue int|bigint)

  Description: Converts **cvalue** into characters with the specified byte sequence.

  Parameter: **cvalue** can be converted to the int or bigint type within the range of [0,2^32 – 1], which matches the range of unsigned int. Depending on the value of **n**, a character array consisting of one to four bytes will be returned. The byte arrays returned in different character sets are the same. However, due to different encoding rules, the result of the returned character string varies depending on the character set encoding. If the character set is single-byte encoded, the corresponding ASCII character is returned after the mod 256 operation is performed on **cvalue**.

  – If a byte in the input *cvalue* is **0**, the output is truncated.
  – If the input does not comply with the encoding rule of the current character set, an error is reported.
  – If the input is **NULL** or **0**, **NULL** is returned.

  Return type: text.

  Example:

```
gaussdb=# SELECT chr(65);
 chr
-----
 A
(1 row)
gaussdb=# CREATE DATABASE gaussdb_o WITH DBCOMPATIBILITY 'ORA';
gaussdb=# \c gaussdb_o
gaussdb_o=# SET a_format_version='10c';
gaussdb_o=# SET a_format_dev_version = 's1';
gaussdb_o=# SELECT chr(16705);
 chr
-----
 AA
(1 row)

-- The output is truncated.
gaussdb_o=# SELECT chr(4259905);
 chr
-----
 A
(1 row)
gaussdb_o=# \c postgres
gaussdb=#  DROP DATABASE gaussdb_o;
```

NOTE

When **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1** in an ORA-compatible database, the function returns a collation based on the input value. If the current character set for database encoding is a multi-byte character set, the return value contains one to four bytes. If the current character set for database encoding is a single-byte character set, the return value is a single byte obtained by performing the mod 256 operation on the input value. Otherwise, if the current character set for database encoding is UTF-8, the input is encoded as Unicode and a UTF-8 character is returned. For other character sets, an ASCII character is returned.

- nchr(cvalue int|bigint)

  Description: Returns characters (from the national character set) that correspond to the input parameter. The GUC parameter **nls_nchar_characterset** specifies the national character set. Only **AL16UTF16** and **UTF8** are supported.

  – If **nls_nchar_characterset** is set to **AL16UTF16** and the input parameter exceeds two bytes, the input parameter will be truncated, retaining only the lowest two bytes.
  – If **nls_nchar_characterset** is set to **UTF8** and the input parameter exceeds three bytes, the input parameter will be interpreted as **0**.

  This function takes effect in ORA-compatible databases where the GUC parameters **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s4**.

  Parameter: **cvalue** can be converted to the int or bigint type within the range of [0,2^32 – 1], which matches the range of unsigned int. If the input parameter contains the decimal part, the decimal part will be deleted.

  Return type: NVARCHAR2

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The byte length of the return value of the function is different from that in the Oracle Database.
- The return value of the function is restricted by the database character set. If there is no mapping table for converting UTF-8 to the database character set or the mapping table does not contain the UTF-8 code corresponding to the input parameter, the current database character set does not support the UTF-8 character. In this case, the return value of the function is different from that in the Oracle Database.
- If the current database character set does not support the UTF-8 character corresponding to the input parameter, or the national character set is UTF-8 but the input parameter does not comply with the UTF-8 format, the byte array corresponding to the input parameter is returned. If a single byte is in the range of [0x00,0x7F], an ASCII character is returned. If a single byte is in the range of [0x80,0xFF], a question mark (?) is returned.
- For details about how to set and modify the character set and collation, see **Character Sets and Collations**.

- regexp_substr(source_char, pattern)

  Description: Extracts substrings from a regular expression. If the SQL syntax is ORA- and MySQL-compatible and the value of the GUC parameter **behavior_compat_options** contains **aformat_regexp_match**, the period (.) cannot match the **'\n'** character. However, if **aformat_regexp_match** is not contained, the period (.) matches the **'\n'** character by default.

  Parameters:

  – **source_char**: source character string used for matching.

  – **pattern**: regular expression pattern string used for matching.

  Return type: text.

  Example:

```
gaussdb=# SELECT regexp_substr('500 Hello World, Redwood Shores, CA', ',[^,]+,')
"REGEXPR_SUBSTR";
  REGEXPR_SUBSTR
-------------------
 , Redwood Shores,
(1 row)
```

- regexp_replace(string, pattern, replacement [,flags])

  Description: Replaces any substring matching a POSIX regular expression pattern. The source string is returned if there is no match to the pattern. If there is a match, the matching substring will be replaced by the **replacement** string in the returned string. The string specified by **replacement** can contain **\n**, where **\n** ranges from 1 to 9, to indicate that the substring matching the *n*th parenthesized subexpression of the pattern should be inserted, and it can contain **\&** to indicate that the substring matching the entire pattern should be inserted.

  Parameters:

  – **string**: source character string used for matching.

  – **pattern**: regular expression pattern string used for matching.

  – **flags**: contains zero or multiple single-letter flags that change the matching behavior of the function. This parameter is optional. Flag **g** specifies replacement of each matching substring rather than only the first one. **Table 1-42** describes the options supported by **flags** and their meanings.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: text.

Example:

```
gaussdb=# SELECT regexp_replace('Thomas', '.[mN]a.', 'M');
 regexp_replace
----------------
 ThM
(1 row)
gaussdb=# SELECT regexp_replace('foobarbaz','b(..)', E'X\\1Y', 'g') AS
RESULT;
   result
-------------
 fooXarYXazY
(1 row)
```

- regexp_replace(string text, pattern text [, replacement text [, position int [, occurrence int [, flags text]]]])

  Description: Replaces any substring matching a POSIX regular expression pattern. The source string is returned if there is no match to the pattern. If there is a match, the matching substring will be replaced by the **replacement** string in the returned string.

  Parameters:

  - **string**: source character string used for matching.
  - **pattern**: regular expression pattern string used for matching.
  - **replacement**: character string used to replace the matched substring. This parameter is optional. If no parameter value is specified or the parameter value is null, an empty string is used for replacement.
  - **position**: start character of the source string used for matching. This parameter is optional. The default value is **1**.
  - **occurrence**: sequence number of the matched substring to be replaced. This parameter is optional. The default value is **1**, indicating that the first matched substring is replaced.
  - **flags**: contains zero or multiple single-letter flags that change the matching behavior of the function. This parameter is optional. **Table 1-42** describes the options supported by **flags** and their meanings.

  NOTE

  When **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1** in an Oracle-compatible database, the default value of **occurrence** is **0**, indicating that all matched substrings are replaced. In addition, the **pattern** parameter that ends with a backslash (\) is valid.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT regexp_replace('foobarbaz','b(..)', E'X\\1Y', 2, 2, 'n') AS RESULT;
  result
  ------------
  foobarXazY
  (1 row)
  ```

- concat_ws(sep text, str"any" [, str"any" [, ...]])

  Description: Concatenates all but the first parameter, with separators. The first parameter is used as the separator string. Null parameters will be ignored.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- If the first parameter is **NULL**, the returned result will also be **NULL**.
- If the first parameter is provided but the parameter value is an empty string ('') and the database compatibility mode is set to **ORA**, the returned result is **NULL**. This is because the database interprets the empty string ('') as **NULL** in ORA-compatible mode. To prevent this issue, consider switching to the MySQL-, TD-, or PG-compatible mode.

Return type: text.

Example:

```
gaussdb=# SELECT concat_ws(',', 'ABCDE', 2, NULL, 22);
 concat_ws
------------
 ABCDE,2,22
(1 row)
```

- nlssort(string text, sort_method text)

  Description: Returns the encoding value of a string in the sorting mode specified by **sort_method**. The encoding value can be used for sorting and determines the sequence of the string in the sorting mode. Currently, **sort_method** can be set to **nls_sort=schinese_pinyin_m** or **nls_sort=generic_m_ci**. **nls_sort=generic_m_ci** supports only the case-insensitive order for English characters.

  Parameters: **string** and **sort_method**. Both are of the text type.

  Return type: text.

  Example:

```
gaussdb=# CREATE TABLE test(a text);

gaussdb=# INSERT INTO test(a) VALUES ('abC');

gaussdb=# INSERT INTO test(a) VALUES ('abC');

gaussdb=# INSERT INTO test(a) VALUES ('abc');

gaussdb=# SELECT * FROM test ORDER BY nlssort(a,'nls_sort=schinese_pinyin_m');
   a
--------
 abc
 abC
 abC
(3 rows)

gaussdb=# SELECT * FROM test ORDER BY nlssort(a, 'nls_sort=generic_m_ci');
   a
--------
 abC
 abc
 abC
(3 rows)

gaussdb=# DROP TABLE test;
```

- convert(string bytea, src_encoding name, dest_encoding name)

  Description: Converts the string to **dest_encoding**. **src_encoding** specifies the source encoding scheme. The string must be valid in this encoding scheme.

  Return type: bytea

  Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT convert('text_in_utf8', 'UTF8', 'GBK');
         convert
---------------------------
 \x746578745f696e5f75746638
(1 row)
```

NOTE

If the rule for converting between source to target encoding (for example, GBK and LATIN1) does not exist, the string is returned without conversion. See the PG_CONVERSION system catalog for details. **server_encoding** specifies the character encoding for the database and is specified during database initialization.

Example:

```
gaussdb=# SHOW server_encoding;
 server_encoding
-----------------
 LATIN1
(1 row)

gaussdb=# SELECT convert_from('some text', 'GBK');
 convert_from
--------------
 some text
(1 row)

db_latin1=# SELECT convert_to('some text', 'GBK');
      convert_to
----------------------
 \x736f6d652074657874
(1 row)

db_latin1=# SELECT convert('some text', 'GBK', 'LATIN1');
       convert
----------------------
 \x736f6d652074657874
(1 row)
```

- convert(expr, using transcoding_name)

  Description: Converts a parameter, which can be a character or a number, into a character string of the transcoding_name type, and returns the character string.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT convert('asdas' using 'gbk');
   convert
  ---------
   asdas
  (1 row)
  ```

  NOTE

  This function takes effect only in databases in MySQL-compatible mode.

- convert_from(string bytea, src_encoding name)

  Description: Converts a string based on the character encoding scheme of the database. **src_encoding** specifies the source encoding scheme. The string must be valid in this encoding scheme.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT convert_from('text_in_utf8', 'UTF8');
   convert_from
  --------------
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 text_in_utf8
(1 row)
```

- convert_to(string text, dest_encoding name)

  Description: Converts the string to **dest_encoding**.

  Return type: bytea

  Example:

```
gaussdb=# SELECT convert_to('some text', 'UTF8');
     convert_to
----------------------
 \x736f6d652074657874
(1 row)
```

- string [NOT] LIKE pattern [ESCAPE escape-character]

  Description: Checks whether a match of a pattern occurs.

  – If a pattern does not contain a percentage sign (%) or underscore (_), the pattern represents itself. In this case, the behavior of LIKE is equivalent to the equal operator (=). An underscore (_) in the pattern stands for (matches) any single character; a percent sign (%) can match any sequence of zero or more characters.
  – To match the underscore (_) or percent sign (%), you need to add an escape character before the corresponding character in the pattern. The default escape character is a backslash (\), but you can use the ESCAPE clause to customize an escape character. To match the escape character itself, write two escape characters.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT 'AA_BBCC' LIKE '%A@_B%' ESCAPE '@' AS RESULT;
 result
--------
 t
(1 row)
gaussdb=# SELECT 'AA_BBCC' LIKE '%A@_B%' AS RESULT;
 result
--------
 f
(1 row)
gaussdb=# SELECT 'AA@_BBCC' LIKE '%A@_B%' AS RESULT;
 result
--------
 t
(1 row)
```

- regexp_like(source_string, pattern [, match_parameter])

  Description: Checks whether the source string matches a regular expression pattern. **source_string** specifies the source string, and **pattern** specifies the regular expression pattern to match. **match_parameter** indicates the matching options, which include:

  – **'i'**: case-insensitive.
  – **'c'**: case-sensitive.
  – **'m'**: interprets **source_string** as multiple rows.

  If **match_parameter** is ignored, case-sensitive will be used by default, meaning that "." will not be matched with a newline character and **source_string** will be interpreted as a single row. If an unsupported **match_parameter** option is entered, the message "INFO: illegal argument for function" will be displayed and **false** will be returned in the result.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: Boolean

Example:

```
gaussdb=# SELECT regexp_like('ABC', '[A-Z]');
 regexp_like
-------------
 t
(1 row)
gaussdb=# SELECT regexp_like('ABC', '[D-Z]');
 regexp_like
-------------
 f
(1 row)
gaussdb=# SELECT regexp_like('ABC', '[A-Z]','i');
 regexp_like
-------------
 t
(1 row)
gaussdb=# SELECT regexp_like('ABC', '[A-Z]');
 regexp_like
-------------
 t
(1 row)
```

- format(formatstr text [, str"any" [, ...]])

  Description: Formats a string.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT format('Hello %s, %1$s', 'World');
        format
  --------------------
   Hello World, World
  (1 row)
  ```

- md5(string)

  Description: Encrypts the string in MD5 mode and returns a value in hexadecimal form.

  NOTE

  The MD5 cryptographic algorithm has lower security and poses security risks. Therefore, you are advised to use a more secure cryptographic algorithm.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT md5('ABC');
               md5
  ----------------------------------
   902fbdd2b1df0c4f70b4a5d23525e932
  (1 row)
  ```

- sha(string) / sha1(string)

  Description: Encrypts the string using SHA-1 and returns a hexadecimal number. The sha and sha1 functions are the same.

  NOTE

  - The SHA-1 cryptographic algorithm is not recommended because it has lower security and poses security risks.
  - This function is valid only in MySQL-compatible GaussDB (that is, **sql_compatibility** is set to **'MYSQL'**).

  Return type: text.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT sha('ABC');
                 sha
------------------------------------------
 3c01bdbb26f358bab27f267924aa2c9a03fcfdb8
(1 row)
gaussdb=# SELECT sha1('ABC');
                 sha1
------------------------------------------
 3c01bdbb26f358bab27f267924aa2c9a03fcfdb8
(1 row)
```

- sha2(string, hash_length)

  Description: Encrypts the string in SHA-2 mode and returns a value in hexadecimal form.

  Parameter: **hash_length** corresponds to an SHA-2 algorithm. The value can be **0** (SHA-256), **224** (SHA-224), **256** (SHA-256), **384** (SHA-384), or **512** (SHA-512). For other values, **NULL** is returned.

  NOTE

  - The SHA-224 cryptographic algorithm is not recommended because it has lower security and poses security risks.
  - The SHA2 function records hash plaintext in logs. Therefore, you are advised not to use this function to encrypt sensitive information such as keys.
  - This function is valid only in MySQL-compatible GaussDB (that is, **sql_compatibility** is set to **'MYSQL'**).

  Return type: text.

  Example:

```
gaussdb=# SELECT sha2('ABC',224);
                          sha2
------------------------------------------------------------
 107c5072b799c4771f328304cfe1ebb375eb6ea7f35a3aa753836fad
(1 row)
gaussdb=# SELECT sha2('ABC',256);
                            sha2
------------------------------------------------------------------
 b5d4045c3f466fa91fe2cc6abe79232a1a57cdf104f7a26e716e0a1e2789df78
(1 row)
gaussdb=# SELECT sha2('ABC',0);
                            sha2
------------------------------------------------------------------
 b5d4045c3f466fa91fe2cc6abe79232a1a57cdf104f7a26e716e0a1e2789df78
(1 row)
```

- decode(string text, format text)

  Description: Decodes binary data from textual representation.

  Return type: bytea

  Example:

```
gaussdb=# SELECT decode('MTIzAAE=', 'base64');
   decode
--------------
 \x3132330001
(1 row)
```

- similar_escape(pat text, esc text)

  Description: Converts a regular expression of the SQL:2008 style to the POSIX style.

  Return type: text.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT similar_escape('\s+ab','2');
 similar_escape
----------------
 ^(?:\\s+ab)$
(1 row)
```

- find_in_set(text, set)

  Description: Finds the position of a given member in a set, counting from 1. If no record is found, 0 is returned. The distributed system does not support the SET data type. An error will be reported when this function is executed.

  Return type: int2.

- find_in_set(str, strlist)

  Description: Queries whether the **strlist** column contains **str**. If the **strlist** column contains **str**, the position of **str** in the **strlist** field is returned. If **str** is not in **strlist** or **strlist** is an empty string, the return value is **0**.

  Parameters: See **Table 1-44**.

  Table 1-44 find_in_set(str, strlist) parameters

Parameter	Type	Description
str	text	**str** indicates the character string to be queried.
strlist	text	**strlist** is a set of character strings separated by commas (,). Different character strings are combined by separators. **strlist** is also a character string.

  Return type: int.

  Example:

```
gaussdb=# SELECT find_in_set('ee','a,ee,c');
 find_in_set
-------------
           2
(1 row)
```

- encode(data bytea, format text)

  Description: Encodes binary data into a textual representation.

  Return type: text.

  Example:

```
gaussdb=# SELECT encode(E'123\\000\\001', 'base64');
  encode
----------
 MTIzAAE=
(1 row)
```

- strcmp(expr1, expr2)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Compares two input strings based on the current character order. If the strings are the same, **0** is returned. If the first string is smaller than the second string, **–1** is returned. Otherwise, **1** is returned.

Parameters: See **Table 1-45**.

Table 1-45 strcmp parameters

Parameter	Type	Description
expr1/expr2	• Character types: CHAR, VARCHAR, NVARCHAR2, and TEXT. • Binary type: BYTEA. • Numeric types: TINYINT [UNSIGNED], SMALLINT [UNSIGNED], INTEGER [UNSIGNED], BIGINT [UNSIGNED], FLOAT4, FLOAT8, and NUMERIC. • Date and time types: DATE, TIME WITHOUT TIME ZONE, DATETIME, and TIMESTAMPTZ.	Character string involved in comparison.

Return type: INTEGER

Example:

```
-- Switch to the MySQL-compatible database.
gaussdb=# CREATE DATABASE gaussdb_m dbcompatibility='MYSQL';
gaussdb=# \c gaussdb_m

-- Set the compatible version control parameter to enable the function of specifying the collation for constant character strings.
gaussdb_m=# SET b_format_version='5.7';
gaussdb_m=# SET b_format_dev_version='s2';

gaussdb_m=# SELECT strcmp('abc', 'ABC');
 strcmp
--------
      0
(1 row)

gaussdb_m=# SELECT strcmp('abc     ', 'abc');
 strcmp
--------
      0
(1 row)

gaussdb_m=# SELECT strcmp('1', 1);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 strcmp
--------
      0
(1 row)

gaussdb_m=# SELECT strcmp(123, 2);
 strcmp
--------
     -1
(1 row)
```

NOTE

- The strcmp function is valid only when **sql_compatibility** is set to **'MYSQL'**.
- Since the version whose **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s1'**, the sql_mode parameter **"pad_char_to_full_length"** specifies whether to add spaces at the end of the char type, which affects the strcmp comparison result. For details, see **Table 1-8**.
- Since the version whose **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s1'**, the behavior of the character, binary, numeral, and date and time types is M-compatible. This affects the result of strcmp comparison. For details, see **Data Types**. To ensure consistent precision, it is advisable to use the NUMERIC type rather than floating-point (under the numeric type). The use of floating-point may lead to varying precision compared to M-compatible mode, due to differences in connection parameter settings.
- Since the version whose **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s2'**, character escape and constant character string obtaining collation are supported. The collation affects the strcmp comparison result. For details, see the SET NAMES syntax in **SET**. For details about the rules for combining different collations of the character types, see **Rules for Combining Character Sets and Collations**.

NOTE

- For a string containing linefeed characters, for example, a string consisting of a linefeed character and a space, the values of **LENGTH** and **LENGTHB** in GaussDB are **2**.
- In GaussDB, *n* in the CHAR(n) type indicates the number of characters. Therefore, for multiple-octet coded character sets, the length returned by the LENGTHB function may be longer than *n*.
- GaussDB supports multiple types of databases, including Oracle-, MySQL-, TD-, and PG-compatible. The lexical analyzer of Oracle-compatible database is different from that of the other three databases. In an Oracle-compatible database, an empty string is considered as **NULL**. Therefore, when an Oracle-compatible database is used, if an empty string is used as a parameter in the preceding character operation function, no output is displayed. For example:

  ```
  gaussdb=# SELECT translate('12345','123','');
  translate
  -----------

  (1 row)
  ```

  This is because the database checks whether the input parameter contains **NULL** before calling the corresponding function. If the input parameter contains **NULL**, the kernel does not call the corresponding function. As a result, no output is displayed. In PG-compatible mode, the processing of character strings is the same as that of PG. Therefore, the preceding problem does not occur.

Extension Functions and Operators

- pkg_bpchar_opc()

  Description: Serves as an extension API to add the comparison operator between bpchar and text or between text and bpchar policies, so as to solve

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

the problem that indexes cannot be matched when data of the bpchar and text types is compared. Only the initial user can install extensions.

Example:

- Compare the bpchar type with the text type (initial state, compatible with earlier versions).

```
/*
Create a table and initialize data.
*/
gaussdb=# CREATE TABLE logs_nchar(log_id nchar(16), log_message text);
gaussdb=# INSERT INTO logs_nchar SELECT GENERATE_SERIES(1,100000),MD5(RANDOM());
gaussdb=# INSERT INTO logs_nchar VALUES ('FE306991300002  ','002');
gaussdb=# CREATE INDEX idx_nchar_logid ON logs_nchar(log_id);
gaussdb=# VACUUM ANALYZE logs_nchar;

/*
The purpose is to obtain the detailed execution plan of all nodes.
*/
gaussdb=# SET max_datanode_for_plan = 1;
SET

/*
If no extension is installed, nchar is implicitly converted to text (fixed-length character type to variable-length character type) when nchar and text are compared because the bpchar or text index operator is not available. As a result, the execution plan changes and index cannot be matched.
*/
gaussdb=# EXPLAIN SELECT * FROM logs_nchar WHERE log_id = RPAD(TRIM('FE306991300002
'),16,' ');
                                                 QUERY
PLAN
-------------------------------------------------------------------------------------------------------------------------
 Data Node Scan  (cost=0.00..0.00 rows=0 width=0)
   Node/s: All datanodes

 Remote SQL: SELECT log_id, log_message FROM public.logs_nchar WHERE log_id::text =
rpad(btrim('FE306991300002  '::text), 16, ' '::text)
 Datanode Name: datanode1
   Seq Scan on logs_nchar  (cost=0.00..2539.01 rows=500 width=50)
     Filter: ((log_id)::text = 'FE306991300002  '::text)

(8 rows)

/*
The log_id column in the logs_nchar table is of nchar(16) type. The inserted data is 'FE306991300002  ', which is implicitly converted to text. During comparison, spaces are deleted, that is, 'FE306991300002'='FE306991300002  '. Therefore, the data is not matched.
*/
gaussdb=# SELECT * FROM logs_nchar WHERE log_id = RPAD(TRIM('FE306991300002  '),16,' ');
 log_id | log_message
--------+-------------
(0 rows)

/*
Delete the table.
*/
gaussdb=# DROP TABLE logs_nchar;
```

- Compare the bpchar type with the text type (the pkg_bpchar_opc extension is installed, which is ORA-compatible).

```
/*
Create a table and initialize data.
*/
gaussdb=# CREATE TABLE logs_nchar(log_id nchar(16), log_message text);
gaussdb=# INSERT INTO logs_nchar SELECT GENERATE_SERIES(1,100000),MD5(RANDOM());
gaussdb=# INSERT INTO logs_nchar VALUES ('FE306991300002  ','002');
gaussdb=# CREATE INDEX idx_nchar_logid ON logs_nchar(log_id);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# VACUUM ANALYZE logs_nchar;

/*
A system administrator installs the pkg_bpchar_opc extension. The comparison operators of the
bpchar and text types and index-related content are added to the database.
*/
gaussdb=# CREATE EXTENSION pkg_bpchar_opc;
CREATE EXTENSION

gaussdb=# SET max_datanode_for_plan = 1;
SET

gaussdb=# EXPLAIN SELECT * FROM logs_nchar WHERE log_id = RPAD(TRIM('FE306991300002
'),16,' ');
                                                QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------
-------------------
 Data Node Scan  (cost=0.00..0.00 rows=0 width=0)
   Node/s: All datanodes

 Remote SQL: SELECT log_id, log_message FROM public.logs_nchar WHERE log_id =
rpad(btrim('FE306991300002  '::text), 16, ' '::text)
 Datanode Name: datanode1
   [Bypass]
   Index Scan using idx_nchar_logid on logs_nchar  (cost=0.00..8.27 rows=1 width=50)
     Index Cond: (log_id = 'FE306991300002  '::text)

(9 rows)

/*
In this case, when log_id is implicitly converted to the bpchar type and compared with the text
type, the comparison operator and index information can be found, and the index can be
matched.
*/
gaussdb=# SELECT * FROM logs_nchar WHERE log_id = RPAD(TRIM('FE306991300002  '),16,' ');
      log_id      | log_message
------------------+-------------
 FE306991300002   | 002
(1 row)

/*
Delete the table and extension.
*/
gaussdb=# DROP TABLE logs_nchar;
gaussdb=# DROP EXTENSION pkg_bpchar_opc;
```

- Compare the text type with the bpchar type (initial state, compatible with earlier versions).

```
/*
Create a table and initialize data.
*/
gaussdb=# CREATE TABLE logs_text(log_id nchar(16), log_message text);
gaussdb=# INSERT INTO logs_text SELECT GENERATE_SERIES(1,100000),MD5(RANDOM());
gaussdb=# INSERT INTO logs_text VALUES ('FE306991300002  ','002');
gaussdb=# CREATE INDEX idx_text_logid ON logs_text(log_id);
gaussdb=# VACUUM ANALYZE logs_text;

gaussdb=# SET max_datanode_for_plan = 1;
SET

gaussdb=# EXPLAIN SELECT * FROM logs_text WHERE log_id = 'FE306991300002  '::bpchar;
                                        QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------
 Data Node Scan  (cost=0.00..0.00 rows=0 width=0)
   Node/s: All datanodes

 Remote SQL: SELECT log_id, log_message FROM public.logs_text WHERE log_id =
'FE306991300002  '::bpchar::text
 Datanode Name: datanode1
   [Bypass]
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  Index Scan using idx_text_logid on logs_text  (cost=0.00..8.27 rows=1 width=38)
    Index Cond: (log_id = 'FE306991300002'::text)

(9 rows)

gaussdb=# SELECT * FROM logs_text WHERE log_id = 'FE306991300002  '::bpchar;
 log_id | log_message
--------+-------------
(0 rows)

gaussdb=# SELECT * FROM logs_text WHERE log_id = 'FE306991300002  '::text;
      log_id      | log_message
------------------+-------------
 FE306991300002   | 002
(1 row)

/*
Delete the table.
*/
gaussdb=# DROP TABLE logs_text;
```

- Compare the text type with the bpchar type (the pkg_bpchar_opc extension is installed, which is ORA-compatible).

```
/*
Create a table and initialize data.
*/
gaussdb=# CREATE TABLE logs_text(log_id nchar(16), log_message text);
gaussdb=# INSERT INTO logs_text SELECT GENERATE_SERIES(1,100000),MD5(RANDOM());
gaussdb=# INSERT INTO logs_text VALUES ('FE306991300002  ','002');
gaussdb=# CREATE INDEX idx_text_logid on logs_text(log_id);
gaussdb=# VACUUM ANALYZE logs_text;

gaussdb=# CREATE EXTENSION pkg_bpchar_opc;
CREATE EXTENSION

gaussdb=# SET max_datanode_for_plan = 1;
SET

gaussdb=# EXPLAIN SELECT * FROM logs_text t1 WHERE t1.log_id ='FE306991300002  '::bpchar;
                                    QUERY PLAN
-----------------------------------------------------------------------------------------------------------------
 Data Node Scan  (cost=0.00..0.00 rows=0 width=0)
   Node/s: All datanodes

 Remote SQL: SELECT log_id, log_message FROM public.logs_text t1 WHERE log_id =
'FE306991300002  '::bpchar
 Datanode Name: datanode1
   [Bypass]
   Index Scan using idx_text_logid on logs_text t1  (cost=0.00..8.27 rows=1 width=38)
     Index Cond: (log_id = 'FE306991300002  '::bpchar)

(9 rows)

gaussdb=# SELECT * FROM logs_text WHERE log_id = 'FE306991300002  '::bpchar;
      log_id      | log_message
------------------+-------------
 FE306991300002   | 002
(1 row)

gaussdb=# SELECT * FROM logs_text WHERE log_id = 'FE306991300002  '::text;
      log_id      | log_message
------------------+-------------
 FE306991300002   | 002
(1 row)

/*
Delete the table and extension.
*/
gaussdb=# DROP TABLE logs_text;
gaussdb=# DROP EXTENSION pkg_bpchar_opc;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Compare the hash join and the text type with the bpchar type (initial state, compatible with earlier versions).

```
/*
Create a table and initialize data.
*/
gaussdb=# CREATE TABLE logs_varchar2(log_id varchar2, log_message text);
gaussdb=# INSERT INTO logs_varchar2 VALUES ('FE306991300002  ','002');
gaussdb=# INSERT INTO logs_varchar2 VALUES ('FE306991300003  ','003');
gaussdb=# INSERT INTO logs_varchar2 VALUES ('FE306991300004  ','004');
gaussdb=# VACUUM ANALYZE logs_varchar2;

gaussdb=# CREATE TABLE logs_char(log_id char(16), log_message text);
gaussdb=# INSERT INTO logs_char VALUES ('FE306991300002  ','002');
gaussdb=# INSERT INTO logs_char VALUES ('FE306991300003  ','003');
gaussdb=# INSERT INTO logs_char VALUES ('FE306991300004  ','004');
gaussdb=# VACUUM ANALYZE logs_char;

gaussdb=# SET max_datanode_for_plan = 1;
SET
gaussdb=# SELECT * FROM logs_varchar2 t1, logs_char t2 WHERE t1.log_id = t2.log_id;
      log_id      | log_message |      log_id      | log_message
------------------+-------------+------------------+-------------
 FE306991300002   | 002         | FE306991300002   | 002
 FE306991300003   | 003         | FE306991300003   | 003
 FE306991300004   | 004         | FE306991300004   | 004
(3 rows)

gaussdb=# EXPLAIN SELECT * FROM logs_varchar2 t1, logs_char t2 WHERE t1.log_id = t2.log_id;
                                                
```

```
gaussdb=# INSERT INTO logs_varchar2 VALUES ('FE306991300004  ','004');
gaussdb=# VACUUM ANALYZE logs_varchar2;

gaussdb=# CREATE TABLE logs_char(log_id char(16), log_message text);
gaussdb=# INSERT INTO logs_char VALUES ('FE306991300002  ','002');
gaussdb=# INSERT INTO logs_char VALUES ('FE306991300003  ','003');
gaussdb=# INSERT INTO logs_char VALUES ('FE306991300004  ','004');
gaussdb=# VACUUM ANALYZE logs_char;

gaussdb=# CREATE EXTENSION pkg_bpchar_opc;
CREATE EXTENSION

gaussdb=# SET max_datanode_for_plan = 1;
SET

/*
This format is not recommended. After the extension is installed, the varchar2 type of log_id in the t1 table is implicitly converted to the text type. When it is compared with the log_id in the t2 table, the char type of log_id in the t2 table is implicitly converted to the bpchar type. In this case, spaces after log_id is removed by the database, that is, 'FE306991300002'='FE306991300002  '. Therefore, no data is matched.
*/
/*
Incorrect example:
*/
gaussdb=# SELECT * FROM logs_varchar2 t1, logs_char t2 WHERE t1.log_id = t2.log_id;
 log_id | log_message | log_id | log_message
--------+-------------+--------+-------------
(0 rows)

gaussdb=# EXPLAIN SELECT * FROM logs_varchar2 t1, logs_char t2 WHERE t1.log_id = t2.log_id;
                                                                QUERY
PLAN
------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------
 Data Node Scan  (cost=0.00..0.00 rows=0 width=0)
   Node/s: All datanodes

 Remote SQL: SELECT t1.log_id, t1.log_message, t2.log_id, t2.log_message FROM
public.logs_varchar2 t1, public.logs_char t2 WHERE t1.log_id::text = t2.log_id
 Datanode Name: datanode1
   Hash Join  (cost=1.07..2.14 rows=3 width=42)
     Hash Cond: ((t1.log_id)::text = t2.log_id)
     ->  Seq Scan on logs_varchar2 t1  (cost=0.00..1.03 rows=3 width=21)
     ->  Hash  (cost=1.03..1.03 rows=3 width=21)
           ->  Seq Scan on logs_char t2  (cost=0.00..1.03 rows=3 width=21)

(11 rows)

gaussdb=# SELECT * FROM logs_varchar2 t1, logs_char t2 WHERE t1.log_id = 'FE306991300002
';
     log_id      | log_message |      log_id      | log_message
-----------------+-------------+------------------+-------------
 FE306991300002  | 002         | FE306991300002   | 002
 FE306991300002  | 002         | FE306991300003   | 003
 FE306991300002  | 002         | FE306991300004   | 004
(3 rows)

/*
This format is recommended because it ensures successful data matching, such as 'FE306991300002'='FE306991300002', by forcibly converting the type of the t1 table to bpchar (the type used before the extension is installed). Previously, when log_id in the t1 table was converted to the text type, spaces were retained, leading to mismatches with log_id in the t2 table.
*/
/*
Correct example:
*/
gaussdb=# SELECT * FROM logs_varchar2 t1, logs_char t2 WHERE t1.log_id::bpchar = t2.log_id;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
     log_id      | log_message |      log_id      | log_message
-----------------+-------------+------------------+-------------
 FE306991300002  | 002         | FE306991300002   | 002
 FE306991300003  | 003         | FE306991300003   | 003
 FE306991300004  | 004         | FE306991300004   | 004
(3 rows)

/*
The execution plan is the same as that before the extension is installed.
*/
gaussdb=# EXPLAIN SELECT * FROM logs_varchar2 t1, logs_char t2 WHERE t1.log_id::bpchar =
t2.log_id;
                                                        QUERY
PLAN
------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------
 Data Node Scan  (cost=0.00..0.00 rows=0 width=0)
   Node/s: All datanodes

 Remote SQL: SELECT t1.log_id, t1.log_message, t2.log_id, t2.log_message FROM
public.logs_varchar2 t1, public.logs_char t2 WHERE t1.log_id::bpchar = t2.log_id
 Datanode Name: datanode1
   Hash Join  (cost=1.07..2.14 rows=3 width=42)
     Hash Cond: ((t1.log_id)::bpchar = t2.log_id)
     ->  Seq Scan on logs_varchar2 t1  (cost=0.00..1.03 rows=3 width=21)
     ->  Hash  (cost=1.03..1.03 rows=3 width=21)
           ->  Seq Scan on logs_char t2  (cost=0.00..1.03 rows=3 width=21)

(11 rows)

gaussdb=# SELECT * FROM logs_varchar2 t1, logs_char t2 WHERE t1.log_id = 'FE306991300002
';
     log_id      | log_message |      log_id      | log_message
-----------------+-------------+------------------+-------------
 FE306991300002  | 002         | FE306991300002   | 002
 FE306991300002  | 002         | FE306991300003   | 003
 FE306991300002  | 002         | FE306991300004   | 004
(3 rows)

/*
Implicit conversion between bpchar and text of indexes.
*/
gaussdb=# CREATE DATABASE ora_compatible_db DBCOMPATIBILITY 'ORA';
CREATE DATABASE
gaussdb=# \c ora_compatible_db
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "ora_compatible_db" as user "tom".
ora_compatible_db=# CREATE EXTENSION pkg_bpchar_opc;
CREATE EXTENSION
ora_compatible_db=# CREATE TABLE t1_1(c1 char(5), c2 int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'c1' as the distribution column by
default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
ora_compatible_db=# ALTER TABLE t1_1 ADD CONSTRAINT t1_1_pkey PRIMARY KEY (c1) WITH
(storage_type=USTORE);
NOTICE:  ALTER TABLE / ADD PRIMARY KEY will create implicit index "t1_1_pkey" for table
"t1_1"
ALTER TABLE
ora_compatible_db=# INSERT INTO t1_1 VALUES('abc  ', 0);
INSERT 0 1
ora_compatible_db=# CREATE TABLE t1_2(c1 varchar(5));
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'c1' as the distribution column by
default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
ora_compatible_db=# INSERT INTO t1_2 VALUES('abc  ');
INSERT 0 1
ora_compatible_db=# CREATE /*+INDEXSCAN (t1_1 t1_1_pkey)*/ c1 FROM t1_1 WHERE c2 = 0
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
AND c1 IN (SELECT c1 FROM t1_2);
 c1
-------
 abc
(1 row)
ora_compatible_db=# SET max_datanode_for_plan = 64;
SET
ora_compatible_db=# EXPLAIN CREATE /*+INDEXSCAN (t1_1 t1_1_pkey)*/ c1 FROM t1_1
WHERE c2 = 0 AND c1 IN (SELECT c1 FROM t1_2);
                                                QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------
------------------
 Data Node Scan  (cost=0.00..0.00 rows=0 width=0)
   Node/s: All datanodes

 Remote SQL: SELECT/*+ IndexScan(t1_1 t1_1_pkey)*/ c1 FROM public.t1_1 WHERE c2 = 0 AND
(c1 IN (SELECT t1_2.c1 FROM public.t1_2))
 Datanode Name: dist1_datanode1
   Nested Loop  (cost=28.91..93.76 rows=4 width=24)
     ->  HashAggregate  (cost=28.91..30.91 rows=200 width=24)
           Group By Key: (t1_2.c1)::text
           ->  Seq Scan on t1_2  (cost=0.00..25.13 rows=1513 width=24)
     ->  Index Scan using t1_1_pkey on t1_1  (cost=0.00..0.30 rows=1 width=24)
           Index Cond: (c1 = (t1_2.c1)::text)
           Filter: (c2 = 0)

(13 rows)
/*
Delete the table and extension.
*/
gaussdb=# DROP TABLE t1_1;
gaussdb=# DROP TABLE t1_2;
gaussdb=# DROP TABLE logs_varchar2;
gaussdb=# DROP TABLE logs_char;
gaussdb=# DROP EXTENSION pkg_bpchar_opc;
```

NOTE

- This solves the problem that data and indexes cannot be properly matched when equality matching is performed between the bpchar type (containing multiple spaces) and the text type.
- UB-tree and B-tree are involved. The comparison operators include >, >=, <, <=, and <>.
- The impact scope involves implicit conversion between character types. For example, when a variable-length data type is compared with a fixed-length data type, the variable-length data type is preferentially converted to the text type instead of the original bpchar type.
- The pkg_bpchar_opc extension is disabled by default. You can query the PG_EXTENSION system catalog to check whether the extension is enabled. If the corresponding record exists, the extension is enabled. If the record does not exist, the extension is not enabled. When the extension is disabled, compatibility with earlier versions is maintained. When the extension is enabled, the ORA-compatible mode is maintained. After the pkg_bpchar_opc extension is enabled, you need to set the **max_datanode_for_plan** parameter as required. This parameter specifies the number of execution plans displayed on DNs when the FQS plan is generated. For example, to set **max_datanode_for_plan** to **64**, run the following command:
  ```
  SET max_datanode_for_plan = 64;
  ```
- In the example, the table definition uses **log_id** as the index and has two columns: **log_id** and **log_message**. The table name is followed by the **log_id** column type. For example, if the table name is **logs_text**, the **log_id** column type is text.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-46 Functions supported by pkg_bpchar_opc

API	Description
pg_catalog.bpchar_text_lt	Compares the bpchar type with the text type to check whether the value on the left is less than the value on the right.
pg_catalog.bpchar_text_le	Compares the bpchar type with the text type to check whether the value on the left is less than or equal to the value on the right.
pg_catalog.bpchar_text_eq	Compares the bpchar type with the text type to check whether the value on the left is equal to the value on the right.
pg_catalog.bpchar_text_ge	Compares the bpchar type with the text type to check whether the value on the left is greater than or equal to the value on the right.
pg_catalog.bpchar_text_gt	Compares the bpchar type with the text type to check whether the value on the left is greater than the value on the right.
pg_catalog.bpchar_text_ne	Compares the bpchar type with the text type to check whether the value on the left is different from the value on the right.
pg_catalog.bpchar_text_cmp	Specifies that the index of the bpchar and text types supports comparison functions.
pg_catalog.text_bpchar_lt	Compares the text type with the bpchar type to check whether the value on the left is less than the value on the right.
pg_catalog.text_bpchar_le	Compares the text type with the bpchar type to check whether the value on the left is less than or equal to the value on the right.
pg_catalog.text_bpchar_eq	Compares the text type with the bpchar type to check whether the value on the left is equal to the value on the right.
pg_catalog.text_bpchar_ge	Compares the text type with the bpchar type to check whether the value on the left is greater than or equal to the value on the right.
pg_catalog.text_bpchar_gt	Compares the text type with the bpchar type to check whether the value on the left is greater than the value on the right.
pg_catalog.text_bpchar_ne	Compares the text type with the bpchar type to check whether the value on the left is different from the value on the right.
pg_catalog.text_bpchar_cmp	Specifies that the index of the text and bpchar types supports comparison functions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
pg_catalog.hashbpchartext	Specifies that the hash of the bpchar and text types supports comparison functions.
pg_catalog.hashtextbpchar	Specifies that the hash of the text and bpchar types supports comparison functions.

- bpcharlikebpchar(BPCHAR, BPCHAR)

  Description: Specifies whether the BPCHAR character string of the first input parameter is LIKE and whether the BPCHAR character string of the second input parameter is LIKE. It is used to add the BPCHAR type and the LIKE operator of the BPCHAR type. To enable the ~~ operator, ensure that the value of the GUC parameter **behavior_compat_options** contains the **enable_bpcharlikebpchar_compare** configuration item.

  Return type: Boolean. The value range is as follows:

  – **t**: Two parameters of the BPCHAR type are matched.

  – **f**: Two parameters of the BPCHAR type are not matched.

  Example:

```
gaussdb=#  SELECT bpcharlikebpchar('455'::BPCHAR(10), '455  '::BPCHAR);
 bpcharlikebpchar
------------------
 f
(1 row)
gaussdb=#  SELECT bpcharlikebpchar('455 '::BPCHAR(10), '455 '::BPCHAR(10));
 bpcharlikebpchar
------------------
 t
(1 row)
gaussdb=#  SELECT bpcharlikebpchar('455 '::BPCHAR(10), '455'::BPCHAR(10));
 bpcharlikebpchar
------------------
 t
(1 row)
gaussdb=#  SELECT bpcharlikebpchar('455 '::BPCHAR(10), '455'::BPCHAR(11));
 bpcharlikebpchar
------------------
 f
(1 row)
gaussdb=#  CREATE TABLE op_test (
col BPCHAR(2) DEFAULT NULL
 );
CREATE TABLE
gaussdb=#  CREATE INDEX op_index ON op_test(col);
CREATE INDEX
gaussdb=#  INSERT INTO op_test VALUES ('a');
INSERT 0 1
gaussdb=#  INSERT INTO op_test VALUES ('1');
INSERT 0 1
gaussdb=#  INSERT INTO op_test VALUES ('11');
INSERT 0 1
gaussdb=#  INSERT INTO op_test VALUES ('12');
INSERT 0 1
gaussdb=#  INSERT INTO op_test VALUES ('sd');
INSERT 0 1
gaussdb=#  INSERT INTO op_test VALUES ('aa');
INSERT 0 1
gaussdb=#  SHOW behavior_compat_options;
 behavior_compat_options
-------------------------
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- If behavior_compat_options does not contain enable_bpcharlikebpchar_compare, the latest
bpcharlikebpchar operator is not enabled and the result set returned by the matching between
bpchars is not the same as expected (all data should be returned in normal cases).
gaussdb=# EXPLAIN (COSTS OFF) SELECT * FROM op_test WHERE col LIKE col::BPCHAR ORDER BY
col;
                QUERY PLAN
-------------------------------------------
 Streaming (type: GATHER)
   Merge Sort Key: col
   Node/s: All datanodes
   ->  Sort
         Sort Key: col
         ->  Seq Scan on op_test
               Filter: (col !~~ (col)::text)
(7 rows)
gaussdb=#  SELECT * FROM op_test WHERE col LIKE col::BPCHAR ORDER BY col;
 col
-----
 11
 12
 aa
 sd
(4 rows)
gaussdb=#  SET behavior_compat_options = 'enable_bpcharlikebpchar_compare';
SET
gaussdb=#  SHOW behavior_compat_options;
     behavior_compat_options
---------------------------------
 enable_bpcharlikebpchar_compare
(1 row)
-- After this parameter is enabled, the latest bpcharlikebpchar operator is enabled, and the returned
behavior meets the expected behavior during matching.
gaussdb=#  EXPLAIN (COSTS OFF) SELECT * FROM op_test WHERE col LIKE col::BPCHAR ORDER BY
col;
             QUERY PLAN
------------------------------------
 Streaming (type: GATHER)
   Merge Sort Key: col
   Node/s: All datanodes
   ->  Sort
         Sort Key: col
         ->  Seq Scan on op_test
               Filter: (col ~~ col)
(7 rows)
gaussdb=#  SELECT * FROM op_test WHERE col LIKE col::BPCHAR ORDER BY col;
 col
-----
 1
 11
 12
 a
 aa
 sd
(6 rows)
gaussdb=#  DROP TABLE op_test;
DROP TABLE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- bpcharlikebpchar can be used only when the database compatibility parameter **SQL_COMPATIBILITY** is set to **ORA** and the GUC parameter **behavior_compat_options** contains the **enable_bpcharlikebpchar_compare** column.
- If this feature is enabled, the result set and execution plan for the LIKE pattern matching of the BPCHAR types are affected.
- If the GUC parameter **behavior_compat_options** does not contain the **enable_bpcharlikebpchar_compare** column, this feature is disabled. If the GUC parameter **behavior_compat_options** contains the **enable_bpcharlikebpchar_compare** column, this feature is enabled.
- After the new feature is enabled, fixed-length matching is used (bpchar matches bpchar). The parameter lengths on the left and right sides must be the same. During pattern matching, ensure that the length of the pattern column is the same as the forcible conversion length to avoid the difference between the result and the expected result caused by filling spaces after the length is too long.

- bpcharnlikebpchar(BPCHAR, BPCHAR)

  Description: Specifies whether the first input BPCHAR string is NOT LIKE the second input BPCHAR string. It is used to add a NOT LIKE operator for comparing BPCHAR types. To enable the !~~ operator, ensure that the value of the GUC parameter **behavior_compat_options** contains the **enable_bpcharlikebpchar_compare** configuration item.

  Return type: Boolean. The value range is as follows:

  – **t**: Two BPCHAR parameters do not match.

  – **f**: Two BPCHAR parameters match.

  Example:

```
gaussdb=#  SELECT bpcharnlikebpchar('455 '::BPCHAR(10), '455'::BPCHAR(11));
 bpcharnlikebpchar
-------------------
 t
(1 row)

gaussdb=#  SELECT bpcharnlikebpchar('455 '::BPCHAR(10), '455'::BPCHAR(10));
 bpcharnlikebpchar
-------------------
 f
(1 row)

gaussdb=#  SELECT bpcharnlikebpchar('455 '::BPCHAR(10), '455 '::BPCHAR);
 bpcharnlikebpchar
-------------------
 t
(1 row)
gaussdb=#  CREATE TABLE op_test (
 col BPCHAR(2) DEFAULT NULL
);
CREATE TABLE
gaussdb=#  CREATE INDEX op_index ON op_test(col);
CREATE INDEX
gaussdb=#  INSERT INTO op_test VALUES ('a');
INSERT 0 1
gaussdb=#  INSERT INTO op_test VALUES ('1');
INSERT 0 1
gaussdb=#  INSERT INTO op_test VALUES ('11');
INSERT 0 1
gaussdb=#  INSERT INTO op_test VALUES ('12');
INSERT 0 1
gaussdb=#  INSERT INTO op_test VALUES ('sd');
INSERT 0 1
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=#  INSERT INTO op_test VALUES ('aa');
INSERT 0 1
gaussdb=#  SHOW behavior_compat_options;
 behavior_compat_options
-------------------------
(1 row)
-- If behavior_compat_options does not contain enable_bpcharlikebpchar_compare, the latest
bpcharnlikebpchar operator is not enabled and the result set returned by the matching between
bpchars is not the same as expected (no data record is returned in normal cases).
gaussdb=#  SELECT * FROM op_test WHERE col NOT LIKE col::BPCHAR ORDER BY col;
 col
-----
 1
 a
(2 rows)

gaussdb=#  EXPLAIN (COSTS OFF) SELECT * FROM op_test WHERE col NOT LIKE col::BPCHAR ORDER
BY col;
                QUERY PLAN
-------------------------------------------------
 Streaming (type: GATHER)
   Merge Sort Key: col
   Node/s: All datanodes
   ->  Sort
         Sort Key: col
         ->  Seq Scan on op_test
               Filter: (col !~~ (col)::text)
(7 rows)
gaussdb=#  SET behavior_compat_options = 'enable_bpcharlikebpchar_compare';
SET
gaussdb=#  SHOW behavior_compat_options;
     behavior_compat_options
---------------------------------
 enable_bpcharlikebpchar_compare
(1 row)
-- After this parameter is enabled, the latest bpcharnlikebpchar operator is enabled, and the returned
behavior meets the expected behavior during matching.

gaussdb=#  SELECT * FROM op_test WHERE col NOT LIKE col::BPCHAR ORDER BY col;
 col
-----
(0 rows)

gaussdb=#  EXPLAIN (COSTS OFF) SELECT * FROM op_test WHERE col NOT LIKE col::BPCHAR ORDER
BY col;
             QUERY PLAN
------------------------------------
 Streaming (type: GATHER)
   Merge Sort Key: col
   Node/s: All datanodes
   ->  Sort
         Sort Key: col
         ->  Seq Scan on op_test
               Filter: (col !~~ col)
(7 rows)
gaussdb=#  DROP TABLE op_test;
DROP TABLE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- bpcharlikebpchar can be used only when the database compatibility parameter **SQL_COMPATIBILITY** is set to **ORA** and the GUC parameter **behavior_compat_options** contains the **enable_bpcharlikebpchar_compare** column.
- If this feature is enabled, the result set and execution plan for the NOT LIKE pattern matching of the BPCHAR types are affected.
- If the GUC parameter **behavior_compat_options** does not contain the **enable_bpcharlikebpchar_compare** column, this feature is disabled. If the GUC parameter **behavior_compat_options** contains the **enable_bpcharlikebpchar_compare** column, this feature is enabled.
- After the new feature is enabled, fixed-length matching is used (bpchar matches bpchar). The parameter lengths on the left and right sides must be the same. During pattern matching, ensure that the length of the pattern column is the same as the forcible conversion length to avoid the difference between the result and the expected result caused by filling spaces after the length is too long.

1.6.4 Binary String Functions and Operators

String Operators

SQL defines some string functions that use keywords, rather than commas, to separate arguments.

- octet_length(string)

  Description: Specifies the number of bytes in a binary string.

  Return type: int

  Example:

  ```
  gaussdb=# SELECT octet_length(E'jo\\000se'::bytea) AS RESULT;
   result
  --------
       5
  (1 row)
  ```

- overlay(string placing string from int [for int])

  Description: Replaces substrings.

  Return type: bytea

  Example:

  ```
  gaussdb=# SELECT overlay(E'Th\\000omas'::bytea placing E'\\002\\003'::bytea from 2 for 3) AS
  RESULT;
      result
  ----------------
   \x5402036d6173
  (1 row)
  ```

- position(substring in string)

  Description: Specifies the position of a specified substring.

  Return type: int

  Example:

  ```
  gaussdb=# SELECT position(E'\\000om'::bytea in E'Th\\000omas'::bytea) AS RESULT;
   result
  --------
       3
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- substring(string [from int] [for int])

  Description: Truncates a substring.

  Return type: bytea

  Example:

```
gaussdb=# SELECT substring(E'Th\\000omas'::bytea from 2 for 3) AS RESULT;
 result
----------
 \x68006f
(1 row)
```

- substr(bytea [from int] [for int])

  Description: Truncates a substring.

  Return type: bytea

  Example:

```
gaussdb=# SELECT substr(E'Th\\000omas'::bytea,2, 3) AS RESULT;
 result
----------
 \x68006f
(1 row)
```

- trim([both] bytes from string)

  Description: Removes the longest string containing only **bytes** from the start and end of **string**.

  Return type: bytea

  Example:

```
gaussdb=# SELECT trim(E'\\000'::bytea from E'\\000Tom\\000'::bytea) AS RESULT;
 result
----------
 \x546f6d
(1 row)
```

Other Binary String Functions

GaussDB provides common syntax used for calling functions.

- btrim(string bytea, bytes bytea)

  Description: Removes the longest string containing only **bytes** from the start and end of **string**.

  Return type: bytea

  Example:

```
gaussdb=# SELECT btrim(E'\\000trim\\000'::bytea, E'\\000'::bytea) AS RESULT;
  result
------------
 \x7472696d
(1 row)
```

- get_bit(string, offset)

  Description: Extracts bits from a string.

  Return type: int

  Example:

```
gaussdb=# SELECT get_bit(E'Th\\000omas'::bytea, 45) AS RESULT;
 result
--------
      1
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> The first parameter can only be of the bit or bytea type.

- get_byte(string, offset)

  Description: Extracts bytes from a string.

  Return type: int

  Example:

```
gaussdb=# SELECT get_byte(E'Th\\000omas'::bytea, 4) AS RESULT;
 result
--------
    109
(1 row)
```

- set_bit(string, offset, newvalue)

  Description: Sets bits in a string.

  Return type: bytea

  Example:

```
gaussdb=# SELECT set_bit(E'Th\\000omas'::bytea, 45, 0) AS RESULT;
      result
------------------
 \x5468006f6d4173
(1 row)
```

> **NOTE**
>
> The first parameter can only be of the bit or bytea type.

- set_byte(string, offset, newvalue)

  Description: Sets bytes in a string.

  Return type: bytea

  Example:

```
gaussdb=# SELECT set_byte(E'Th\\000omas'::bytea, 4, 64) AS RESULT;
      result
------------------
 \x5468006f406173
(1 row)
```

- rawcmp(raw, raw)

  Description: Specifies the raw data type comparison function.

  Return type: integer

- raweq(raw, raw)

  Description: Specifies the raw data type comparison function.

  Return type: Boolean

- rawge(raw, raw)

  Description: Specifies the raw data type comparison function.

  Return type: Boolean

- rawgt(raw, raw)

  Description: Specifies the raw data type comparison function.

  Return type: Boolean

- rawin(cstring)

  Description: Specifies the raw data type parsing function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: bytea

- rawle(raw, raw)
  Description: Specifies the raw data type parsing function.
  Return type: Boolean
- rawlike(raw, raw)
  Description: Specifies the raw data type parsing function.
  Return type: Boolean
- rawlt(raw, raw)
  Description: Specifies the raw data type parsing function.
  Return type: Boolean
- rawne(raw, raw)
  Description: Compares whether the raw types are the same.
  Return type: Boolean
- rawnlike(raw, raw)
  Description: Checks whether the raw type matches the pattern.
  Return type: Boolean
- rawout(bytea)
  Description: Specifies the output API of the raw type.
  Return type: cstring
- rawsend(raw)
  Description: Converts the bytea type to the binary type.
  Return type: bytea
- rawtohex(text)
  Description: Converts the raw format to the hexadecimal format.
  Return type: text

1.6.5 Bit String Functions and Operators

Bit String Operators

Aside from the usual comparison operators, the following operators can be used. Bit string operands of **&**, **|**, and **#** must be of equal length. In case of bit shifting, the original length of the bit string is preserved by padding with 0s.

- ||
  Description: Connects bit strings.
  Example:

```
gaussdb=# SELECT B'10001' || B'011' AS RESULT;
  result
----------
 10001011
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- A field can have a maximum of 180 consecutive internal joins. A field with excessive joins will be split into joined consecutive strings.

  Example: **str1||str2||str3||str4** is split into **(str1||str2)||(str3||str4)**.
- In ORA-compatible mode, if bit strings contain a null string, the null string is ignored and other strings are joined. In other compatibility modes, the null string is returned.

  Take **str1||NULL||str2** as an example. **str1str2** is returned in ORA-compatible mode and **NULL** is returned in other compatibility modes.

- &

  Description: Specifies the AND operation between bit strings.

  Example:

```
gaussdb=# SELECT B'10001' & B'01101' AS RESULT;
 result
--------
 00001
(1 row)
```

- |

  Description: Specifies the OR operation between bit strings.

  Example:

```
gaussdb=# SELECT B'10001' | B'01101' AS RESULT;
 result
--------
 11101
(1 row)
```

- #

  Description: Specifies the OR operation between bit strings if they are inconsistent. If the same positions in the two bit strings are both 1 or 0, the position returns **0**.

  Example:

```
gaussdb=# SELECT B'10001' # B'01101' AS RESULT;
 result
--------
 11100
(1 row)
```

- ~

  Description: Specifies the NOT operation between bit strings.

  Example:

```
gaussdb=# SELECT ~B'10001'AS RESULT;
 result
----------
 01110
(1 row)
```

- <<

  Description: Shifts left in a bit string.

  Example:

```
gaussdb=# SELECT B'10001' << 3 AS RESULT;
 result
----------
 01000
(1 row)
```

- >>

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Shifts right in a bit string.

Example:

```
gaussdb=# SELECT B'10001' >> 2 AS RESULT;
 result
----------
 00100
(1 row)
```

The following SQL-standard functions work on bit strings as well as strings: length, bit_length, octet_length, position, substring, and overlay.

The following functions work on bit strings as well as binary strings: get_bit and set_bit. When working with a bit string, these functions number the first (leftmost) bit of the string as bit 0.

In addition, it is possible to convert between integral values and type bit. Example:

```
gaussdb=# SELECT 44::bit(10) AS RESULT;
   result
------------
 0000101100
(1 row)

gaussdb=# SELECT 44::bit(3) AS RESULT;
 result
--------
 100
(1 row)

gaussdb=# SELECT cast(-44 as bit(12)) AS RESULT;
    result
--------------
 111111010100
(1 row)

gaussdb=# SELECT '1110'::bit(4)::integer AS RESULT;
 result
--------
     14
(1 row)

gaussdb=# SELECT substring('10101111'::bit(8), 2);
 substring
-----------
 0101111
(1 row)
```

NOTE

Casting to just "bit" means casting to bit(1), and so will deliver only the least significant bit of the integer.

1.6.6 Pattern Matching Operators

The database provides three independent methods for implementing pattern matching: SQL LIKE operator, SIMILAR TO operator, and POSIX regular expressions. Besides these basic operators, functions can be used to extract or replace matching substrings and to split a string at matching locations.

- LIKE

  Description: Specifies whether the string matches the pattern string following LIKE. The LIKE expression returns true if the string matches the supplied pattern. (As expected, the NOT LIKE expression returns false if LIKE returns true, and vice versa.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Matching rules:

- LIKE pattern matching always covers the entire string. Therefore, if it is desired to match a sequence anywhere within a string, the pattern must start and end with a percent sign.
- An underscore (_) represents (matches) any single character. A percent sign (%) represents the wildcard character of any string.
- To match a literal underscore or percent sign, the respective character in the given pattern must be preceded by the escape character. Consequently, the special meanings of metacharacters will be disabled. The default escape character is the backslash but a different one can be selected by using the ESCAPE clause.
- To match the escape character itself, write two escape characters. For example, to write a pattern constant that contains a backslash, you must write two backslashes in the SQL statement.

  NOTE

  - When **standard_conforming_strings** is set to **off**, any backslashes included in literal string constants will undergo dual-writes. Therefore, writing a pattern that actually matches a literal backslash means writing four backslashes in the SQL statement. You can avoid this by selecting a different escape character with ESCAPE; then backslash is not special to LIKE anymore. (But it is still special to the string literal parser, so you still need two of them.)
  - In MySQL-compatible mode, it is also possible to select no escape character by writing **ESCAPE ''**. This effectively disables the escape mechanism, which makes it impossible to disable the special meanings of underscores and percent signs in the pattern.
  - In Oracle-compatible mode, if **standard_conforming_strings** is set to **on**, you can set **behavior_compat_options** to **ignore_like_default_escape**. If **ESCAPE '\'** is not explicitly specified, the special meaning of the backslash (\) is disabled and a single backslash (\) is matched.
    - This option takes effect only for keywords (LIKE, NOT LIKE, ILIKE, and NOT ILIKE) and does not take effect for operators (~~, !~~, ~~*, and !~~*).
    - This option does not take effect when the pattern string contains the ANY, SOME, or ALL clause.
- The keyword ILIKE can be used instead of LIKE to make the matching case-insensitive.
- Operator **~~** is equivalent to LIKE, and operator **~~*** corresponds to ILIKE.

Example:

```
gaussdb=# SELECT 'abc' LIKE 'abc' AS RESULT;
 result
------------
 t
(1 row)
gaussdb=# SELECT 'abc' LIKE 'a%' AS RESULT;
 result
------------
 t
(1 row)
gaussdb=# SELECT 'abc' LIKE '_b_' AS RESULT;
 result
------------
 t
(1 row)
gaussdb=# SELECT 'abc' LIKE 'c' AS RESULT;
 result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-----------
 f
(1 row)
```

- SIMILAR TO

  Description: Returns true or false depending on whether the pattern matches the given string. It is similar to LIKE, except that it interprets the pattern using the SQL standard's definition of a regular expression.

  Matching rules:

  – Same as LIKE, this operator succeeds only if its pattern matches the entire string. To match a sequence anywhere within a string, the pattern must start and end with a percent sign (%).

  – An underscore (_) represents (matches) any single character. A percent sign (%) represents the wildcard character of any string.

  – SIMILAR TO supports these pattern-matching metacharacters borrowed from POSIX regular expressions.

Metacharacter	Description
|	Specifies alternation (either of two alternatives).
*	Specifies repetition of the previous item zero or more times.
+	Specifies repetition of the previous item one or more times.
?	Specifies repetition of the previous item zero or one time.
{m}	Specifies repetition of the previous item exactly *m* times.
{m,}	Specifies repetition of the previous item *m* or more times.
{m,n}	Specifies repetition of the previous item at least *m* times and does not exceed *n* times.
()	Combines multiple items into a logical item.
[...]	Specifies a character class which is the same as the POSIX regular expression.

  – A preamble escape character disables the special meaning of any of these metacharacters. The rules for using escape characters are the same as those for LIKE.

  Regular expressions:

  The **substring(string from pattern for escape)** function extracts a substring that matches an SQL regular expression pattern.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT 'abc' SIMILAR TO 'abc' AS RESULT;
 result
-----------
 t
(1 row)
gaussdb=# SELECT 'abc' SIMILAR TO 'a' AS RESULT;
 result
-----------
 f
(1 row)
gaussdb=# SELECT 'abc' SIMILAR TO '%(b|d)%' AS RESULT;
 result
-----------
 t
(1 row)
gaussdb=# SELECT 'abc' SIMILAR TO '(b|c)%'  AS RESULT;
 result
-----------
 f
(1 row)
```

- POSIX regular expressions

  Description: A regular expression is a character sequence that is an abbreviated definition of a set of strings (a regular set). A string is said to match a regular expression if it is a member of the regular set described by the regular expression. POSIX regular expressions provide a more powerful means for pattern matching than the LIKE and SIMILAR TO operators. **Table 1-47** lists all available operators for pattern matching using POSIX-style regular expressions.

  Table 1-47 Regular expression matching operators

Operator	Description	Example
~	Matches a regular expression, which is case-sensitive.	'thomas' ~ '.*thomas.*'
~*	Matches a regular expression, which is case-insensitive.	'thomas' ~* '.*Thomas.*'
!~	Does not match a regular expression, which is case-sensitive.	'thomas' !~ '.*Thomas.*'
!~*	Does not match a regular expression, which is case-insensitive.	'thomas' !~* '.*vadim.*'

  Matching rules:

  – Unlike LIKE patterns, a regular expression is allowed to match anywhere within a string, unless the regular expression is explicitly anchored to the beginning or end of the string.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- In addition to the metacharacters mentioned above, POSIX regular expressions support the following pattern matching metacharacters.

Metacharacter	Description
^	Specifies the match starting with a string.
$	Specifies the match at the end of a string.
.	Matches any single character.

Regular expression functions: POSIX regular expressions support the following functions:

- The **substring(string from pattern)** function provides a method for extracting a substring that matches a POSIX regular expression pattern.
- The **regexp_count(string text, pattern text [, position int [, flags text]])** function obtains the number of substrings that match a POSIX regular expression pattern.
- The **regexp_instr(string text, pattern text [, position int [, occurrence int [, return_opt int [, flags text]]]])** function obtains the position of a substring that matches a POSIX regular expression pattern.
- The **regexp_substr(string text, pattern text [, position int [, occurrence int [, flags text]]])** function provides a method for extracting a substring that matches a POSIX regular expression pattern.
- The **regexp_replace(string, pattern, replacement [,flags])** function replaces the substring that matches the POSIX regular expression pattern with the new text.
- The **regexp_matches(string text, pattern text [, flags text])** function returns a text array consisting of all captured substrings that match a POSIX regular expression pattern.
- The **regexp_split_to_table(string text, pattern text [, flags text])** function splits a string using a POSIX regular expression pattern as a delimiter.
- The **regexp_split_to_array(string text, pattern text [, flags text])** function behaves the same as regexp_split_to_table, except that regexp_split_to_array returns its result as an array of text.

  NOTE

  The regular expression split functions ignore zero-length matches, which occur at the beginning or end of a string or after the previous match. This is contrary to the strict definition of regular expression matching. The latter is implemented by **regexp_matches**, but the former is usually the most commonly used behavior in practice.

Example:

```
gaussdb=# SELECT 'abc' ~ 'Abc' AS RESULT;
result
--------
 f
(1 row)
gaussdb=# SELECT 'abc' ~* 'Abc' AS RESULT;
 result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------
 t
(1 row)
gaussdb=# SELECT 'abc' !~ 'Abc' AS RESULT;
 result
--------
 t
(1 row)
gaussdb=# SELECT 'abc'!~* 'Abc' AS RESULT;
 result
--------
 f
(1 row)
gaussdb=# SELECT 'abc' ~ '^a' AS RESULT;
 result
--------
 t
(1 row)
gaussdb=# SELECT 'abc' ~ '(b|d)'AS RESULT;
 result
--------
 t
(1 row)
gaussdb=# SELECT 'abc' ~ '^(b|c)'AS RESULT;
 result
--------
 f
(1 row)
```

Although most regular expression searches can be executed quickly, they can still be artificially processed to require any length of time and any amount of memory. Avoid accepting regular expression search patterns from non-security pattern sources. If you have to, consider setting a statement timeout interval. The search with the SIMILAR TO pattern has the same security risks, as SIMILAR TO provides many capabilities that are the same as those of the POSIX regular expression. The LIKE search is much simpler than the other two options. Therefore, it is more secure to accept the non-secure pattern source search.

- [NOT] REGEXP/ [NOT] RLIKE

  Description: The REGEXP operator is used for regular expression matching and complies with POSIX regular expression matching rules. True or false is returned based on whether the pattern matches the given string.

  The following table describes the regular expression operators.

Operator Name	Description	Syntax
REGEXP	Specifies whether the character string matches the regular expression.	expr REGEXP pat
RLIKE	Specifies whether the character string matches the regular expression (same as REGEXP).	expr RLIKE pat

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operator Name	Description	Syntax
NOT REGEXP	Specifies whether the character string does not match the regular expression.	expr NOT REGEXP pat
NOT RLIKE	Specifies whether the character string does not match the regular expression (same as NOT REGEXP).	expr NOT RLIKE pat

Matching rules:

- A regular expression is allowed to match anywhere within a string, unless the regular expression is explicitly anchored to the beginning or end of the string.
- The pattern matching metacharacters supported by the REGEXP operator are the same as those supported by POSIX regular expressions.
- The operator supports matching the following escape characters.

Escape Character	Description
\b	Backspace key.
\f	Form feed character, for example, C language.
\n	Newline character, for example, C language.
\r	Carriage return character, for example, C language.
\t	Horizontal tab character, for example, C language.
\uwxyz	Character whose hexadecimal value is **0xwxyz**, where **wxyz** is four hexadecimal digits.
\xhhh	Character whose hexadecimal value is **0xhhh**, where **hhh** is any sequence of hexadecimal digits.
\xy	Character whose octal value is **0xy**, where **xy** is two octal digits.
\xyz	Character whose octal value is **0xyz**, where **xyz** is three octal digits.

- In the pattern string **pat**, **[a-dX]** and **[^a-dX]** stand for the range of matching.

  [a-dX] matches any characters of a, b, c, d, and X. **[^a-dX]** matches characters other than a, b, c, d, or X.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The hyphen (-) between two characters forms a range, indicating that all characters in the range are matched. To include a right square bracket (]), it must follow the left square bracket ([). To include a hyphen (-), it must be after the left square bracket ([) or before the right square bracket (]). Any character that does not have any special meaning enclosed in square brackets ([]) matches itself.

Example:

```
gaussdb=# SELECT 'abd' REGEXP 'a[bc]d' AS RESULT;
 result
--------
 t
(1 row)
gaussdb=# SELECT 'aed' REGEXP 'a[^bc]d' AS RESULT;
 result
--------
 t
(1 row)
gaussdb=# SELECT 'a-' REGEXP 'a[-b]' AS RESULT;
 result
--------
 t
(1 row)
gaussdb=# SELECT 'aX]bc' REGEXP '^[]a-dXYZ]*$' AS RESULT;
 result
--------
 t
(1 row)
```

- In the pattern string **pat**, **[.characters.]** stands for the character sequence of an element. Within a bracket expression (enclosed in []), it matches the character sequence of a collating element. These characters can be individual characters or character names such as **space**. A complete list of character names can be found in the **regex/regc_locale** file.

  Example:

  ```
  gaussdb=# SELECT ' ' REGEXP '[[.space.]]' AS RESULT;
   result
  --------
   t
  (1 row)
  ```

- Character class matching the **[=character_class=]** character in the pattern string pat. It is written in the square bracket expression. **[=character_class=]** indicates the equivalence class. A character matches all characters with the same sort proofreading value, including itself. For example, if o and (+) are of the same class, [[=o=]], [[=(+)=]], and [o(+)] are synonyms. The same class cannot be used as an endpoint of a range.
- Character class matching the **[:character_class:]** character in the pattern string **pat**. It is written in square brackets ([]). **[:character_class:]** is used to match the characters that match the character class. Other class names may be provided for specific regions. The character class cannot be used as an endpoint of a range. The following table lists the standard class names. If the backslash (\) is involved, set the parameters according to the description.

Character Class	Description	Character Range
alnum	Alphanumeric numeric character.	[0-9a-zA-Z]

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Character Class	Description	Character Range
alpha	Alphanumeric character.	[a-zA-Z]
blank	Blank character.	[\t], indicating a blank character
cntrl	Control character.	[\x01-\x1F]
digit	Digit character.	[0-9]
graph	Graphic character.	[^\x01-\x20]
lower	Lowercase character.	[a-z]
print	Graphic character.	[^\x01-\x20]
punct	Punctuation character.	[-!"#$%&'( )*+,./:;<=>?@[\ \]^_`{|}~]
space	Space, tab, new line character, and carriage return.	[\n\r\t\x0B]
upper	Uppercase character.	[A-Z]
xdigit	Hexadecimal numeric character.	[0-9a-fA-F]

Example:

```
gaussdb=# SELECT '\n' REGEXP '[[:space:]]' AS RESULT;
 result
--------
 t
(1 row)
```

- In the pattern string **pat**, [[:<:]] and [[:>:]] match the null string at the start and end positions of a string, respectively.

  Example:

  ```
  gaussdb=# SELECT 'a word a' REGEXP '[[:<:]]word[[:>:]]' AS RESULT;
   result
  --------
   t
  (1 row)
  ```

- To match a text instance with special characters, you need to add two backslashes (\\) before the following special characters: ( ) '' ^ + ? If single quotation marks are used, you need to write **\''** in the source string to match **'\\\''** in the pattern string. If the backslash (\) is involved, set the parameters according to the description.

  Example:

  ```
  gaussdb=# SELECT 'a+b' REGEXP 'a\\+b' AS RESULT;
   result
  --------
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 t
(1 row)
gaussdb=# SELECT '\" REGEXP '\\\" AS RESULT;
 result
--------
 t
(1 row)
gaussdb=# SELECT '\\' REGEXP '\\\\' AS RESULT;
 result
--------
 t
(1 row)
```

- When the case-sensitive character set and collation are set, regular expression matching is also case-sensitive.

  Example:

```
gaussdb=# SELECT 'abc' REGEXP 'ABC' COLLATE utf8mb4_bin AS RESULT;
 result
--------
 f
(1 row)
```

- If the input parameter **Expr** or **pat** is null, null will be returned.

```
gaussdb=# SELECT NULL REGEXP '*' AS RESULT;
 result
--------

(1 row)
gaussdb=# SELECT '-' REGEXP NULL AS RESULT;
 result
--------

(1 row)
```

NOTE

- This operator is supported only in MySQL-compatible databases, that is, it is valid only when **sql_compatibility** is set to **'MYSQL'**. After **b_format_version** is set to **'5.7'**, the REGEXP operator is equivalent to the ~* operator, and the NOT REGEXP operator is equivalent to the !~* operator.
- When **b_format_dev_version** is set to **s2**, **standard_conforming_strings** is set to **off** and **escape_string_warning** is set to **off** by default. In this case, any backslash written in the string constant must be entered again. Therefore, writing a pattern matching a single backslash is actually going to write four backslashes in the statement.
- This operator supports only the string type, text type, and other data types that can be implicitly converted to the string type and text type. The bytea type is not supported.

1.6.7 Arithmetic Functions and Operators

Arithmetic Operators

- +

  Description: Addition

  Example:

```
gaussdb=# SELECT 2+3 AS RESULT;
 result
--------
      5
(1 row)
```

- -

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Subtraction

Example:

```
gaussdb=# SELECT 2-3 AS RESULT;
 result
--------
     -1
(1 row)
```

- *

  Description: Multiplication

  Example:

  ```
  gaussdb=# SELECT 2*3 AS RESULT;
   result
  --------
       6
  (1 row)
  ```

- /

  Description: Division (The result is not rounded.)

  Example:

  ```
  gaussdb=# SELECT 4/2 AS RESULT;
   result
  --------
       2
  (1 row)
  gaussdb=# SELECT 4/3 AS RESULT;
        result
  ------------------
   1.33333333333333
  (1 row)
  ```

- +/-

  Description: Positive/Negative

  Example:

  ```
  gaussdb=# SELECT -2 AS RESULT;
   result
  --------
     -2
  (1 row)
  ```

- %

  Description: Model (to obtain the remainder)

  Example:

  ```
  gaussdb=# SELECT 5%4 AS RESULT;
   result
  --------
       1
  (1 row)
  ```

- @

  Description: Absolute value

  Example:

  ```
  gaussdb=# SELECT @ -5.0 AS RESULT;
   result
  --------
    5.0
  (1 row)
  ```

- ^

  Description: Power (exponent calculation)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT 2.0^3.0 AS RESULT;
       result
--------------------
 8.0000000000000000
(1 row)
```

- |/

  Description: Square root

  Example:

```
gaussdb=# SELECT |/ 25.0 AS RESULT;
 result
--------
      5
(1 row)
```

- ||/

  Description: Cubic root

  Example:

```
gaussdb=# SELECT ||/ 27.0 AS RESULT;
 result
--------
      3
(1 row)
```

- !

  Description: Factorial

  Example:

```
gaussdb=# SELECT 5! AS RESULT;
 result
--------
    120
(1 row)
```

- !!

  Description: Factorial (prefix operator)

  Example:

```
gaussdb=# SELECT !!5 AS RESULT;
 result
--------
    120
(1 row)
```

- &

  Description: Binary AND

  Example:

```
gaussdb=# SELECT 91&15  AS RESULT;
 result
--------
     11
(1 row)
```

- |

  Description: Binary OR

  Example:

```
gaussdb=# SELECT 32|3  AS RESULT;
 result
--------
     35
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- #

  Description: Binary XOR

  Example:

```
gaussdb=# SELECT 17#5  AS RESULT;
 result
--------
     20
(1 row)
```

- ~

  Description: Binary NOT

  Example:

```
gaussdb=# SELECT ~1 AS RESULT;
 result
--------
     -2
(1 row)
```

- <<

  Description: Binary shift left

  Example:

```
gaussdb=# SELECT 1<<4 AS RESULT;
 result
--------
     16
(1 row)
```

- >>

  Description: Binary shift right

  Example:

```
gaussdb=# SELECT 8>>2 AS RESULT;
 result
--------
      2
(1 row)
```

Arithmetic Functions

- abs(x)

  Description: Absolute value

  Return type: same as the input.

  Example:

```
gaussdb=# SELECT abs(-17.4);
 abs
------
 17.4
(1 row)
```

- acos(x)

  Description: Arc cosine

  Return type: double precision

  Example:

```
gaussdb=# SELECT acos(-1);
       acos
------------------
 3.14159265358979
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- asin(x)

  Description: Arc sine

  Return type: double precision

  Example:

```
gaussdb=# SELECT asin(0.5);
       asin
------------------
 .523598775598299
(1 row)
```

- atan(x)

  Description: Arc tangent

  Return type: double precision

  Example:

```
gaussdb=# SELECT atan(1);
       atan
------------------
 .785398163397448
(1 row)
```

- atan2(y, x)

  Description: Arc tangent of y/x

  Return type: double precision

  Example:

```
gaussdb=# SELECT atan2(2, 1);
      atan2
------------------
 1.10714871779409
(1 row)
```

- bitand(integer, integer)

  Description: Performs the AND (&) operation on two integers.

  Return type: bigint

  Example:

```
gaussdb=# SELECT bitand(127, 63);
 bitand
--------
     63
(1 row)
```

- cbrt(dp)

  Description: Cubic root

  Return type: double precision

  Example:

```
gaussdb=# SELECT cbrt(27.0);
 cbrt
------
    3
(1 row)
```

- ceil(x)

  Description: Minimum integer greater than or equal to the parameter

  Return type: integer

  Example:

```
gaussdb=# SELECT ceil(-42.8);
 ceil
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
------
  -42
(1 row)
```

- ceiling(dp or numeric)

  Description: Minimum integer (alias of ceil) greater than or equal to the parameter

  Return type: same as the input.

  Example:

```
gaussdb=# SELECT ceiling(-95.3);
 ceiling
---------
     -95
(1 row)
```

- cos(x)

  Description: Cosine

  Return type: double precision

  Example:

```
gaussdb=# SELECT cos(-3.1415927);
        cos
-------------------
 -.999999999999999
(1 row)
```

- cosh(x)

  Description: Hyperbolic cosine

  Return type: dp or numeric. If implicit type conversion is not considered, the return type is the same as the input type.

  Example:

```
gaussdb=# SELECT cosh(4);
        cosh
-------------------
 27.3082328360165
(1 row)
```

  NOTE

  This function is valid only when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s2** in an ORA-compatible database.

- cot(x)

  Description: Cotangent

  Return type: double precision

  Example:

```
gaussdb=# SELECT cot(1);
       cot
------------------
 .642092615934331
(1 row)
```

- degrees(dp)

  Description: Converts radians to angles.

  Return type: double precision

  Example:

```
gaussdb=# SELECT degrees(0.5);
    degrees
```

```
------------------
 28.6478897565412
(1 row)
```

- div(y numeric, x numeric)

  Description: Integer part of y/x

  Return type: numeric

  Example:

```
gaussdb=# SELECT div(9,4);
 div
-----
   2
(1 row)
```

- exp(x)

  Description: Natural exponent

  Return type: same as the input.

  Example:

```
gaussdb=# SELECT exp(1.0);
        exp
--------------------
 2.7182818284590452
(1 row)
```

- floor(x)

  Description: Not larger than the maximum integer of the parameter

  Return type: same as the input.

  Example:

```
gaussdb=# SELECT floor(-42.8);
 floor
-------
   -43
(1 row)
```

- int1(in)

  Description: Converts the input text parameter into the int1 type and returns the result.

  Return type: int1

  Example:

```
gaussdb=# SELECT int1('123');
 int1
------
 123
(1 row)
gaussdb=# SELECT int1('1.1');
 int1
------
   1
(1 row)
```

- int2(in)

  Description: Converts the input parameter into the int2 type and returns the result. The supported input parameter types include bigint, float4, float8, int16, integer, numeric, real, and text.

  Return type: int2.

  Example:

```
gaussdb=# SELECT int2('1234');
 int2
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
------
 1234
(1 row)
gaussdb=# SELECT int2(25.3);
 int2
------
   25
(1 row)
```

- int4(in)

  Description: Converts the input parameter into the int4 type and returns the result. The supported input parameter types include bit, Boolean, char, double precision, int16, numeric, real, smallint, and text

  Return type: int4

  Example:

```
gaussdb=# SELECT int4('789');
 int4
------
 789
(1 row)
gaussdb=# SELECT int4(99.9);
 int4
------
  100
(1 row)
```

- int8(in)

  Description: Converts the input parameter into the int8 type and returns the result. The supported input parameter types include bit, double precision, int16, integer, numeric, oid, real, smallint, and text.

  Return type: int8

  Example:

```
gaussdb=# SELECT int8('789');
 int8
------
 789
(1 row)
gaussdb=# SELECT int8(99.9);
 int8
------
   99
(1 row)
```

- float4(in)

  Description: Converts the input parameter into the float4 type and returns the result. The supported input parameter types include bigint, double precision, int16, integer, numeric, smallint, and text.

  Return type: float4

  Example:

```
gaussdb=# SELECT float4('789');
 float4
--------
    789
(1 row)

gaussdb=# SELECT float4(99.9);
 float4
--------
   99.9
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- float8(in)

  Description: Converts the input parameter into the float8 type and returns the result. The supported input parameter types include bigint, int16, integer, numeric, real, smallint, and text.

  Return type: float8

  Example:

```
gaussdb=# SELECT float8('789');
 float8
--------
    789
(1 row)

gaussdb=# SELECT float8(99.9);
 float8
--------
   99.9
(1 row)
```

- int16(in)

  Description: Converts the input parameter into the int16 type and returns the result. The supported input parameter types include bigint, Boolean, double precision, integer, numeric, oid, real, smallint, and tinyint.

  Return type: int16

  Example:

```
gaussdb=# SELECT int16('789');
 int16
--------
    789
(1 row)

gaussdb=# SELECT int16(99.9);
 int16
--------
   100
(1 row)
```

- numeric(in)

  Description: Converts the input parameter into the numeric type and returns the result. The supported input parameter types include bigint, Boolean, double precision, int16, integer, money, real, and smallint.

  Return type: numeric

  Example:

```
gaussdb=# SELECT "numeric"('789');
 numeric
---------
     789
(1 row)

gaussdb=# SELECT "numeric"(99.9);
 numeric
---------
    99.9
(1 row)
```

- oid(in)

  Description: Converts the input parameter into the oid type and returns the result. The supported input parameter types include bigint and int16.

  Return type: oid

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- radians(dp)

  Description: Converts angles to radians.

  Return type: double precision

  Example:

```
gaussdb=# SELECT radians(45.0);
     radians
------------------
 .785398163397448
(1 row)
```

- random()

  Description: Random number between 0.0 and 1.0.

  Return type: double precision

  Example:

```
gaussdb=# SELECT random();
      random
------------------
 .824823560658842
(1 row)
```

  NOTE

  You are advised not to use this function in security scenarios.

- rand([seed])

  Description: The input can be without any parameters or only include a **seed** parameter of the bigint type. Typically, this function returns a random number from 0 to 1. If **seed** is specified, this function will return a random value for **seed**.

  Parameter: a random number seed of the bigint type.

  Return type: double

  Example:

```
b_compatible_db=# SELECT rand();
       rand
------------------
 .327476012520492
(1 row)

b_compatible_db=# SELECT rand(12321);
       rand
------------------
 .326073104515672
(1 row)
```

  NOTE

  - This function takes effect when the database parameter **sql_compatibility** is set to **'MYSQL'**.
  - You are advised not to use this function in security scenarios.

- multiply(x double precision or text, y double precision or text)

  Description: product of x and y.

  Return type: double precision

  Example:

```
gaussdb=# SELECT multiply(9.0, '3.0');
     multiply
------------------
```

```
            27
(1 row)
gaussdb=# SELECT multiply('9.0', 3.0);
   multiply
------------------
            27
(1 row)
```

- ln(x)

  Description: Natural logarithm.

  Return type: same as the input.

  Example:

  ```
  gaussdb=# SELECT ln(2.0);
         ln
  -------------------
   .6931471805599453
  (1 row)
  ```

- log(x)

  Description: Logarithm with 10 as the base.

  Return type: same as the input.

  Example:

  ```
  gaussdb=# SELECT log(100.0);
         log
  --------------------
   2.0000000000000000
  (1 row)
  ```

- log(b numeric, x numeric)

  Description: Logarithm with b as the base.

  Return type: numeric

  Example:

  ```
  gaussdb=# SELECT log(2.0, 64.0);
         log
  --------------------
   6.0000000000000000
  (1 row)
  ```

- log2(x)

  Description: Logarithm with 2 as the base.

  Return type: double precision

  Example:

  ```
  gaussdb=# SELECT log2(2);
   log2
  ------
     1
  (1 row)
  ```

  NOTE

  This function takes effect when the database parameter **sql_compatibility** is set to **'MYSQL'**.

- log10(x)

  Description: Logarithm with 10 as the base.

  Return type: double precision

  Example:

  ```
  gaussdb=# SELECT log10(10);
   log10
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-------
    1
(1 row)
```

> **NOTE**
>
> This function takes effect when the database parameter **sql_compatibility** is set to **'MYSQL'**.

- mod(x, y)

  Description: Returns the remainder (MOD) of **x**/**y**. If **x** is **0**, the function will return **y**.

  Return type: If the input parameter is of the (text, int) type, the return value is of the NUMERIC type. If the input parameter is of other types, the return type is the same as the parameter type.

  Example:

```
gaussdb=# SELECT mod(9,4);
 mod
-----
   1
(1 row)
gaussdb=# SELECT mod(9,0);
 mod
-----
   9
(1 row)
```

> **NOTE**
>
> In ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s6**, the mod function supports overloading with mod(text, int). When the mod function is called with (text, int) as the input type, mod(text, int) will take precedence.

```
gaussdb=# SET a_format_version='10c';
SET
gaussdb=# SET a_format_dev_version='s6';
SET
gaussdb=# SELECT mod('170141183460469231731687303715884105728', 7);
 mod
-----
   2
(1 row)
```

- pi()

  Description: π constant value.

  Return type: double precision

  Example:

```
gaussdb=# SELECT pi();
        pi
------------------
 3.14159265358979
(1 row)
```

- power(a double precision or numeric, b double precision or numeric)

  Description: b power of a.

  Return type: double precision or numeric. If implicit type conversion is not considered and both input parameters are of the numeric type, the numeric type is returned. If either input parameter is of the double precision type, the double precision type is returned.

  Example:

```
gaussdb=# SELECT power(9.0, 3.0);
       power
---------------------
 729.0000000000000000
(1 row)
```

- remainder(x, y)

  Description: Returns the remainder of **x**/**y**. If **y** is **0**, the function will report an error.

  Return type: same as the input (float4, float8, or numeric).

  Example:

```
gaussdb=# SELECT remainder(11,4);
 remainder
-----------
        -1
(1 row)
gaussdb=# SELECT remainder(9,0);
ERROR: division by zero
CONTEXT:  referenced column: remainder
```

  NOTE

  This function is valid only when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s2** in an ORA-compatible database.

- round(x)

  Description: Integer closest to the input parameter.

  Return type: same as the input (double precision or numeric).

  Example:

```
gaussdb=# SELECT round(42.4);
 round
-------
    42
(1 row)

gaussdb=# SELECT round(42.6);
 round
-------
    43
(1 row)
```

  NOTE

  The output of the float/double type may be **-0**. This also occurs in functions such as trunc and ceil. If the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s1** in an ORA-compatible database, the returned result is **0**. The following is an example:

```
gaussdb=# SELECT round(-0.2::float8);
 round
-------
    -0
(1 row)
```

- round(v numeric, s int)

  Description: The number of *s* digits are kept after the decimal point, and the number is rounded off.

  Return type: numeric

  Example:

```
gaussdb=# SELECT round(42.4382, 2);
 round
-------
```

```
 42.44
(1 row)
```

NOTE

- When the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s1** in an ORA-compatible database, **s** is truncated to an integer. Otherwise, **s** is rounded off to an integer.
- If the value of **a_format_version** is **10c** and the value of **a_format_dev_version** is **s1**, the round function supports round(timestamp, text) overloading. When **(text, text)** or **(text, '')** is used as the input parameter to call the round function, round (timestamp, text) is preferred.

- setseed(dp)

  Description: Sets a seed, within the range of [–1.0,1.0], for future calls to the random() function.

  Return type: void

  Example:

  ```
  gaussdb=# SELECT setseed(0.54823);
   setseed
  ---------

  (1 row)
  ```

  NOTE

  You are advised not to use this function in security scenarios.

- sign(x)

  Description: Returns symbols of this parameter.

  Return type: **–1** indicates negative numbers. **0** indicates 0, and **1** indicates positive numbers.

  Example:

  ```
  gaussdb=# SELECT sign(-8.4);
   sign
  ------
    -1
  (1 row)
  ```

- sin(x)

  Description: Sine

  Return type: double precision

  Example:

  ```
  gaussdb=# SELECT sin(1.57079);
        sin
  ------------------
   .999999999979986
  (1 row)
  ```

- sinh(x)

  Description: Hyperbolic sine

  Return type: dp or numeric. If implicit type conversion is not considered, the return type is the same as the input type.

  Example:

  ```
  gaussdb=# SELECT sinh(4);
        sinh
  ------------------
   27.2899171971277
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> This function is valid only when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s2** in an ORA-compatible database.

- sqrt(x)

  Description: Square root

  Return type: same as the input.

  Example:

```
gaussdb=# SELECT sqrt(2.0);
       sqrt
-------------------
 1.414213562373095
(1 row)
```

> **NOTE**
>
> This function uses the Karatsuba sqrt square root algorithm when the GUC parameter **gs_format_behavior_compat_options** is set to **sqrt_karatsuba**. Otherwise, this function uses the Newton iteration algorithm. The performance of the Karatsuba sqrt square root algorithm is better. In a few scenarios, the precision of the Karatsuba sqrt square root algorithm is different from that of the Newton iteration algorithm.

- tan(x)

  Description: Tangent

  Return type: double precision

  Example:

```
gaussdb=# SELECT tan(20);
       tan
------------------
 2.23716094422474
(1 row)
```

- tanh(x)

  Description: Hyperbolic tangent

  Return type: same as the input (double precision or numeric).

  Example:

```
gaussdb=# SELECT tanh(0.1);
                       tanh
-------------------------------------------
0.099667994624955817118305083678352183538 9
(1 row)
```

> **NOTE**
>
> This function is valid only when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s2** in an ORA-compatible database.

- trunc(x)

  Description: Truncation (the integer part is retained).

  Return type: same as the input.

  Example:

```
gaussdb=# SELECT trunc(42.8);
 trunc
-------
    42
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- trunc(v numeric, s int)

  Description: Truncates the last *s* digits after a decimal point.

  Return type: numeric

  Example:

  ```
  gaussdb=# SELECT trunc(42.4382, 2);
   trunc
  -------
   42.43
  (1 row)
  ```

  NOTE

  This function is valid only when **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1** in ORA-compatible mode. If the value of **s** is a decimal, the value is truncated instead of being rounded off.

- width_bucket(op numeric, b1 numeric, b2 numeric, count int)

  Description: Returns the SN of the bucket to which the operand would be assigned in the range of [b1,b2], which is divided into *count* equal-sized intervals based on the principle of equal interval bucketing.

  Return type: int.

  Example:

  ```
  gaussdb=# SELECT width_bucket(5.35, 0.024, 10.06, 5);
   width_bucket
  --------------
             3
  (1 row)
  ```

- width_bucket(op dp, b1 dp, b2 dp, count int)

  Description: Returns the SN of the bucket to which the operand would be assigned in the range of [b1,b2], which is divided into *count* equal-sized intervals based on the principle of equal interval bucketing.

  Return type: int.

  Example:

  ```
  gaussdb=# SELECT width_bucket(5.35, 0.024, 10.06, 5);
   width_bucket
  --------------
             3
  (1 row)
  ```

- smgrne(a smgr, b smgr)

  Description: Compares two integers of the smgr type to check whether they are different.

  Return type: Boolean

- smgreq(a smgr, b smgr)

  Description: Compares two integers of the smgr type to check whether they are equivalent.

  Return type: Boolean

- int1abs(tinyint)

  Description: Returns the absolute value of data of the uint8 type.

  Return type: tinyint

- int1and(tinyint, tinyint)

  Description: Returns the bitwise AND result of two data records of the uint8 type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: tinyint

- int1cmp(tinyint, tinyint)

  Description: Returns the comparison result of two data records of the uint8 type. If the value of the first parameter is greater, **1** is returned. If the value of the second parameter is greater, **-1** is returned. If they are the same, **0** is returned.

  Return type: integer

- int1div(tinyint, tinyint)

  Description: Returns the result of dividing two data records of the uint8 type. The result is of the tinyint type.

  Return type: tinyint

- int1eq(tinyint, tinyint)

  Description: Compares two pieces of data of the uint8 type to check whether they are the same.

  Return type: Boolean

- int1ge(tinyint, tinyint)

  Description: Determines whether the value of the first parameter is greater than or equal to the value of the second parameter in two data records of the uint8 type.

  Return type: Boolean

- int1gt(tinyint, tinyint)

  Description: Checks whether an unsigned 1-byte integer is greater than the other.

  Return type: Boolean

- int1larger(tinyint, tinyint)

  Description: Returns the larger value of unsigned one-byte integers.

  Return type: tinyint

- int1le(tinyint, tinyint)

  Description: Checks whether an unsigned 1-byte integer is less than or equal to the other.

  Return type: Boolean

- int1lt(tinyint, tinyint)

  Description: Checks whether an unsigned 1-byte integer is less than the other.

  Return type: Boolean

- int1smaller(tinyint, tinyint)

  Description: Returns the smaller of two unsigned one-byte integers.

  Return type: tinyint

- int1inc(tinyint)

  Description: Unsigned 1-byte integer plus 1.

  Return type: tinyint

- int1mi(tinyint, tinyint)

  Description: Performs a minus operation on an unsigned 1-byte integer.

  Return type: tinyint

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- int1mod(tinyint, tinyint)

  Description: Performs a MOD operation on unsigned 1-byte integers.

  Return type: tinyint

- int1mul(tinyint, tinyint)

  Description: Performs a multiplication operation on an unsigned 1-byte integer.

  Return type: tinyint

- int1ne(tinyint, tinyint)

  Description: Checks whether an unsigned 1-byte integer is not equal to the other.

  Return type: Boolean

- int1pl(tinyint, tinyint)

  Description: Performs an addition operation on an unsigned 1-byte integer.

  Return type: tinyint

- int1um(tinyint)

  Description: Returns an unsigned 2-byte integer after subtracting the opposite number from the unsigned 1-byte integer.

  Return type: smallint

- int1xor(tinyint, tinyint)

  Description: Performs an XOR operation on an unsigned 1-byte integer.

  Return type: tinyint

- cash_div_int1(money, tinyint)

  Description: Performs a division operation on the money type.

  Return type: money

- cash_mul_int1(money, tinyint)

  Description: Performs a multiplication operation on the money type.

  Return type: money

- int1not(tinyint)

  Description: Reverts binary bits of an unsigned 1-byte integer.

  Return type: tinyint

- int1or(tinyint, tinyint)

  Description: Performs an OR operation on an unsigned 1-byte integer.

  Return type: tinyint

- int1shl(tinyint, integer)

  Description: Shifts an unsigned 1-byte integer leftwards by a specified number of bits.

  Return type: tinyint

- int1shr(tinyint, integer)

  Description: Shifts an unsigned 1-byte integer rightwards by a specified number of bits.

  Return type: tinyint

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- analyze_tgtype_for_type(n smallint)

  Description: Parses **pg_trigger.tgtype**, parses *n* by bit, and returns one of **before each row**, **after each row**, **before statement**, **after statement**, and **instead of**.

  Return type: varchar2(16)

- analyze_tgtype_for_event(n smallint)

  Description: Parses **pg_trigger.tgtype**, parses *n* by bit, and returns one or more of **insert**, **update**, **delete**, and **truncate**.

  Return type: varchar2(246)

- nanvl(n2, n1)

  Description: Two parameters are entered. The parameters must be of the numeric type or a non-numeric type that can be implicitly converted to the numeric type. If the first parameter **n2** is NaN, **n1** is returned. Otherwise, **n2** is returned.

  Return type: double precision > float4 > numeric, depending on the higher priority between the types of the input parameters.

  Example:

  ```
  gaussdb=# SELECT nanvl('NaN', 1.1);
   nanvl
  -------
    1.1
  (1 row)
  ```

  NOTE

  This function is valid only when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s2** in an ORA-compatible database.

- numeric_eq_text(numeric, text)

  Description: Checks whether the numeric variable is equal to the numeric value converted from the text variable.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT numeric_eq_text(1, '1');
   numeric_eq_text
  -----------------
   t
  (1 row)
  ```

- numeric_ne_text(numeric, text)

  Description: Checks whether the numeric variable is not equal to the numeric value converted from the text variable.

  Return type: Boolean

- numeric_gt_text(numeric, text)

  Description: Checks whether the numeric variable is greater than the numeric value converted from the text variable.

  Return type: Boolean

- numeric_ge_text(numeric, text)

  Description: Checks whether the numeric variable is greater than or equal to the numeric value converted from the text variable.

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- numeric_lt_text(numeric, text)

  Description: Checks whether the numeric variable is smaller than the numeric value converted from the text variable.

  Return type: Boolean

- numeric_le_text(numeric, text)

  Description: Checks whether the numeric variable is less than or equal to the numeric value converted from the text variable.

  Return type: Boolean

- text_eq_numeric(text, numeric)

  Description: Checks whether the numeric value converted from a text variable is equal to the numeric variable.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT text_eq_numeric('1', 1);
   text_eq_numeric
  -----------------
   t
  (1 row)
  ```

- text_ne_numeric(text, numeric)

  Description: Checks whether the numeric value converted from a text variable is not equal to the numeric variable.

  Return type: Boolean

- text_gt_numeric(text, numeric)

  Description: Checks whether the numeric value converted from a text variable is greater than the numeric variable.

  Return type: Boolean

- text_ge_numeric(text, numeric)

  Description: Checks whether the numeric value converted from a text variable is greater than or equal to the numeric variable.

  Return type: Boolean

- text_lt_numeric(text, numeric)

  Description: Checks whether the numeric value converted from a text variable is less than the numeric variable.

  Return type: Boolean

- text_le_numeric(text, numeric)

  Description: Checks whether the numeric value converted from a text variable is less than or equal to the numeric variable.

  Return type: Boolean

- bigint_eq_text(bigint, text)

  Description: Checks whether the bigint variable is equal to the bigint value converted from a text variable.

  Return type: Boolean

  ```
  gaussdb=# SELECT bigint_eq_text(1, '1');
   bigint_eq_text
  ----------------
   t
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- bigint_ne_text(bigint, text)

  Description: Checks whether the bigint variable is not equal to the bigint value converted from a text variable.

  Return type: Boolean

- bigint_gt_text(bigint, text)

  Description: Checks whether the bigint variable is greater than the bigint value converted from a text variable.

  Return type: Boolean

- bigint_ge_text(bigint, text)

  Description: Checks whether the bigint variable is greater than or equal to the bigint value converted from a text variable.

  Return type: Boolean

- bigint_lt_text(bigint, text)

  Description: Checks whether the bigint variable is smaller than the bigint value converted from a text variable.

  Return type: Boolean

- bigint_le_text(bigint, text)

  Description: Checks whether the bigint variable is less than or equal to the bigint value converted from a text variable.

  Return type: Boolean

- text_eq_bigint(text, bigint)

  Description: Checks whether the bigint value converted from a text variable is equal to the bigint variable.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT text_eq_bigint('1', 1);
   text_eq_bigint
  ----------------
   t
  (1 row)
  ```

- text_ne_bigint(text, bigint)

  Description: Checks whether the bigint value converted from a text variable is not equal to the bigint variable.

  Return type: Boolean

- text_gt_bigint(text, bigint)

  Description: Checks whether the bigint value converted from a text variable is greater than the bigint variable.

  Return type: Boolean

- text_ge_bigint(text, bigint)

  Description: Checks whether the bigint value converted from a text variable is greater than or equal to the bigint variable.

  Return type: Boolean

- text_lt_bigint(text, bigint)

  Description: Checks whether the bigint value converted from a text variable is less than the bigint variable.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: Boolean

- text_le_bigint(text, bigint)

  Description: Checks whether the bigint value converted from a text variable is less than or equal to the bigint variable.

  Return type: Boolean

1.6.8 Date and Time Processing Functions and Operators

Time and Date Operators

Table 1-48 describes the time and date operators.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

When working with time and date operators, use explicit type prefixes to modify operands. This will help the database parse the operands correctly according to user expectations and avoid any unexpected results.

Example: Not specifying a data type results in an error.

```
gaussdb=# SELECT date '2001-10-01' - '7' AS RESULT;
ERROR:
GAUSS-10416: invalid input syntax for type timestamp: "7"
SQLSTATE: 22007
LINE 1: SELECT date '2001-10-01' - '7' AS RESULT;
                                   ^
CONTEXT:  referenced column: result
```

In Oracle-compatible mode, the DATEA type supports arithmetic operations with strings of the unspecified types, including integer strings or floating-point strings. The return type of the operation result is DATEA. Only the plus sign (+) and minus sign (-) are supported.

```
gaussdb=# SET nls_date_format='yyyy-mm-dd hh24:mi:ss';
SET
gaussdb=# SELECT '2024-1-2'::DATEA + '0.00000579';
       ?column?
------------------------
 2024-01-02 00:00:01 AD
(1 row)

gaussdb=# SELECT '0.00000579' + '2024-1-2'::DATEA;
       ?column?
------------------------
 2024-01-02 00:00:01 AD
(1 row)

gaussdb=# SELECT '2024-1-2'::DATEA + '1';
       ?column?
------------------------
 2024-01-03 00:00:00 AD
(1 row)

gaussdb=# SELECT '2024-1-2'::DATEA - '0.00000579';
       ?column?
------------------------
 2024-01-01 23:59:59 AD
(1 row)

gaussdb=# SELECT '2' - '2024-1-2'::DATEA;
ERROR:  Invalid data exists in a string for matching date.
LINE 1: SELECT '2' - '2024-1-2'::DATEA;
               ^
```

In ORA-compatible mode, the DATEA type can be subtracted from an unspecified type (complying with the time format).

```
gaussdb=# SET nls_date_format='yyyy-mm-dd hh24:mi:ss';
SET
gaussdb=# SELECT '2024-1-3' - '2024-1-2'::DATEA;
 ?column?
----------
        1
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-48 Time and date operators

Operator	Example
+	gaussdb=# SELECT date '2001-9-28' + integer '7' AS RESULT; result ------------ 2001-10-05 (1 row) **NOTE** In ORA-compatible mode, the query result is 2001-10-05 00:00:00.
	gaussdb=# SELECT date '2001-09-28' + interval '1 hour' AS RESULT; result --------------------- 2001-09-28 01:00:00 (1 row)
	gaussdb=# SELECT date '2001-09-28' + time '03:00' AS RESULT; result --------------------- 2001-09-28 03:00:00 (1 row)
	gaussdb=# SELECT interval '1 day' + interval '1 hour' AS RESULT; result ---------------- 1 day 01:00:00 (1 row)
	gaussdb=# SELECT timestamp '2001-09-28 01:00' + interval '23 hours' AS RESULT; result --------------------- 2001-09-29 00:00:00 (1 row)
	gaussdb=# SELECT time '01:00' + interval '3 hours' AS RESULT; result ---------- 04:00:00 (1 row)
	gaussdb=# SET nls_date_format='yyyy-mm-dd hh24:mi:ss'; SET gaussdb=# SELECT '2024-1-2'::DATEA - '2024-1-1'::DATEA AS RESULT; result -------- 1 (1 row)
-	gaussdb=# SELECT date '2001-10-01' - date '2001-09-28' AS RESULT; result -------- 3 (1 row)
	gaussdb=# SELECT date '2001-10-01' - integer '7' AS RESULT; result --------------------- 2001-09-24 00:00:00 (1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operator	Example
	gaussdb=# SELECT date '2001-09-28' - interval '1 hour' AS RESULT; result --------------------- 2001-09-27 23:00:00 (1 row)
	gaussdb=# SELECT time '05:00' - time '03:00' AS RESULT; result ---------- 02:00:00 (1 row)
	gaussdb=# SELECT time '05:00' - interval '2 hours' AS RESULT; result ---------- 03:00:00 (1 row)
	gaussdb=# SELECT timestamp '2001-09-28 23:00' - interval '23 hours' AS RESULT; result --------------------- 2001-09-28 00:00:00 (1 row)
	gaussdb=# SELECT interval '1 day' - interval '1 hour' AS RESULT; result ---------- 23:00:00 (1 row)
	gaussdb=# SELECT timestamp '2001-09-29 03:00' - timestamp '2001-09-27 12:00' AS RESULT; result ---------------- 1 day 15:00:00 (1 row)
	gaussdb=# SET nls_date_format='yyyy-mm-dd hh24:mi:ss'; SET gaussdb=# SELECT '2024-1-2'::DATEA - '2024-1-1'::DATEA AS RESULT; result -------- 1 (1 row) **NOTE** In the scenario where a DATEA type is subtracted from another DATEA type, only 15 valid digits are supported due to the precision issues with the double type. In addition, the 15th digit may be rounded incorrectly due to precision issues with the 16th digit. Example: SELECT '2020-10-02 00:00:00'::datea - '2020-10-01 00:00:34'::datea; The manually calculated result should be: 0.99960648148148148... However, due to the precision issues with double, the actual calculated result becomes 0.999606481481482, where rounding occurs at the 16th digit.
*	gaussdb=# SELECT 900 * interval '1 second' AS RESULT; result ---------- 00:15:00 (1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operator	Example
	gaussdb=# SELECT 21 * interval '1 day' AS RESULT; result --------- 21 days (1 row)
	gaussdb=# SELECT double precision '3.5' * interval '1 hour' AS RESULT; result ---------- 03:30:00 (1 row)
/	gaussdb=# SELECT interval '1 hour' / double precision '1.5' AS RESULT; result ---------- 00:40:00 (1 row)

NOTE

In ORA-compatible mode, the following functions are supported:

- The left operand of the DATEA type can be added to the right operand of the TIME WITHOUT TIME ZONE, INTERVAL DAY TO SECOND, INTERVAL DAY, or RELTIME type. The left and right operands can be exchanged.
- The left operand of the DATEA type can be subtracted from the right operand of the TIME WITHOUT TIME ZONE, TIMESTAMP WITH TIME ZONE, INTERVAL DAY TO SECOND, INTERVAL DAY, RELTIME, SMALLDATETIME, or ABSTIME type.
- The left operand of the TIMESTAMP WITHOUT TIME ZONE, TIMESTAMP WITH TIME ZONE, SMALLDATETIME, or ABSTIME type can be subtracted from the right operand of the DATEA type.

Time and Date Functions

- age(timestamp, timestamp)

  Description: Subtracts parameters, producing a result in YYYY-MM-DD format. If the result is negative, the returned result is also negative. The input parameters may or may not contain time zones.

  Return type: interval

  Example:

  ```
  gaussdb=# SELECT age(timestamp '2001-04-10', timestamp '1957-06-13');
           age
  --------------------------
   43 years 9 mons 27 days
  (1 row)
  ```

- age(timestamp)

  Description: Subtracts the parameter value from the system time when the current SQL statement starts to be executed. The input parameters can either both include timezone information or both exclude it.

  Return type: interval

  Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT age(timestamp '1957-06-13');
          age
------------------------
 60 years 2 mons 18 days
(1 row)
```

- clock_timestamp()

  Description: Returns the timestamp of the system time when the current function is called. This volatile function obtains the latest timestamp for each scan. Therefore, the result of each function call in a query is different.

  Return type: timestamp with time zone

  Example:

```
gaussdb=# SELECT clock_timestamp();
       clock_timestamp
-------------------------------
 2017-09-01 16:57:36.636205+08
(1 row)
```

- current_date

  Description: Returns the system date when the current SQL statement starts.

  Return type: date

  Example:

```
gaussdb=# SELECT current_date;
    date
------------
 2017-09-01
(1 row)
-- When the GUC parameter a_format_date_timestamp is enabled in ORA-compatible mode:
gaussdb=# SET a_format_date_timestamp=on;
SET
gaussdb=# SELECT current_date;
    current_date
---------------------
 2023-11-24 11:25:09
(1 row)
-- In ORA-compatible mode, if the GUC parameter mapping_date_to_datea is enabled, the return type is datea.
gaussdb=# SELECT current_date;
      current_date
-------------------------
 2025-05-21 11:31:15 AD
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

When **sql_compatibility** is **'ORA'**, this function operates as follows:

- If the GUC parameter **mapping_date_to_datea** is set to **on**, the timestamp of the system time when the current SQL statement is started is returned. The return value type is datea, and the column name is **current_date**.
- If the GUC parameter **mapping_date_to_datea** is set to **off**, the following rules are followed:
  - If the GUC parameter **a_format_date_timestamp** is set to **on**, it returns a timestamp that shows the system time when the current SQL statement starts. The return type is timestamp without time zone, accurate to seconds, and the column name is **current_date**.
  - If the GUC parameter **a_format_date_timestamp** is **off**, it returns the system time when the transaction starts. If **a_format_version** is **10c** and **a_format_dev_version** is **s2**, the return type is timestamp without time zone, accurate to seconds, and the column name is **timestamp**. Otherwise, the return type is date, accurate to days, and the column name is **date**.

This function has the following behaviors when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7**, and **b_format_dev_version** is set to **s1**:

- current_date can be called with parentheses.
- The actually called function of current_date is curdate. You can run the **\df curdate** command to query the detailed input parameters and return values of the function.

- current_time()

  Description: Specifies the system time when the current transaction starts. If **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, the system time when the current SQL statement starts is returned.

  Return type: time with time zone. When **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, the return type is time without time zone.

  Example:

```
gaussdb=# SELECT current_time;
      timetz
--------------------
 16:58:07.086215+08
(1 row)
-- When the parameter is enabled in MySQL-compatible mode:
gaussdb_m=# SELECT current_time;
 current_time
--------------
 15:14:00
(1 row)
```

- current_time([precision])

  Description: Returns the system time when the current SQL statement starts.

  Parameter: **precision** indicates the precision (number of decimal places after the second). The value is of the int type and in the range [0,6]. The default value is **0**. If the precision is invalid, an error is reported.

  Return type: time without time zone

  Implementation mode: This function is mapped to the system function curtime.

  Example:

```
gaussdb_m=# SELECT current_time();
 current_time
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------------
 15:14:05
(1 row)
gaussdb_m=# SELECT current_time(3);
 current_time
--------------
 15:14:08.433
(1 row)
```

> **NOTE**
>
> This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**. The actually called function of current_time is curtime. You can run the **\df curtime** command to query the detailed input parameters and return values of the function.

- current_timestamp

  Description: Specifies the current date and time. This is a statement-level timestamp. The returned results within the same statement remain unchanged.

  Return type: timestamp without time zone when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, or timestamp with time zone in other scenarios

  Example:

```
gaussdb=# SELECT current_timestamp;
       pg_systimestamp
-------------------------------
 2017-09-01 16:58:19.22173+08
(1 row)

-- When the version is 5.7 in MySQL-compatible database:
gaussdb_m=# SELECT current_timestamp;
     timestamp
---------------------
 2023-08-21 15:08:24
(1 row)
-- When the GUC parameter a_format_date_timestamp is enabled in ORA-compatible mode:
gaussdb=# SET a_format_date_timestamp=on;
SET
gaussdb=# SELECT current_timestamp;
       current_timestamp
-------------------------------
 2023-11-24 11:31:04.895312+08
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When **sql_compatibility** is set to **'ORA'**, or **sql_compatibility** is set to **'MYSQL'** and the **b_format_version** and **b_format_dev_version** parameters are not set, the return result is the timestamp of the system time when the latest SQL statement starts to be executed, and the return type is timestamp with time zone. If the GUC parameter **a_format_date_timestamp** is set to **on**, the column name is **current_timestamp**.
- In PL/SQL, a simple assignment statement, such as time1 := current_timestamp, is considered as an expression. Therefore, the return result is the timestamp of the system time when the latest SQL statement is started.
- This function has the following behaviors when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7**, and **b_format_dev_version** is set to **s1**:
  - The return type is timestamp without time zone.
  - The precision of the returned result is 0.
  - The timestamp of the system time is returned when the current SQL statement starts.
  - This function is implemented through TYPE conversion and has no registered function. Therefore, you can run the **\df+** command of gsql to view the function information in other compatible modes, not the function information in version 5.7 in the MySQL-compatible mode.
  - When the function is called multiple times in an anonymous block, the entire anonymous block is considered as a statement. Therefore, the same result is returned each time the function is called.

- current_timestamp()

  Description: Returns the system time when the current SQL statement starts.

  Return type: timestamp without time zone

  Example:

```
gaussdb=# SELECT current_timestamp();
     timestamp
---------------------
 2023-08-21 14:34:30
(1 row)
```

  NOTE

  This function can be used only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**. In addition, this function has the following behavior:

  - The return type is timestamp without time zone.
  - The precision of the returned result is 0.
  - The timestamp of the system time is returned when the current SQL statement starts to be executed.
  - This function is implemented through TYPE conversion and has no registered function. Therefore, you can run the **\df+** command of gsql to view the function information in other modes, not the function information in version 5.7 in the MySQL-compatible mode.

- current_timestamp(precision)

  Description: Obtains the current date and time, and rounds the microseconds of the result to the specified decimal place.

  Parameter: **precision** (number of decimal places after the second). The precision is of the int type, within the range of [0,6] and with a default value of **0**. Setting a value greater than 6 will trigger an alarm and return a time

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

value with the precision of 6. Using any other invalid precision will trigger an error.

Return type: timestamp without time zone when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, or timestamp with time zone in other scenarios

Example:

```
gaussdb=# SELECT current_timestamp(1);
      timestamptz
-------------------------------
 2017-09-01 16:58:19.2+08
(1 row)

-- When the version is 5.7 in MySQL-compatible database:
gaussdb_m=# SELECT current_timestamp(1);
      timestamp
-----------------------
 2023-08-21 15:09:35.3
(1 row)

-- When the GUC parameter a_format_date_timestamp is enabled in ORA-compatible mode:
gaussdb=# SET a_format_date_timestamp=on;
SET
gaussdb=# SELECT current_timestamp(6);
      current_timestamp
-------------------------------
 2023-11-24 11:35:57.268592+08
(1 row)
-- If a_format_version is set to 10c and a_format_dev_version is set to s2 in an ORA-compatible database, precision can be an integer of the numeric type.
gaussdb=# SET a_format_version='10c';
SET
gaussdb=# SET a_format_dev_version='s2';
SET
gaussdb=# SELECT current_timestamp(6.0);
     current_timestamp
-------------------------------
 2023-11-25 11:14:48.74763+08
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- Zeros at the end of microseconds are not displayed. For example, 2017-09-01 10:32:19.212000 is displayed as 2017-09-01 10:32:19.212.
- This function has the following behaviors when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7**, and **b_format_dev_version** is set to **s1**:
  - The return type is timestamp without time zone.
  - The timestamp of the system time is returned when the current SQL statement starts to be executed.
  - This function is implemented through TYPE conversion and has no registered function. Therefore, you can run the **\df+** command of gsql to view the function information in other modes, not the function information in version 5.7 in the MySQL-compatible mode.
- This function has the following behaviors when **sql_compatibility** is set to **'ORA'** and the GUC parameter **a_format_date_timestamp** is set to **on**:
  - The return value type is timestamp with time zone, and the column name is **current_timestamp**.
  - The timestamp of the system time is returned when the current SQL statement starts to be executed.
  - When **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s2**, the **precision** parameter can be an integer of the numeric type. Otherwise, only the int type is supported.
- When **sql_compatibility** is set to **'ORA'** and the GUC parameter **a_format_date_timestamp** is set to **off**, if the input parameter is an integer without a decimal point, the system date and time when the transaction starts is returned; if the input parameter is an integer with a decimal point, the timestamp of the system time when the current SQL execution starts is returned.

- pg_systimestamp()

  Description: Current date and time (start of the current statement).

  Return type: timestamp with time zone

  Example:

```
gaussdb=# SELECT pg_systimestamp();
       pg_systimestamp
-------------------------------
 2015-10-14 11:21:28.317367+08
(1 row)
```

- systimestamp()

  Description: Gets the date and time of the current system.

  Return type: timestamp with time zone

  Example:

```
gaussdb=# SELECT systimestamp();
        systimestamp
-------------------------------
 2024-04-16 15:55:21.614342+08
(1 row)
-- After the behavior_compat_options parameter is set to sys_function_without_brackets, calling without parentheses is supported.
gaussdb=# SET behavior_compat_options='sys_function_without_brackets';
gaussdb=# SELECT systimestamp;
        systimestamp
-------------------------------
 2024-04-16 15:55:21.614342+08
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> This function takes effect only when **sql_compatibility** is set to **ORA**.

- systimestamp(precision)

  Description: Obtains the date and time of the current system and displays the microsecond part of the result based on the specified precision.

  Parameter: precision (number of decimal places after the second). INT type. Range: [0,6]. The default value is **6**. When the value exceeds 6, a warning message is displayed, the time value is output with precision of 6.

  Return type: timestamp with time zone

  Example:

```
gaussdb=# SELECT systimestamp(6);
        systimestamp
-------------------------------
 2024-04-16 15:55:21.614342+08
(1 row)
gaussdb=# SELECT systimestamp(9);
WARNING:  TIMESTAMP(9) WITH TIME ZONE precision reduced to maximum allowed, 6
CONTEXT:  referenced column: systimestamp
        systimestamp
-------------------------------
 2024-04-16 15:59:49.492505+08
(1 row)
gaussdb=# SELECT systimestamp(5.0);
        systimestamp
-------------------------------
 2024-04-16 16:00:23.48079+08
(1 row)
```

> **NOTE**
>
> This function takes effect only when **sql_compatibility** is set to **ORA**.

- date_part(text, timestamp)

  Description: Retrieves fields such as year or hour from date/time values. It is equivalent to **extract(field from timestamp)**.

  Parameter: **timestamp** of the abstime, date, interval, reltime, time with time zone, time without time zone, timestamp with time zone, timestamp without time zone, datea type (valid only in Oracle-compatible mode).

  Return type: double precision

  Example:

```
gaussdb=# SELECT date_part('hour', timestamp '2001-02-16 20:38:40');
 date_part
-----------
        20
(1 row)
```

- date_part(text, interval)

  Description: Obtains the month. If the value is greater than 12, obtain the remainder after it is divided by 12. It is equivalent to **extract(field from timestamp)**.

  Return type: double precision

  Example:

```
gaussdb=# SELECT date_part('month', interval '2 years 3 months');
 date_part
-----------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
       3
(1 row)
```

- date_trunc(text, timestamp)

  Description: Truncates to the precision specified by **text**.

  Return type: interval, timestamp with time zone, timestamp without time zone

  Example:

```
gaussdb=# SELECT date_trunc('hour', timestamp  '2001-02-16 20:38:40');
     date_trunc
---------------------
 2001-02-16 20:00:00
(1 row)
```

- trunc(timestamp)

  Description: Truncates to day by default.

  Parameter: **timestamp** can be set to **timestamp with time zone**, **timestamp without time zone**, or **datea** (valid only in Oracle-compatible mode).

  Return type: timestamp without time zone, datea (valid only for ORA compatibility)

  Example:

```
gaussdb=# SELECT trunc(timestamp  '2001-02-16
20:38:40');
                      trunc
---------------------
2001-02-16 00:00:00
(1 row)

gaussdb=# SELECT trunc(timestamp with time zone '2001-02-16 20:38:40');
        trunc
---------------------
 2001-02-16 00:00:00
(1 row)
```

- trunc(arg1, arg2)

  Description: Truncates to the precision specified by **arg2**.

  Parameters:

  – Types of **arg1**: interval, timestamp with time zone, timestamp without time zone, and datea (effective only in ORA-compatible mode).

  – Type of **arg2**: text

  When **arg1** is of the datea type, the units supported by **arg2** are listed in **Table 1-49**.

  When **arg1** is of another supported type, if **sql_compatibility** is set to **'ORA'**, **a_format_version** is set to **10c**, and **a_format_dev_version** is set to **s1**, the units supported by **arg2** are listed in **Table 1-49**. In other cases, the units supported by **arg2** are listed in **Table 1-50**.

Table 1-49 Unit mapping table 1

Unit	Description
cc, scc	Century
syyyy, yyyy, year, syear, yyy, yy, y	Year

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit	Description
iyyy, iy, i	The year defined by the ISO 8601 standard, including the calendar week
q	Quarter
month, mon, mm, rm	Month
ww	The date in a given week that corresponds to the same day of the week as the first day of the year
iw	The week that matches the first day of a calendar week defined in the ISO 8601 standard
w	The date in a given week that corresponds to the same day of the week as the first day of the month
ddd, dd, j	Day
day, dy, d	Start date of a week (with Sunday as the first day of the week)
hh, hh12, hh24	Hour
mi	Minute

Table 1-50 Unit mapping table 2

Unit	Description
mil, mils, millennia, millennium	Millennium
c, cc, cent, centuries, century	Century
dec, decade, decades, decs	Decade
y, year, years, yr, yrs, yyyy	Year
q, qtr, quarter	Quarter
mm, mon, mons, month, months	Month
w, week, weeks	Week
j, d, day, days, dd, ddd	Day
h, hh, hour, hours, hr, hrs	Hour
m, mi, min, mins, minute, minutes	Minute
s, sec, second, seconds, secs	Second

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit	Description
millisecon, ms, msec, msecond, mseconds, msecs	Millisecond
microsecon, us, usec, usecond, useconds, usecs	Microsecond

Return types: interval, timestamp with time zone, timestamp without time zone, and datea (valid only in Oracle-compatible mode).

Example:

- When **arg1** is of the datea type:

```
-- The first Monday of 2022 is January 3.
gaussdb=# SELECT trunc(to_date('2022-12-30', 'YYYY-MM-
DD'),'iy');
                      trunc
-------------------------
 2022-01-03 00:00:00 AD
(1 row)

-- January 1, 2022 is Saturday. Therefore, the Saturday preceding the following input date is returned:
gaussdb=# SELECT trunc(to_date('2022-12-30', 'YYYY-MM-DD'),'ww');
trunc
-------------------------
 2022-12-24 00:00:00 AD
(1 row)

-- In the week of December 30, 2022, the start date (Monday) of the week is December 26.
gaussdb=# SELECT trunc(to_date('2022-12-30', 'YYYY-MM-DD'),'iw');
trunc
-------------------------
 2022-12-26 00:00:00 AD
(1 row)

-- December 1, 2022 is Thursday. Therefore, the Thursday preceding the following input date is returned:
gaussdb=# SELECT trunc(to_date('2022-12-30', 'YYYY-MM-DD'),'w');
trunc
-------------------------
 2022-12-29 00:00:00 AD
(1 row)

-- In the week of December 30, 2022, the start date (Sunday) of the week is December 25.
gaussdb=# SELECT trunc(to_date('2022-12-30', 'YYYY-MM-DD'),'day');
trunc
-------------------------
 2022-12-25 00:00:00 AD
(1 row)
```

- When **arg1** is of the other supported type:

```
gaussdb=# SELECT trunc(timestamp  '2001-02-16 20:38:40',
'hour');
                      trunc
---------------------
2001-02-16 20:00:00
(1 row)
```

- round(arg1, arg2)

  Description: Rounds off to the precision specified by **arg2**.

  Parameters:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Types of **arg1**: timestamp without time zone, datea (valid only for ORA compatibility)
  - Type of **arg2**: text

  Return type: timestamp without time zone, datea (valid only for ORA compatibility)

  Example:

```
gaussdb=# SELECT round(timestamp  '2001-02-16 20:38:40',
'hour');
                round
---------------------
2001-02-16 21:00:00
(1 row)
-- When the GUC parameter mapping_date_to_datea is enabled in ORA-compatible mode:
gaussdb=# SET nls_date_format = 'YYYY-MM-DD HH24:MI:SS';
gaussdb=# SELECT round(datea '2001-02-16 20:38:40', 'yyyy');
        round
------------------------
 2001-01-01 00:00:00 AD
(1 row)
```

  NOTE

  - This function is valid only when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s1** in an ORA-compatible database.
  - When the input parameter **arg1** is of the datea type, units supported by **arg2** are specified in the trunc function. For details, see **Table 1-49**.

- daterange(arg1, arg2)

  Description: Obtains time boundary information.

  Parameters: **arg1** and **arg2** are of the date type.

  Return type: daterange

  Example:

```
gaussdb=# SELECT daterange('2000-05-06','2000-08-08');
       daterange
-------------------------
 [2000-05-06,2000-08-08)
(1 row)
```

  NOTE

  In ORA-compatible mode:

  - After the **mapping_date_to_datea** parameter is enabled, the DATE type is mapped to the DATEA type. The daterange function does not support the DATEA type as an input parameter.
  - After the **mapping_date_to_datea** parameter is disabled, the DATE type is mapped to the TIMESTAMP(0) WITHOUT TIME ZONE type. The daterange function does not support the TIMESTAMP(0) WITHOUT TIME ZONE type as an input parameter.

- daterange(arg1, arg2, text)

  Description: Obtains time boundary information.

  Parameters:

  - **arg1**: start date of the time range. The type is date.
  - **arg2**: end date of the time range. The type is date.
  - **text**: string of boundary symbols. The type is text. The value must be one of the following: **()**, **(]**, **[)**, or **[]**. It is used to specify whether the boundary values are included in the range. **[** indicates that the start date

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

is included, **(** indicates that the start date is not included, **]** indicates that the end date is included, and **)** indicates that the end date is not included.

Return type: daterange

Example:

```
gaussdb=# SELECT daterange('2000-05-06','2000-08-08','[]');
       daterange
-------------------------
 [2000-05-06,2000-08-09)
(1 row)
```

NOTE

In ORA-compatible mode:

- After the **mapping_date_to_datea** parameter is enabled, the DATE type is mapped to the DATEA type. The daterange function does not support the DATEA type as an input parameter.
- After the **mapping_date_to_datea** parameter is disabled, the DATE type is mapped to the TIMESTAMP(0) WITHOUT TIME ZONE type. The daterange function does not support the TIMESTAMP(0) WITHOUT TIME ZONE type as an input parameter.

- isfinite(date)

  Description: Checks whether a date is a finite value. If the date is a finite value, **t** is returned. Otherwise, **f** is returned.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT isfinite(date '2001-02-16');
   isfinite
  ----------
   t
  (1 row)
  gaussdb=# SELECT isfinite(date 'infinity');
   isfinite
  ----------
   f
  (1 row)
  ```

- isfinite(timestamp)

  Description: Checks whether a timestamp is a finite value. If the timestamp is a finite value, **t** is returned. Otherwise, **f** is returned.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT isfinite(timestamp '2001-02-16 21:28:30');
   isfinite
  ----------
   t
  (1 row)
  gaussdb=# SELECT isfinite(timestamp 'infinity');
   isfinite
  ----------
   f
  (1 row)
  ```

- isfinite(interval)

  Description: Checks whether the interval is a finite value. If the interval is a finite value, **t** is returned. Currently, **f** cannot be returned. If **'infinity'** is entered, an error is reported.

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT isfinite(interval '4 hours');
 isfinite
----------
 t
(1 row)
```

- justify_days(interval)

  Description: Sets the time interval in months (30 days as a month).

  Return type: interval

  Example:

```
gaussdb=# SELECT justify_days(interval '35 days');
 justify_days
--------------
 1 mon 5 days
(1 row)
```

- justify_hours(interval)

  Description: Sets the time interval in days (24 hours as a day).

  Return type: interval

  Example:

```
gaussdb=# SELECT justify_hours(interval '27 hours');
 justify_hours
---------------
 1 day 03:00:00
(1 row)
```

- justify_interval(interval)

  Description: Adjusts **interval** using **justify_days** and **justify_hours**.

  Return type: interval

  Example:

```
gaussdb=# SELECT justify_interval(interval '1 mon -1 hour');
 justify_interval
------------------
 29 days 23:00:00
(1 row)
```

- localtime()

  Description: Specifies the system time when the current transaction starts. If **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7**, and **b_format_dev_version** is set to **s1**, the system date and time when the current SQL query execution starts is returned.

  Return type: time. If **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, a timestamp without time zone is returned.

  Example:

```
gaussdb=# SELECT localtime AS RESULT;
     result
----------------
 16:05:55.664681
(1 row)
-- When the parameter is enabled in MySQL-compatible mode:
gaussdb_m=# select localtime;
      localtime
---------------------
 2023-08-21 15:21:57
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- localtime[([precision])]

  Description: Returns the system date and time when the current SQL statement starts.

  Parameter: **precision** indicates the precision (number of decimal places after the second). The value is of the int type and in the range [0,6]. The default value is **0**. If the value is invalid, an error is reported.

  Return type: timestamp without time zone

  Implementation method: Register the system function localtime.

  Example:

```
gaussdb_m=# SELECT localtime();
     localtime
---------------------
 2023-08-21 15:23:49
(1 row)
gaussdb_m=# SELECT localtime(3);
       localtime
-------------------------
 2023-08-21 15:23:51.965
(1 row)
```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- localtimestamp

  Description: Specifies the current date and time.

  Return type: timestamp

  Example:

```
gaussdb=# SELECT localtimestamp;
        timestamp
----------------------------
 2017-09-01 17:03:30.781902
(1 row)
-- When the parameter is enabled in MySQL-compatible mode:
gaussdb_m=# SELECT localtimestamp;
     timestamp
---------------------
 2023-08-21 15:27:55
(1 row)
-- When the GUC parameter a_format_date_timestamp is enabled in ORA-compatible mode:
gaussdb=# SET a_format_date_timestamp=on;
SET
gaussdb=# SELECT localtimestamp;
      localtimestamp
----------------------------
 2023-11-24 11:38:25.633231
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This function has the following behaviors when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7**, and **b_format_dev_version** is set to **s1**:
  - It returns the system date and time that indicates when the current SQL statement starts.
  - The return type is timestamp without time zone, and the column name is **timestamp**.
- This function has the following behaviors when **sql_compatibility** is set to **'ORA'** and the GUC parameter **a_format_date_timestamp** is set to **on**:
  - The return value type is timestamp without time zone, and the column name is **localtimestamp**.
  - The timestamp of the system time is returned when the current SQL statement starts to be executed.
  - This prevents the optimizer from obtaining the constant time in advance. As a result, the obtained time is incorrect in the gplan scenario.
- If **sql_compatibility** is set to **'ORA'** and the GUC parameter **a_format_date_timestamp** is set to **off**, this function returns the system date and time when the transaction starts.

- localtimestamp[([precision])]

  Description: Specifies the current date and time.

  Parameter: **precision** (number of decimal places after the second). The precision is of the int type, within the range of [0,6] and with a default value of **0**. Setting a value greater than 6 will trigger an alarm and return a time value with the precision of 6. Using any other invalid precision will trigger an error.

  Return type: timestamp without time zone

  Example:

```
-- Calls with parentheses and without input parameters are supported only in MySQL-compatible mode.
gaussdb_m=# SELECT localtimestamp();
     timestamp
---------------------
 2023-08-21 15:27:59
(1 row)
gaussdb_m=# SELECT localtimestamp(3);
       timestamp
-------------------------
 2023-08-21 15:28:02.445
(1 row)
-- When the GUC parameter a_format_date_timestamp is enabled in ORA-compatible mode:
gaussdb=# SET a_format_date_timestamp=on;
SET
gaussdb=# SELECT localtimestamp(6);
      localtimestamp
----------------------------
 2023-11-24 11:41:14.086227
(1 row)
-- If a_format_version is set to 10c and a_format_dev_version is set to s2 in an ORA-compatible database, precision can be an integer of the numeric type.
gaussdb=# SET a_format_version='10c';
SET
gaussdb=# SET a_format_dev_version='s2';
SET
gaussdb=# SELECT localtimestamp(6.0);
      localtimestamp
----------------------------
 2023-11-24 11:42:45.642167
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- Zeros at the end of microseconds are not displayed. For example, 2017-09-01 10:32:19.212000 is displayed as 2017-09-01 10:32:19.212.
- When **sql_compatibility** is **'MYSQL'**, **b_format_version** is **5.7**, and **b_format_dev_version** is **s1**, this function returns the system date and time that indicates when the current SQL statement starts. It can be called with parentheses and without input parameters.
- This function has the following behaviors when **sql_compatibility** is set to **'ORA'** and the GUC parameter **a_format_date_timestamp** is set to **on**:
  - It returns a timestamp that shows the system time when the current SQL statement starts.
  - The return value type is timestamp without time zone, and the column name is **localtimestamp**.
  - When **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s2**, the **precision** parameter can be an integer of the numeric type. Otherwise, only the int type is supported.
- If **sql_compatibility** is set to **'ORA'** and the GUC parameter **a_format_date_timestamp** is set to **off**, this function returns the system date and time when the transaction starts.

- maketime(hour, minute, second)

  Description: Generates the time (of the time type) based on the input parameters hour, minute, and second. The three input parameters are of the bigint, bigint, and numeric types, respectively.

  Return type: time

  Example:

```
gaussdb=# SELECT maketime(8, 15, 26.53);
  maketime
-------------
 08:15:26.53
(1 row)

gaussdb=# SELECT maketime(-888, 15, 26.53);
  maketime
------------
 -838:59:59
(1 row)
```

  NOTE

  This function can be used only when **sql_compatibility** is set to **'MYSQL'** and the GUC parameters **b_format_version** and **b_format_dev_version** are set to **'5.7'** and **'s1'**, respectively. In addition, this function has the following behavior:

  - The function returns **NULL** if any of the following conditions is met:
    - The input parameter **minute** is less than 0 or greater than or equal to 60.
    - The input parameter **second** is less than 0 or greater than or equal to 60.
    - The value of any parameter is **NULL**.
  - The returned result of the time type contains six decimal places. If the value of **second** contains more than six decimal places, the value is rounded off.
  - The returned value of the time type is in the range [–838:59:59,838:59:59]. If the value is out of the range, the specified boundary value is returned based on the positive and negative values of **hour**.
  - **maketime** does not support self-nesting.

- now()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Returns the system date and time when the current transaction starts. The results returned in the same transaction are the same. Only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7**, and **b_format_dev_version** is set to **s1**, the system date and time when the current SQL query execution starts is returned.

Return type: timestamp with time zone. If **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7**, and **b_format_dev_version** is set to **s1**, a timestamp without time zone is returned.

Example:

```
gaussdb=# SELECT now();
              now
-------------------------------
 2017-09-01 17:03:42.549426+08
(1 row)
-- When the compatibility parameter is enabled and set to 'MYSQL':
gaussdb_m=# SELECT now();
      timestamp
---------------------
 2023-08-21 17:17:42
(1 row)
```

- now(precision)

  Description: Returns the system date and time when the current SQL statement starts.

  Parameter: **precision** (number of decimal places after the second). The precision is of the int type, within the range of [0,6] and with a default value of **0**. Setting a value greater than 6 will trigger an alarm and return a time value with the precision of 6. Using any other invalid precision will trigger an error.

  Return type: timestamp without time zone

  Implementation method: Register the system function now_b.

  Example:

  ```
  gaussdb_m=# SELECT now(3);
        timestamp
  -------------------------
   2023-08-21 17:17:48.819
  (1 row)
  ```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- now_b(precision)

  Description: Returns the system date and time that indicates when the current SQL statement starts. It is equivalent to the now function but is intended for internal implementation only. It is recommended not to use this function.

  Parameter: **precision** (number of decimal places after the second). The precision is of the int type, within the range of [0,6]. Setting a value greater than 6 will trigger an alarm and return a time value with a precision of 6.

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- timenow()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Returns the system date and time when the current SQL statement starts.

Return type: abstime

Example:

```
gaussdb=# SELECT timenow();
       timenow
------------------------
 2020-06-23 20:36:56+08
(1 row)
```

- dbtimezone

  Description: Time zone of the current database.

  Return type: text

  Example:

  ```
  gaussdb=# SELECT dbtimezone;
         dbtimezone
  ------------------------
   PRC
  (1 row)
  ```

  NOTE

  This function is valid only when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s2** in an ORA-compatible database.

- numtodsinterval(num, interval_unit)

  Description: Converts a number to the interval type. **num** is a numeric-typed number. **interval_unit** is a string in the following format: 'DAY' | 'HOUR' | 'MINUTE' | 'SECOND'

  You can set the GUC parameter **intervalstyle** to **oracle** to maintain compatibility with the output format interval.

  Return type: interval

  Example:

  ```
  gaussdb=# SELECT numtodsinterval(100, 'HOUR');
   numtodsinterval
  -----------------
   100:00:00
  (1 row)

  gaussdb=# SET intervalstyle = oracle;
  SET
  gaussdb=# SELECT numtodsinterval(100, 'HOUR');
         numtodsinterval
  -------------------------------
   +000000004 04:00:00.000000000
  (1 row)
  ```

  NOTE

  When **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s2** in an ORA-compatible database, an error is reported if **interval_unit** is set to **'DAY'** and **num** is set to a value greater than **1000000000**.

- numtoyminterval(num, interval_unit)

  Description: Converts a number to the interval type. **num** is a number of the numeric type, and **interval_unit** is a string of the fixed format ('YEAR'|'MONTH').

  You can set the GUC parameter **intervalstyle** to **oracle** to maintain compatibility with the output format interval.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: interval

Example:

```
gaussdb=# \c gaussdb_o;
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "gaussdb_o" as user "omm".
gaussdb_o=# SET a_format_version='10c';
SET
gaussdb_o=# SET a_format_dev_version='s2';
SET
gaussdb=# SELECT numtoyminterval(100, 'MONTH');
 numtoyminterval
-----------------
 8 years 4 mons
(1 row)
gaussdb_o=# SET intervalstyle = oracle;
SET
gaussdb_o=# SELECT numtoyminterval(100, 'MONTH');
 numtoyminterval
-----------------
 8-4
(1 row)

gaussdb_o=# \c postgres
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "postgres" as user "omm".
gaussdb=# DROP DATABASE gaussdb_o;
DROP DATABASE
```

NOTE

This function is valid only when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s2** in an ORA-compatible database.

- new_time(date, timezone1,timezone2)

  Description: Returns the date and time of the time zone specified by **timezone2** when the date and time of the time zone specified by **timezone1** are date.

  Return type: timestamp, datea (valid only for ORA compatibility)

  Example:

```
gaussdb=# SELECT new_time('1997-10-10','AST','EST');
      new_time
---------------------
 1997-10-09 23:00:00
(1 row)
gaussdb=# SELECT new_time(to_timestamp ('10-sep-02 14:10:10.123000','dd-mon-rr hh24:mi:ss.ff'),
'ast', 'pst');
        new_time
-------------------------
 2002-09-10 10:10:10.123
(1 row)
-- When the GUC parameter mapping_date_to_datea is enabled in ORA-compatible mode:
gaussdb=# SET nls_date_format = 'YYYY-MM-DD HH24:MI:SS';
gaussdb=# SELECT new_time(datea '2001-02-16 20:38:40', 'AST', 'EST');
        new_time
------------------------
 2001-02-16 19:38:40 AD
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When **sql_compatibility** is set to **'ORA'** and the GUC parameter **mapping_date_to_datea** is set to **on**, the database parses the date type in the input parameter to the datea type. In this case, the return value is of the datea type.
- In an Oracle-compatible database, if **a_format_version** is set to **10c**, **a_format_dev_version** is set to **s2**, and the GUC parameter **mapping_date_to_datea** is set to **off**, the database parses the date type in the input parameter to the TIMESTAMP(0) WITHOUT TIME ZONE type. In this case, the return value is of the timestamp type.

- sessiontimezone()

  Description: Returns the time zone of the current session. There is no input parameter.

  Return type: text

  Example:

```
gaussdb=# SELECT sessiontimezone;
 session_time_zone
--------------------
 PST8PDT
(1 row)
gaussdb=# SELECT lower(sessiontimezone);
   lower
-----------
 @ 8 hours
(1 row)
```

  NOTE

  - This function is valid only when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s2** in an ORA-compatible database.
  - When **session time zone** is set to **GMT+08:00** or **GMT-08:00**, positive offsets indicate areas west of Greenwich. For example, **GMT+08:00** corresponds to the GMT-8 area, while **GMT-08:00** corresponds to GMT+8.

- sys_extract_utc(timestamp| timestamptz)

  Description: Extracts Coordinated Universal Time (UTC, also formerly known as Greenwich Mean Time) from a date-time value with a time zone offset or time zone region name. If no time zone is specified, the date and time are associated with the session time zone. The input parameter can be in timestamp or timestamptz format.

  Return type: timestamp

  Example:

```
gaussdb=# SELECT sys_extract_utc(timestamp '2000-03-28 11:30:00.00');
   sys_extract_utc
---------------------
 2000-03-28 03:30:00
(1 row)
gaussdb=# SELECT sys_extract_utc(timestamptz '2000-03-28 11:30:00.00 -08:00');
   sys_extract_utc
---------------------
 2000-03-28 19:30:00
(1 row)
```

  NOTE

  This function is valid only when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s2** in an ORA-compatible database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- tz_offset('time_zone_name' | '(+/-)hh:mi' | SESSIONTIMEZONE | DBTIMEZONE)

  Description: Returns the UTC offset of the time zone indicated by the input parameter. The input parameter has the preceding four formats.

  Return type: text

  Example:

```
gaussdb=# SELECT tz_offset('us/pacific');
 tz_offset
-----------
 -08:00
(1 row)
gaussdb=# SELECT tz_offset(sessiontimezone);
 tz_offset
-----------
 +08:00
(1 row)
```

  NOTE

  This function is valid only when the value of **a_format_version** is **10c** and the value of **a_format_dev_version** is **s2** in an ORA-compatible database.

- pg_sleep(seconds)

  Description: Specifies the delay time of the server thread in unit of second. When the database calls this function, it obtains a transaction snapshot that is equivalent to a long transaction. If the time specified in the input parameter is excessively long, **oldestxmin** may not execute successfully, impacting the database's table recycling and query performance.

  Return type: void

  Example:

```
gaussdb=# SELECT pg_sleep(10);
 pg_sleep
----------

(1 row)
```

- statement_timestamp()

  Description: Returns the current date and time (start of the current statement).

  Return type: timestamp with time zone

  Example:

```
gaussdb=# SELECT statement_timestamp();
      statement_timestamp
-------------------------------
 2017-09-01 17:04:39.119267+08
(1 row)
```

- sysdate

  Description: Returns the system date and time when the current SQL statement starts.

  Return type: timestamp, datea (valid only for ORA compatibility)

  Example:

```
gaussdb=# SELECT sysdate;
       sysdate
---------------------
 2017-09-01 17:04:49
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Result returned when fixed_date is enabled.
gaussdb=# SET fixed_date='2024-05-15 11:47:02';
SET
gaussdb=# SELECT sysdate;
      timestamp
---------------------
 2024-05-15 11:47:02
(1 row)
-- Negative time settings are not supported. To indicate BC time, append BC to the end of the time.
gaussdb=# SET fixed_date = '2012-01-01 00:00:00 BC';
SET
gaussdb=# SELECT sysdate;
      timestamp
------------------------
 2012-01-01 00:00:00 BC
(1 row)
-- In ORA-compatible mode, if the GUC parameter mapping_date_to_datea is enabled, the return type is datea.
gaussdb=# SELECT sysdate;
      sysdate_a
------------------------
 2025-05-21 11:37:26 AD
(1 row)
```

NOTE

When **sql_compatibility** is set to **'ORA'** and the GUC parameter **mapping_date_to_datea** is set to **on**, this function is mapped to the sysdate_a function and the return value is changed to the datea type.

- sysdate([precision])

  Description: Returns the system date and time when a function is executed.

  Parameter: Indicates the time precision. The value is of the int type and in the range [0,6]. The default value is **0**.

  Return type: timestamp without time zone

  Example:

```
gaussdb_m=# SELECT sysdate();
     sysdate()
---------------------
 2023-08-21 17:17:42
(1 row)
gaussdb_m=# SELECT sysdate(3);
       sysdate(3)
-------------------------
 2023-08-21 17:17:48.819
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- current_sysdate()

  Description: Returns the system date and time when the current SQL statement starts.

  Return type: timestamp

  Example:

```
gaussdb=# SELECT current_sysdate();
  current_sysdate
---------------------
 2023-06-20 20:09:02
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- timeofday()

  Description: Returns the timestamp (such as **clock_timestamp**, but the return type is text) of the system time when the current function is called.

  Return type: text

  Example:

```
gaussdb=# SELECT timeofday();
              timeofday
-------------------------------------
 Fri Sep 01 17:05:01.167506 2017 CST
(1 row)
```

- transaction_timestamp()

  Description: Specifies the system date and time when the current transaction starts.

  Return type: timestamp with time zone. If **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, a timestamp without time zone is returned.

  Example:

```
gaussdb=# SELECT transaction_timestamp();
     transaction_timestamp
-------------------------------
 2017-09-01 17:05:13.534454+08
(1 row)
-- In MySQL-compatible mode:
gaussdb=#  SELECT transaction_timestamp();
   transaction_timestamp
----------------------------
 2023-09-07 09:32:09.728998
(1 row)
```

- add_months(d,n)

  Description: Returns the time point *d* plus *n* months.

  Parameters:

  – **d**: indicates a value of the timestamp type or a value that can be implicitly converted into the timestamp type.

  – **n**: indicates the value of the INTEGER type and the value that can be implicitly converted to the INTEGER type.

  Return type: timestamp, datea (valid only for ORA compatibility)

  Example:

```
gaussdb=# SELECT add_months(to_date('2017-5-29', 'yyyy-mm-dd'), 11) FROM sys_dummy;
     add_months
---------------------
 2018-04-29 00:00:00
(1 row)
-- When the GUC parameter mapping_date_to_datea is enabled in ORA-compatible mode:
gaussdb=# SET nls_date_format = 'YYYY-MM-DD HH24:MI:SS';
gaussdb=# SELECT add_months(datea '2020-11-12 11:12:13', 3);
       add_months
-------------------------
 2021-02-12 11:12:13 AD
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When **sql_compatibility** is set to **'ORA'** and the GUC parameter **mapping_date_to_datea** is set to **on**, the database parses the date type in the input parameter to the datea type. In this case, the return value is of the datea type.
- In this function, if the GUC parameter **mapping_date_to_datea** is set to **off**, the database parses the date type in the input parameter to the TIMESTAMP(0) WITHOUT TIME ZONE type. In this case, the return value is of the timestamp type. When the input parameter type of this function is parsed to TIMESTAMP(0) WITHOUT TIME ZONE, the function has the following behaviors in the Oracle-compatible database where the values of **a_format_version** and **a_format_dev_version** are **10c** and **s1**, respectively:
  - If the returned result is later than 9999, an error is reported.
  - When the input parameter **n** is a decimal, the function does not perform rounding off on the decimal places but truncates them directly.
- When **sql_compatibility** is set to **'ORA'**, if the parameter is of the datea type (for example, in the **add_months(to_datea('2025-2-28', 'yyyy-mm-dd'),1)** scenario), and the value of **behavior_compat_options** contains **end_month_calculate** (that is, **SET behavior_compat_options = 'end_month_calculate'**), the return result is **'2025-03-31 00:00:00 AD'**. Otherwise, the return result is **'2025-03-28 00:00:00 AD'**.

- last_day(d)

  Description: Returns the date of the last day of the month that contains *d*.

  Return type: timestamp, datea (valid only for ORA compatibility)

  Example:

```
gaussdb=# SELECT last_day(to_date('2017-01-01', 'yyyy-mm-dd')) AS cal_result;
    cal_result
---------------------
 2017-01-31 00:00:00
(1 row)
-- When the GUC parameter mapping_date_to_datea is enabled in ORA-compatible mode:
gaussdb=# SET nls_date_format = 'YYYY-MM-DD HH24:MI:SS';
gaussdb=# SELECT last_day(datea '2020-11-12 11:12:13');
       last_day
------------------------
 2020-11-30 11:12:13 AD
(1 row)
```

  NOTE

  - When **sql_compatibility** is set to **'ORA'** and the GUC parameter **mapping_date_to_datea** is set to **on**, the database parses the date type in the input parameter to the datea type. In this case, the return value is of the datea type.
  - When **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, the last_day function calls the built-in function b_db_last_day. The input parameter type can be TEXT, DATE, DATETIME, or TIME. The return value is of the date type and can be a number in the datetime format.

- months_between(d1, d2)

  Description: Calculates the month difference between time points **d1** and **d2**. If both dates are the end of a month or are the same day, an integer is returned. Otherwise, the return value is a decimal and is calculated as 31 days per month.

  Return type: numeric

  Example:

```
gaussdb=# SELECT months_between(to_date('2022-10-31', 'yyyy-mm-dd'), to_date('2022-09-30',
'yyyy-mm-dd'));
 months_between
----------------
              1
(1 row)

gaussdb=# SELECT months_between(to_date('2022-10-30', 'yyyy-mm-dd'), to_date('2022-09-30',
'yyyy-mm-dd'));
 months_between
----------------
              1
(1 row)

gaussdb=# SELECT months_between(to_date('2022-10-29', 'yyyy-mm-dd'), to_date('2022-09-30',
'yyyy-mm-dd'));
    months_between
-----------------------
 .96774193548387096774
(1 row)
-- In ORA-compatible mode, if the GUC parameter mapping_date_to_datea is enabled, the date type
is mapped to the datea type.
gaussdb=# SET nls_date_format = 'YYYY-MM-DD HH24:MI:SS';
gaussdb=# SELECT months_between('2020-11-12 11:12:13'::date, '2020-11-13 11:12:13'::date);
     months_between
------------------------
 -.03225806451612903226
(1 row)
```

NOTE

- This function is valid only when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s1** in an ORA-compatible database.
- When **sql_compatibility** is set to **'ORA'** and the GUC parameter **mapping_date_to_datea** is set to **on**, the database parses the date type in the input parameter to the datea type.

- next_day(x,y)

  Description: Calculates the time of the next week y started from x.

  Return type: timestamp, datea (valid only for ORA compatibility)

  Example:

```
gaussdb=# SELECT next_day(timestamp '2017-05-25 00:00:00','Sunday')AS cal_result;
     cal_result
---------------------
 2017-05-28 00:00:00
(1 row)
gaussdb=# SELECT next_day(timestamp '2024-01-17 00:00:00', 7.9999999);
ERROR:  Week day out of range, the range is 1 ... 7
CONTEXT:  referenced column: next_day
gaussdb=# SET a_format_version='10c';
SET
gaussdb=# SET a_format_dev_version='s1';
SET
gaussdb=# SELECT next_day(timestamp '2024-01-17 00:00:00', 7.9999999);
      next_day
---------------------
 2024-01-20 00:00:00
(1 row)
-- When the GUC parameter mapping_date_to_datea is enabled in ORA-compatible mode:
gaussdb=# SET nls_date_format = 'YYYY-MM-DD HH24:MI:SS';
gaussdb=# SELECT next_day(datea '2020-11-12 11:12:13', 'Monday');
       next_day
------------------------
 2020-11-16 11:12:13 AD
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If **a_format_version** is set to **10c**, **a_format_dev_version** is set to **s1**, and the second parameter is a floating-point number, this function discards the decimal part in the ORA-compatible database. Otherwise, this function rounds off the decimal part.
- In an ORA-compatible database, the first parameter can be of the TIMESTAMP WITH TIME ZONE type only when **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1**.
- When **sql_compatibility** is set to **'ORA'** and the GUC parameter **mapping_date_to_datea** is set to **on**, the database parses the date type in the input parameter to the datea type.
- The second parameter of **next_day** supports abbreviations (case-insensitive), including **SUN**, **MON**, **TUE**, **WED**, **THU**, **THUR**, **FRI**, and **SAT**.

- tinterval(abstime, abstime)

  Description: Creates a time interval with two specific time points.

  Return type: tinterval

  Example:

```
gaussdb=# CALL tinterval(abstime 'May 10, 1947 23:59:12', abstime 'Mon May  1 00:30:30 1995');
                    tinterval
----------------------------------------------------
 ["1947-05-10 23:59:12+08" "1995-05-01 00:30:30+08"]
(1 row)
```

- tintervalend(tinterval)

  Description: Returns the end time of tinterval.

  Return type: abstime

  Example:

```
gaussdb=# SELECT tintervalend('["Sep 4, 1983 23:59:12" "Oct4, 1983 23:59:12"]');
      tintervalend
-------------------------
 1983-10-04 23:59:12+08
(1 row)
```

- tintervalrel(tinterval)

  Description: Calculates and returns the relative time of **tinterval**.

  Return type: reltime

  Example:

```
gaussdb=# SELECT tintervalrel('["Sep 4, 1983 23:59:12" "Oct4, 1983 23:59:12"]');
 tintervalrel
--------------
 1 mon
(1 row)
```

- smalldatetime_ge(smalldatetime, smalldatetime)

  Description: Checks whether the value of the first parameter is greater than or equal to that of the second parameter.

  Return type: Boolean

- smalldatetime_cmp(smalldatetime, smalldatetime)

  Description: Compares two smalldatetime values to check whether they are the same.

  Return type: integer

- smalldatetime_eq(smalldatetime, smalldatetime)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Compares two smalldatetime values to check whether they are the same.

Return type: Boolean

- smalldatetime_gt(smalldatetime, smalldatetime)

  Description: Determines whether the first parameter is greater than the second parameter.

  Return type: Boolean

- smalldatetime_hash(smalldatetime)

  Description: Calculates the hash value corresponding to the smalldatetime value.

  Return type: integer

- smalldatetime_in(cstring, oid, integer)

  Description: Converts the input into a smalldatetime value.

  Return type: smalldatetime

- smalldatetime_larger(smalldatetime, smalldatetime)

  Description: Returns a larger smalldatetime value.

  Return type: smalldatetime

- smalldatetime_le(smalldatetime, smalldatetime)

  Description: Checks whether the value of the first parameter is less than or equal to that of the second parameter.

  Return type: Boolean

- smalldatetime_lt(smalldatetime, smalldatetime)

  Description: Determines whether the first parameter is less than the second parameter.

  Return type: Boolean

- smalldatetime_ne(smalldatetime, smalldatetime)

  Description: Compares two smalldatetime values to check whether they are different.

  Return type: Boolean

- smalldatetime_out(smalldatetime)

  Description: Converts a smalldatetime value to an external string.

  Return type: cstring

- smalldatetime_send(smalldatetime)

  Description: Converts a smalldatetime value to the binary format.

  Return type: bytea

- smalldatetime_smaller(smalldatetime, smalldatetime)

  Description: Returns a smaller smalldatetime.

  Return type: smalldatetime

- smalldatetime_to_abstime(smalldatetime)

  Description: Converts smalldatetime to abstime.

  Return type: abstime

- smalldatetime_to_time(smalldatetime)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Converts smalldatetime to time.
Return type: time without time zone

- smalldatetime_to_timestamp(smalldatetime)
  Description: Converts smalldatetime to timestamp.
  Return type: timestamp without time zone
- smalldatetime_to_timestamptz(smalldatetime)
  Description: Converts smalldatetime to timestamptz.
  Return type: timestamp with time zone
- smalldatetime_to_varchar2(smalldatetime)
  Description: Converts smalldatetime to varchar2.
  Return type: character varying.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

There are multiple methods for obtaining the current time. Select an appropriate API based on the actual service scenario.

- The following APIs return values based on the start time of the current transaction:

```
CURRENT_DATE CURRENT_TIME
CURRENT_TIME(precision)
CURRENT_TIMESTAMP(precision)
LOCALTIME
LOCALTIMESTAMP
LOCALTIME(precision)
LOCALTIMESTAMP(precision)
transaction_timestamp()
now()
```

  The values passed by **CURRENT_TIME** and **CURRENT_TIMESTAMP(precision)** contain time zone information. The values passed by **LOCALTIME** and **LOCALTIMESTAMP** do not contain time zone information. CURRENT_TIME, LOCALTIME, and LOCALTIMESTAMP can specify a precision parameter, which rounds the seconds field of the result to the decimal place. If there is no precision parameter, the result is given the full precision that can be obtained.

  These functions maintain consistent values throughout the transaction by returning results based on the start time of the current transaction. This ensures that the transaction can maintain consistency at the "current" moment and that multiple modifications within the same transaction retain the same timestamp. **transaction_timestamp()** is equivalent to **CURRENT_TIMESTAMP(precision)**, indicating the start time of the transaction where the current statement is located. **now()** is equivalent to **transaction_timestamp()**.

  When a transaction starts, **sql_compatibility** is set to **'ORA'**, and **a_format_date_timestamp** is set to **on**, the results returned by the CURRENT_DATE, CURRENT_TIMESTAMP(precision), LOCALTIMESTAMP and LOCALTIMESTAMP (precision) functions are the timestamp when the current SQL statement starts to be executed. If the GUC parameter **a_format_date_timestamp** is disabled, the returned result is the transaction start date or date and time.

  When **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, CURDATE, CURRENT_DATE, CURTIME, CURRENT_TIME, CURRENT_TIMESTAMP, NOW, LOCALTIME, LOCALTIMESTAMP, UTC_TIME and UTC_TIMESTAMP all return the start time of SQL statement execution, and SYSDATE returns the start time of function calling. transaction_timestamp() still indicates the transaction start time. The return value type is changed to timestamp without time zone.

- The following APIs return the start time of the current statement:

```
statement_timestamp()
```

  statement_timestamp() returns the start time of the current statement (more accurately, the time when the last instruction is received from the client). The return values of **statement_timestamp()** and **transaction_timestamp()** are the same during the execution of the first instruction of a transaction, but may be different in subsequent instructions.

- The following APIs return the actual current time when the function is called:

```
clock_timestamp()
timeofday()
```

  clock_timestamp() returns the actual current time, and its value changes even in the same SQL statement. Similar to **clock_timestamp()**, **timeofday()** also returns the actual current time. However, the result of **timeofday()** is a formatted text string instead of a timestamp with time zone information.

- convert_tz(dt, from_tz, to_tz)

  Description: Converts the date and time value **dt** from the **from_tz** time zone to the **to_tz** time zone.

  Parameters: See **Table 1-51**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-51 convert_tz parameters

Parameter	Type	Description	Value Range
dt	datetime, date, text, and numeric	Time and date value.	[0000-01-01 00:00:00.000000,9999-12-31 23:59:59.999999].
from_tz	text	Source time zone, indicating the time zone to which **dt** belongs.	• The offset format is [–15:59, 5:00], for example, '+08:00'. It indicates the offset from the UTC time. • Named time zone, such as MET and UTC. It indicates the time zone name. For details, see the **PG_TIMEZONE_NAMES** system view.
to_tz	text	Target time zone, indicating the time zone after conversion.	

Return type: datetime

Example:

```
gaussdb=# SET b_format_dev_version = 's1';
SET
gaussdb=# SET b_format_version = '5.7';
SET
gaussdb=# SELECT convert_tz(cast('2023-01-01 10:10:10' as datetime), '+00:00', '+01:00');
    convert_tz
---------------------
 2023-01-01 11:10:10
(1 row)
gaussdb=# SELECT convert_tz(cast('2023-01-01' as date), '+00:00', '+01:00');
    convert_tz
---------------------
 2023-01-01 01:00:00
(1 row)
gaussdb=# SELECT convert_tz('2023-01-01 10:10:10', '+00:00', '+01:00');
    convert_tz
---------------------
 2023-01-01 11:10:10
(1 row)
gaussdb=# SELECT convert_tz('2023-01-01', '+00:00', '+01:00');
    convert_tz
---------------------
 2023-01-01 01:00:00
(1 row)
gaussdb=# SELECT convert_tz(20230101101010, '+00:00', '+01:00');
    convert_tz
---------------------
 2023-01-01 11:10:10
(1 row)
gaussdb=# SELECT convert_tz(20230101, '+00:00', '+01:00');
    convert_tz
---------------------
 2023-01-01 01:00:00
(1 row)
gaussdb=# SELECT convert_tz('2023-01-01 10:10:10', 'UTC', 'PRC');
    convert_tz
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
---------------------
 2023-01-01 18:10:10
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- sec_to_time(seconds)

  Description: Converts the number of seconds into a time of the time type.

  Parameters: See **Table 1-52**.

  Table 1-52 sec_to_time parameters

Parameter	Type	Description	Value Range
seconds	numeric and text	Number of seconds	[–3020399,+3020399], corresponding to the range of the time type [–838:59:59,838:59:59]. An out-of-bounds input will be truncated to the corresponding boundary value.

  Return type: time without time zone

  Example:

```
gaussdb=# SET b_format_dev_version = 's1';
SET
gaussdb=# SET b_format_version = '5.7';
SET
gaussdb=# SELECT sec_to_time(2000);
 sec_to_time
-------------
 00:33:20
(1 row)
gaussdb=# SELECT sec_to_time('-2000');
 sec_to_time
-------------
 -00:33:20
(1 row)
```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- adddate(date, INTERVAL val unit)

  Description: Returns a new date by adding a certain interval to a date.

  Parameters: See **Table 1-53**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-53 adddate(date, INTERVAL val unit) parameters

Parameter	Type	Description	Value Range
date	Expression of the time type, text, datetime, date, or time	Date to be added with an interval.	See the value ranges of the corresponding types.
val	Integer, floating-point number, or character string	Interval to be added.	See the value ranges of the corresponding types.
unit	Keyword	Unit of the interval.	**YEAR**, **QUARTER**, **MONTH**, **WEEK**, **DAY**, **HOUR**, **MINUTE**, **SECOND**, and **MICROSECOND**. For details, see **Time Interval Expressions**.

Return type: TEXT, DATE, DATETIME, or TIME WITHOUT TIMEZONE

Example:

```
gaussdb=# SELECT adddate('2018-05-01', INTERVAL 1 DAY);
  adddate
------------
 2018-05-02
(1 row)
```

NOTE

- This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.
- When working with the PREPARE statement, note that the second parameter of the adddate function is an interval expression. If this parameter is replaced by **$1**, you will encounter unexpected results. For example, running **PREPARE p1 AS SELECT adddate('2023-01-01 10:00:00', $1); EXECUTE p1(interval 13 hour);** will yield the unexpected result of **'2023-01-01 10:00:00'**.

- adddate(expr, days)

  Description: Returns a new date by adding a certain number of days to a date.

  Parameters:

  – **expr**: specifies the start date and time. The value type can be expression of the time type, TEXT, DATE, DATETIME, or TIME.

  – **days**: specifies the number of days to be added. The value is of the int type.

  Return type: TEXT, DATE, DATETIME, or TIME WITHOUT TIMEZONE

  Example:

```
gaussdb=# SELECT ADDDATE('2018-05-01', 1);
  adddate
------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 2018-05-02
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- curdate()

  Description: Returns the system date when the local function calling starts. The time zone can be changed within the same connection. The returned date is affected by the time zone.

  Return type: date

  Example:

```
gaussdb=# SELECT curdate();
  curdate
------------
 2023-08-10
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- curtime([precision])

  Description: Returns the system time when the SQL query call starts.

  Parameter: **precision** (number of decimal places after the second). The precision is of the int type, within the range of [0,6] and with a default value of **0**. If the provided value can be converted into an integer within this range, the function will return a time value with the specified precision. Otherwise, the function will report an error.

  Return type: time without time zone

  Implementation method: Register the system function curtime.

  Example:

```
gaussdb=# SELECT curtime(3);
    curtime
--------------
 16:59:57.203
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- date_add(date,INTERVAL val unit)

  Description: Adds a period of time to a specified date and returns the calculation result.

  Parameters: See **Table 1-54**.

Table 1-54 date_add(date,INTERVAL val unit) parameters

Parameter	Type	Description	Value Range
date	Expression of the time type, text, datetime, date, or time	Date to be added with an interval.	See the value ranges of the corresponding types.
val	Integer, floating-point number, or character string	Interval to be added.	See the value ranges of the corresponding types.
unit	Keyword	Unit of the interval.	**YEAR**, **QUARTER**, **MONTH**, **WEEK**, **DAY**, **HOUR**, **MINUTE**, **SECOND**, and **MICROSECOND**. For details, see **Time Interval Expressions**.

Return type: a single return value. For details, see **Table 1-55**.

Table 1-55 Return value types of date_add(date,INTERVAL val unit)

Return Type	Description
TEXT	The **date** input parameter is of the text type.
DATE	The **date** input parameter is of the date type, and the value of the **unit** input parameter is greater than or equal to **day** (for example, **week** or **month**).
TIME WITHOUT TIMEZONE	The **date** input parameter is of the time type.
DATETIME	The **date** input parameter is of the datetime type, or the **date** input parameter and the value of the **unit** input parameter is less than **day** (for example, **hour** or **second**).

Example:

```
gaussdb=#  SELECT date_add('2018-05-01', INTERVAL 1 DAY);
  date_add
------------
 2018-05-02
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.
- When working with the PREPARE statement, note that the second parameter of the date_add function is an interval expression and cannot be replaced by **$1**, for example, **PREPARE p1 AS SELECT date_add('2023-01-01 10:00:00', $1);**.

- date_add(expr, days)

  Description: Returns a new date by adding a certain number of days to a date.

  Parameters:

  – **expr**: specifies the start date and time. The value type can be expression of the time type, TEXT, DATE, DATETIME, or TIME.

  – **days**: specifies the number of days to be added. The value is of the int type.

  Return type: TEXT, DATE, DATETIME, or TIME WITHOUT TIMEZONE

  Example:

```
gaussdb=# SELECT DATE_ADD('2018-05-01', 1);
  date_add
------------
 2018-05-02
(1 row)
```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- date_format(date, format)

  Description: Outputs the date and time in the specified format.

  Parameters: See **Table 1-56**.

Table 1-56 date_format parameters

Parameter	Type	Description	Value Range
date	Time expression, TEXT, DATETIME, DATE, or TIME	Date to be formatted.	See the value ranges of the corresponding types.
format	text	A formatted string.	For details, see **Table 1-57**.

Table 1-57 lists the format parameters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-57 Values and meanings of format

Value	Meaning
%a	Abbreviation of every day in a week (Sun–Sat).
%b	Abbreviation of every month (Jan–Dec).
%c	Month number (1–12).
%D	Every day in a month with an English prefix (1st, 2nd, 3rd, ...).
%d	Two-digit representation of every day in a month (01–31).
%e	Sequence number of every day in a month (1–31).
%f	Microsecond (000000–999999).
%H	Hour (00–23).
%h	Hour (01–12).
%I	Hour (01–12).
%i	Minutes (00–59).
%j	Every day in a year (001–366).
%k	Hour (0–23).
%l	Hour (1–12).
%M	Month name (January–December).
%m	Two-digit representation of every month (01–12).
%p	AM or PM
%r	12-hour time (hh:mm:ss followed by AM or PM).
%S	Seconds (00–59).
%s	Seconds (00–59).
%T	24-hour time (hh:mm:ss)
%U	Week of a year (00–53). Each week starts from Sunday.
%u	Week of a year (00–53). Each week starts from Monday.
%V	Week of a year (01–53). Each week starts from Sunday.
%v	Week of a year (01–53). Each week starts from Monday.
%W	Name of every day in a week (Sunday–Saturday).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Meaning
%w	Every day in a week (0–6), where **0** represents Sunday, and **6** represents Saturday.
%X	Week of a year. Each week starts from Sunday. The value is a four-digit number and is used for %V.
%x	Week of a year. Each week starts from Monday. The value is a four-digit number and is used for %v.
%Y	Four-digit year
%y	Two-digit year

Return type: text

Example:

```
gaussdb=# SELECT date_format('2023-10-11 12:13:14.151617','%b %c %M %m');
    date_format
--------------------
 Oct 10 October 10
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- date_sub(date, INTERVAL val unit)

  Description: Returns a new date by subtracting a certain interval from a date.

  Parameters: See **Table 1-58**.

Table 1-58 date_sub(date, INTERVAL val unit) parameters

Parameter	Type	Description	Value Range
date	Expression of the time type, text, datetime, date, or time	Date to be added with an interval.	See the value ranges of the corresponding types.
val	Integer, floating-point number, or character string	Interval to be added.	See the value ranges of the corresponding types.
unit	Keyword	Unit of the interval.	**YEAR**, **QUARTER**, **MONTH**, **WEEK**, **DAY**, **HOUR**, **MINUTE**, **SECOND**, and **MICROSECOND**. For details, see **Time Interval Expressions**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: TEXT, DATE, DATETIME, or TIME WITHOUT TIMEZONE

Example:

```
gaussdb=#  SELECT date_sub('2018-05-01', INTERVAL 1 YEAR);
  date_sub
------------
 2017-05-01
(1 row)
```

NOTE

- This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.
- When working with the PREPARE statement, note that the second parameter of the date_sub function is an interval expression and cannot be replaced by **$1**, for example, **PREPARE p1 AS SELECT date_sub('2023-01-01 10:00:00', $1);**.

- date_sub(expr, days)

  Description: Specifies the start date and time and the number of days to be subtracted from the start date and time, and returns the subtraction result.

  Parameters:

  – **expr**: specifies the start date and time. The value type can be expression of the time type, TEXT, DATE, DATETIME, or TIME.

  – **days**: specifies the number of days to be subtracted. The value is of the int type.

  Return type: TEXT, DATE, DATETIME, or TIME WITHOUT TIMEZONE

  Example:

  ```
  gaussdb=#  SELECT date_sub('2023-1-1', 20);
    date_sub
  ------------
   2022-12-12
  (1 row)
  ```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- datediff(expr1, expr2)

  Description: Returns the number of days between two time expressions.

  Parameter: Expression of the time type, text, datetime, date, or time

  Return type: int.

  Example:

  ```
  gaussdb=#  SELECT datediff('2021-11-12','2021-11-13');
   datediff
  ----------
         -1
  (1 row)
  ```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- day()/dayofmonth()

  Description: Extracts the number of days in a date and time expression and returns the result. dayofmonth() is the alias of day() and has the same function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter: The input parameter is the date and time to be extracted, which can be an expression of the time type, or of the text, datetime, date, or time type.

Return type: int.

Example:

```
gaussdb=# SELECT day('2023-01-02');
 day
-----
   2
(1 row)
gaussdb=# SELECT dayofmonth('23-05-22');
 dayofmonth
------------
         22
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- dayname()

  Description: Returns the name of a day in a week.

  Parameter: The input parameter is a time expression, TEXT, DATETIME, DATE, or TIME.

  Return type: text

  Example:

  ```
  gaussdb=# SELECT dayname('2023-10-11');
    dayname
  -----------
   Wednesday
  (1 row)
  ```

  NOTE

  - This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.
  - The language used by the return value is specified by the GUC parameter **lc_time_names**.

- dayofweek()

  Description: Returns the working day index of a date (1 = Sunday, 2 = Monday, ..., 7 = Saturday).

  Parameter: Expression of the time type, text, datetime, date, or time

  Return type: bigint

  Example:

  ```
  gaussdb=# SELECT dayofweek('2023-04-16');
   dayofweek
  -----------
           1
  (1 row)
  ```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- dayofyear()

  Description: Returns the number of a day in the year.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter: The input parameter is a time expression, TEXT, DATETIME, DATE, or TIME.

Return type: int; value range: 1 to 366.

Example:

```
gaussdb=# SELECT dayofyear('2000-12-31');
 dayofyear
-----------
       366
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- extract(unit FROM date)

  Description: Extracts part of the time. This function can be used in other modes, but the behavior varies greatly. When **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, the function is described as follows:

  Parameters:

  – **unit**: text type. For details about the value types, see **Table 1-59**.

  – **date**: date, time, datetime, text, datea (valid only for ORA compatibility), or other time type expressions.

  Return type: bigint

Table 1-59 Values and meanings of unit

Value of UNIT	Description
MICROSECOND	Microsecond
SECOND	Second
MINUTE	Minute
HOUR	Hour
DAY	Day
WEEK	Week
MONTH	Month
QUARTER	Quarter
YEAR	Year
SECOND_MICROSECOND	Concatenated value of second and microsecond
MINUTE_MICROSECOND	Concatenated value of minute, second, and microsecond
MINUTE_SECOND	Concatenated value of minute and second

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value of UNIT	Description
HOUR_MICROSECOND	Concatenated value of hour, minute, second, and microsecond
HOUR_SECOND	Concatenated value of hour, minute, and second
HOUR_MINUTE	Concatenated value of hour and minute
DAY_MICROSECOND	Concatenated value of day, hour, minute, second, and microsecond
DAY_SECOND	Concatenated value of day, hour, minute, and second
DAY_MINUTE	Concatenated value of day and minute
DAY_HOUR	Concatenated value of day and hour
YEAR_MONTH	Concatenated value of year and month
EPOCH	Total number of seconds or interval since 1970-01-01 00:00:00-00 UTC

Example:

```
gaussdb=# SELECT extract(YEAR FROM '2023-10-11');
 extract
---------
    2023
(1 row)

gaussdb=# SELECT extract(QUARTER FROM '2023-10-11');
 extract
---------
       4
(1 row)

gaussdb=# SELECT extract(MONTH FROM '2023-10-11');
 extract
---------
      10
(1 row)

gaussdb=# SELECT extract(WEEK FROM '2023-10-11');
 extract
---------
      41
(1 row)

gaussdb=# SELECT extract(DAY FROM '2023-10-11');
 extract
---------
      11
(1 row)

gaussdb=# SELECT extract(HOUR FROM '2023-10-11 12:13:14');
 extract
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
---------
    12
(1 row)
```

NOTE

- When **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7**, and **b_format_dev_version** is set to **s1**, the actually called function is b_extract. In other cases, the actually called function is date_part, which can be queried using commands like **\df b_extract** for detailed input parameters and return values.
- When **sql_compatibility** is set to **'ORA'**:
  - When the GUC parameter **mapping_date_to_datea** is set to **on**, the database parses the date type in the input parameter to the datea type. In this case, the unit can only be year, month, or day. If other units are specified, an error is reported.
  - When the input parameter is of the datea type, the unit can only be year, month, or day regardless of whether the GUC parameter **mapping_date_to_datea** is set to **on** or **off**. If other units are specified, an error is reported.

The GUC parameter **default_week_format** is used to process special week issues. The default value is **0**. For details, see **Table 1-60**.

Table 1-60 default_week_format

default_week_format	First Day of a Week	Range	Which Week is the First Week
0	Sunday	0–53	Week containing Sunday in this year
1	Monday	0–53	Week containing four or more days in this year
2	Sunday	1–53	Week containing Sunday in this year
3	Monday	1–53	Week containing four or more days in this year
4	Sunday	0–53	Week containing four or more days in this year
5	Monday	0–53	Week containing Monday in this year
6	Sunday	1–53	Week containing four or more days in this year
7	Monday	1–53	Week containing Monday in this year

- from_days(days)

  Description: Returns the corresponding date value given a number of days.

  Parameter: The input parameter is a time expression, TEXT, DATETIME, DATE, or TIME.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: date

Example:

```
gaussdb=# SELECT from_days(36524);--0099-12-31
 from_days
------------
 0099-12-31
(1 row)
```

NOTE

- This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.
- Null characters and zeros are processed as 0. For input parameters that cannot be correctly converted to bigint, an error is reported.
- If the input parameter is less than **366**, the return date is **0000-00-00**.

- from_unixtime(unix_timestamp[,format])

  Description: Converts a Unix timestamp to the date and time format. A Unix timestamp is the number of seconds from 08:00:00 UTC on January 1, 1970 to a specified time.

  Parameters:

  – **unix_timestamp**: Unix timestamp, which is of the numeric type.

  – **format**: time format. The value is of the text type.

  Return type: text/datetime

  Example:

  ```
  gaussdb=# SELECT from_unixtime(1111885200);
     from_unixtime
  ---------------------
   2005-03-27 09:00:00
  (1 row)
  ```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- get_format({DATE | TIME | DATETIME | TIMESTAMP}, {'EUR' | 'USA' | 'JIS' | 'ISO' | 'INTERNAL'})

  Description: Obtains the date and time string corresponding to the specified time type under a certain locale standard.

  Parameters:

  – **DATE|TIME|DATETIME|TIMESTAMP**: time types, which are keywords.

  – **'EUR'|'USA'|'JIS'|'ISO'|'INTERNAL'**: five time formats, which are of the text type.

  Return type: text

  Example:

  ```
  gaussdb=# SELECT get_format(date, 'eur');
   get_format
  ------------
   %d.%m.%y
  (1 row)
  gaussdb=# SELECT get_format(date, 'usa');
   get_format
  ------------
   %m.%d.%y
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.
- When working with the PREPARE statement, note that the first parameter of the get_format function is a keyword and cannot be replaced by **$1**, for example, **PREPARE p1 AS SELECT get_format($1, 'usa');**.

- hour()

  Description: Returns the hour part of a time after you enter a time type.

  Parameter: Expression of the time type, text, datetime, date, or time

  Return type: bigint

  Example:

```
gaussdb=# SELECT hour('10:10:10.1');
 hour
------
   10
(1 row)
```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- makedate(year,dayofyear)

  Description: Returns a date based on the given year and day.

  Parameters:

  – **year**: The parameter type is bigint.

  – **dayofyear**: The parameter type is bigint, indicating the number of days in the year. The value can cross years. If the value is less than or equal to **0**, **null** is returned.

  Return type: date

  Example:

```
gaussdb=# SELECT makedate(2000, 60);
  makedate
------------
 2000-02-29
(1 row)
```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- microsecond()

  Description: Returns the microsecond part of a time type you enter.

  Parameter: Expression of the time type, text, datetime, date, or time

  Return type: bigint

  Example:

```
gaussdb=# SELECT microsecond('2023-5-5 10:10:10.24485');
 microsecond
-------------
      244850
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- minute()

  Description: Returns the minute part of a time type you enter.

  Parameter: Expression of the time type, text, datetime, date, or time

  Return type: bigint

  Example:

```
gaussdb=# SELECT minute(time'10:10:10');
 minute
--------
     10
(1 row)
```

  > **NOTE**
  >
  > This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- month()

  Description: Returns the month of a specified date and time.

  Parameter: The input parameter is the date and time to be extracted, which can be an expression of the time type, or of the TEXT, DATETIME, DATE, or TIME type.

  Return type: int.

  Example:

```
gaussdb=# SELECT month('2021-11-30');
 month
-------
    11
(1 row)
```

  > **NOTE**
  >
  > This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- monthname()

  Description: Returns the full month name of a date.

  Parameter: The input parameter is a time expression, TEXT, DATETIME, DATE, or TIME.

  Return type: text

  Example:

```
gaussdb=# SELECT monthname('2023-02-28');
 monthname
-----------
 February
(1 row)
```

  > **NOTE**
  >
  > - This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.
  > - The language used by the return value is specified by the GUC parameter **lc_time_names**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- period_add(period, month_number)

  Description: Adds a specified number of months to a specified time segment and returns the result as a time segment.

  Parameters:

  – **period**: The parameter type is bigint. The value is a date in *YYYYMM* or *YYMM* format.

  – **month_number**: The parameter type is bigint. The value is the number of months to be added. The value can be a negative number.

  Return type: bigint, which is a date in YYYYMM format.

  Example:

  ```
  gaussdb=# SELECT period_add(202205, -12);
   period_add
  ------------
      202105
  (1 row)
  ```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- period_diff(p1, p2)

  Description: Calculates the month difference between two time segments.

  Parameters: The parameter type is bigint. The two parameters use the *YYMM* or *YYYYMM* format to indicate the time segment.

  Return type: bigint (month difference)

  Example:

  ```
  gaussdb=# SELECT period_diff('202101', '202102');
   period_diff
  -------------
            -1
  (1 row)
  ```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- second()

  Description: Returns the second part of a time type you enter.

  Parameter: Expression of the time type, text, datetime, date, or time

  Return type: bigint

  Example:

  ```
  gaussdb=# SELECT second('2023-5-5 10:10:10');
   second
  --------
      10
  (1 row)
  ```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- quarter()

  Description: Returns the quarter part of a date after you enter a date type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter: The input parameter is the date and time to be extracted, which can be an expression of the time type, or of the TEXT, DATETIME, DATE, or TIME type.

Return type: bigint

Example:

```
gaussdb=# SELECT quarter('2012-1-1');
 quarter
---------
       1
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- str_to_date(str, format)

  Description: Converts a specified character string to a date/time based on the specified date format.

  Parameters:

  - **str**: character string of the text type, formatted into date.
  - **format**: string of the text type, indicating the target format. **Table 1-61** lists the format parameters.

  Table 1-61 Values and meanings of format

Value	Meaning
%a	Abbreviation of every day in a week (Sun–Sat).
%b	Abbreviation of every month (Jan–Dec).
%c	Month number (1–12).
%D	Every day in a month with an English prefix (1st, 2nd, 3rd, ...).
%d	Two-digit representation of every day in a month (01–31).
%e	Sequence number of every day in a month (1–31).
%f	Microsecond (000000–999999).
%H	Hour (00–23).
%h	Hour (01–12).
%I	Hour (01–12).
%i	Minutes (00–59).
%j	Every day in a year (001–366).
%k	Hour (0–23).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Meaning
%l	Hour (1–12).
%M	Month name (January–December).
%m	Two-digit representation of every month (01–12).
%p	AM or PM
%r	12-hour time (hh:mm:ss followed by AM or PM).
%S	Seconds (00–59).
%s	Seconds (00–59).
%T	24-hour time (hh:mm:ss)
%U	Week of a year (00–53). Each week starts from Sunday.
%u	Week of a year (00–53). Each week starts from Monday.
%V	Week of a year (01–53). Each week starts from Sunday.
%v	Week of a year (01–53). Each week starts from Monday.
%W	Name of every day in a week (Sunday–Saturday).
%w	Every day in a week (0–6), where **0** represents Sunday, and **6** represents Saturday.
%X	Week of a year. Each week starts from Sunday. The value is a four-digit number and is used for %V.
%x	Week of a year. Each week starts from Monday. The value is a four-digit number and is used for %v.
%Y	Four-digit year
%y	Two-digit year

Return type: text

Example:

```
gaussdb=# SELECT str_to_date('May 1, 2013','%M %d,%Y');--2013-05-01
 str_to_date
-------------
 2013-05-01
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.
- Only the time in YYYY-MM-DD format can be returned.
- If the time contains 0 year, 0 month, and 0 day or contains only hour, minute, and second, an alarm is generated and **NULL** is returned.

- subdate(expr, days)

  Description: Specifies the start date and time and the number of days to be subtracted from the start date and time, and returns the subtraction result.

  Parameters:

  – **expr**: specifies the start date and time. The value type can be expression of the time type, TEXT, DATE, DATETIME, or TIME.

  – **days**: specifies the number of days to be subtracted. The value is of the int type.

  Return type: TEXT, DATE, DATETIME, or TIME WITHOUT TIMEZONE

  Example:

```
gaussdb=# SELECT subdate('2023-1-1', 20);
  subdate
------------
 2022-12-12
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- subdate(expr, INTERVAL val unit)

  Description: Specifies the start date and time and the interval to be subtracted from the start date and time, and returns the subtraction result.

  Parameters: See **Table 1-62**.

Table 1-62 subdate(expr,INTERVAL val unit) parameters

Parameter	Type	Description	Value Range
expr	Expression of the time type, text, datetime, date, or time	Specifies the start date and time.	See the value ranges of the corresponding types.
val	Integer, floating-point number, or character string	Specifies the interval to be subtracted.	See the value ranges of the corresponding types.
unit	Keyword		**YEAR**, **QUARTER**, **MONTH**, **WEEK**, **DAY**, **HOUR**, **MINUTE**, **SECOND**, and **MICROSECOND**. For details, see **Time Interval Expressions**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: TEXT, DATE, DATETIME, or TIME WITHOUT TIMEZONE

Example:

```
gaussdb=# SELECT subdate('2018-05-01', INTERVAL 1 YEAR);
  subdate
------------
 2017-05-01
(1 row)
```

NOTE

- This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.
- When working with the PREPARE statement, note that the second parameter of the subdate function is an interval expression. If this parameter is replaced by **$1**, you will encounter unexpected results. For example, running **PREPARE p1 AS SELECT subdate('2023-01-01 10:00:00', $1); EXECUTE p1(interval 13 hour);** will yield the unexpected result of **'2023-01-01 10:00:00'**.

- subtime(expr1, expr2)

  Description: Returns the time difference between **expr1** and **expr2**.

  Parameters: **expr1** is an expression of the TIMESTAMP WITHOUT TIME ZONE or TIME type, and **expr2** is a time expression.

  Return type: TEXT, TIMESTAMP WITHOUT TIME ZONE, or TIME. The return value type is related to the **expr1** type. If the two input parameters are of the TEXT type, the return value type is TEXT. If the two input parameters are parsed as TIMESTAMP WITHOUT TIME ZONE, the return value type is TIMESTAMP WITHOUT TIME ZONE. If the two input parameters are parsed as TIME, the return value type is TIME.

  Example:

  ```
  gaussdb=# SELECT subtime('2000-03-01 20:59:59', '22:58');
        subtime
  ---------------------
   2000-02-29 22:01:59
  (1 row)
  ```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- addtime(expr1, expr2)

  Description: Returns the sum of **expr1** and **expr2**. The format of the return value is the same as that of **expr1**.

  Parameters: **expr1** is an expression of the timestamp without time zone or time type, and **expr2** is a time expression.

  Return type: text, timestamp without time zone, or time. The return type is determined by the data type of **expr1**. If **expr1** is parsed as timestamp without time zone, the return type will also be timestamp without time zone. If it parsed as time, the return type will also be time.

  Example:

  ```
  gaussdb=# SELECT addtime('2000-03-01 20:59:59', '00:00:01');
        addtime
  ---------------------
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
2000-03-01 21:00:00
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- time_format(time, format)

  Description: Formats the **time** input parameter based on the format specified by **format**.

  Parameters:

  - The value type of **time** is expression of the time type, text, datetime, date, or time
  - **format** is of the text type. **Table 1-63** lists the supported formats.

    Table 1-63 Values and meanings of format

Value	Meaning
%f	Microsecond (000000–999999).
%H	Hour (00–23).
%h, %I	Hour (00–12).
%l	Hour (0–12).
%k	Hour (0–838).
%i	Minutes (00–59).
%p	AM or PM
%r	Time in 12-hour AM or PM format (hh:mm:ss AM/PM)
%S, %s	Seconds (00–59).
%T	Time in 24-hour format (hh:mm:ss)
%a, %b, %D, %j, %M, %U, %u, %V, %v, %W, %w, %X, %x	**NULL**.
%c, %e	0
%d, %m, %y	00
%Y	0000
%*Other characters*|*Other characters*, for example, %A/A	The character itself is returned, for example, A.
%*Single character*+string *s*	%*Single character* is parsed and then concatenated with s.

  Return type: text

  Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT time_format('25:30:30', '%T|%r|%H|%h|%I|%i|%S|%f|%p|%k');
                time_format
-----------------------------------------------
 25:30:30|01:30:30 AM|25|01|01|30|30|000000|AM|25
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- time_to_sec()

  Description: Converts the input parameter of the time type to the number of seconds.

  Parameter: The input parameter is a time expression, TEXT, DATETIME, DATE, or TIME. The time expression is parsed as TIME.

  Return type: int.

  Example:

```
gaussdb=# SELECT time_to_sec('00:00:01');
 time_to_sec
-------------
           1
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- timediff()

  Description: Calculates the difference between two time points and returns an interval.

  Parameters: two parameters, which can be represented as either a time expression or a type such as text, datetime, date, or time.

  Return type: TIME WITHOUT TIMEZONE

  Example:

```
gaussdb=# SELECT timediff(date'2022-12-30',20221229);
 timediff
----------
 24:00:00
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- timestampadd(unit, interval, datetime_expr)

  Description: Returns a new timestamp calculated by adding multiple intervals of a unit to **datetime_expr**.

  Parameters: See **Table 1-64**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-64 timestampadd parameters

Parameter	Type	Description	Value Range
unit	Keyword	Unit of the interval.	YEAR, QUARTER, MONTH, WEEK, DAY, HOUR, MINUTE, SECOND, MICROSECOND, Y, MM (month), D, H, M (minute), S, US, YRS, QTR, MON, HRS, MIN, YEARS, WEEKS, or HOURS.
interval	numeric	Interval.	See the value ranges of the corresponding types.
datetime_expr	Expression of the time type, text, datetime, date, or time	Date and time to be changed. If the value is of the text type, the return value type is text. If the value is of the time type, the return value type is time. In other cases, the return value type is datetime.	See the value ranges of the corresponding types.

Return type: DATETIME, TEXT, TIME WITHOUT TIMEZONE

Example:

```
gaussdb=# SELECT timestampadd(DAY,-2,'2022-07-27');
 timestampadd
--------------
 2022-07-25
(1 row)
```

NOTE

- This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.
- The actually registered function of timestampadd is timestamp_add. You can run commands such as **\df timestamp_add** to query the detailed input parameter and return value of the function.
- When working with the PREPARE statement, note that the first parameter of the timestampadd function is a keyword and cannot be replaced by **$1**, for example, **PREPARE p1 AS SELECT timestampadd($1, -2, '2023-01-01');**.

- to_days()

  Description: Returns the number of days since January 1, 0000.

  Parameter: The input parameter is a time expression, TEXT, DATETIME, DATE, or TIME. The time expression is parsed as DATE.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: bigint

Example:

```
gaussdb=# SELECT to_days('2000-1-1');
 to_days
---------
  730485
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- to_seconds()

  Description: Returns the number of seconds since the year 0 A.D.

  Parameter: The input parameter is a time expression, TEXT, DATETIME, DATE, or TIME. The time expression is parsed as datetime.

  Return type: bigint

  Example:

  ```
  gaussdb=# SELECT to_seconds('2009-11-29 13:43:32');
   to_seconds
  -------------
   63426721412
  (1 row)
  ```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**.

- unix_timestamp([date])

  Description: Returns a Unix timestamp representing the number of seconds since "1970-01-01 08:00" UTC. If there is no input parameter, the default value is the datetime timestamp when the function is called.

  Parameter: Expression of the time type, text, datetime, date, or time

  Return type: numeric

  Example:

  ```
  gaussdb=# SELECT unix_timestamp('2022-12-22');
   unix_timestamp
  ----------------
      1671638400
  (1 row)
  ```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7** and **b_format_dev_version** is set to **s1**.

- utc_date()

  Description: Returns the current UTC date of function execution as a value in YYYY-MM-DD format.

  Return type: date

  Example:

  ```
  gaussdb=# SELECT utc_date();
    utc_date
  ------------
   2023-08-10
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7** and **b_format_dev_version** is set to **s1**.

- utc_time([precision])

  Description: Returns the current UTC time in the format of *HH:MM:SS* when the function is executed.

  Parameter: Indicates the time precision. The value is of the int type and in the range [0,6]. The default value is **0**.

  Return type: time without time zone

  Example:

```
gaussdb=# SELECT utc_time();
 utc_time
----------
 11:47:53
(1 row)
```

  > **NOTE**
  >
  > This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7** and **b_format_dev_version** is set to **s1**.

- utc_timestamp([precision])

  Description: Returns the current UTC timestamp in the format of *YYYY-MM-DD HH:MM:SS* when the function is executed.

  Parameter: Indicates the time precision. The value is of the int type and in the range [0,6]. The default value is **0**.

  Return type: timestamp without time zone

  Example:

```
gaussdb=# SELECT utc_timestamp();
    utc_timestamp
---------------------
 2023-08-21 11:51:19
(1 row)
```

  > **NOTE**
  >
  > This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7** and **b_format_dev_version** is set to **s1**.

- week(date[, mode])

  Description: Returns the week of a date.

  Parameters:

  - **date**: specifies the date and time, which is of the time expression, text, datetime, date, or time type.
  - **Table 1-65** describes the optional parameter **mode**. The default value is **0**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-65 Description of mode

mode	First Day of a Week	Range	Which Week is the First Week
0	Sunday	0–53	Week containing Sunday in this year
1	Monday	0–53	Week containing four or more days in this year
2	Sunday	1–53	Week containing Sunday in this year
3	Monday	1–53	Week containing four or more days in this year
4	Sunday	0–53	Week containing four or more days in this year
5	Monday	0–53	Week containing Monday in this year
6	Sunday	1–53	Week containing four or more days in this year
7	Monday	1–53	Week containing Monday in this year

Return type: bigint

Example:

```
gaussdb=# SELECT week(date'2000-01-01', 1);
 week
------
    0
(1 row)

gaussdb=# SELECT week('2000-01-01', 2);
 week
------
   52
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7** and **b_format_dev_version** is set to **s1**.

- weekday()

  Description: Returns the working day index of a date, that is, Monday is 0, Tuesday is 1, Wednesday is 2, Thursday is 3, Friday is 4, Saturday is 5, and Sunday is 6.

  Parameter: Expression of the time type, text, datetime, date, or time

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: bigint

Example:

```
gaussdb=# SELECT weekday('1970-01-01 12:00:00');
 weekday
---------
       3
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7** and **b_format_dev_version** is set to **s1**.

- weekofyear(date)

  Description: Returns the calendar week of the date and time. The value ranges from 1 to 53. This function is equivalent to week(date, 3). For details, see **week(date[, mode])**.

  Parameter: date and time, which can be represented as a time expression or a type such as text, datetime, date, or time.

  Return type: bigint

  Example:

  ```
  gaussdb=# SELECT weekofyear('1970-05-22');
   weekofyear
  ------------
           21
  (1 row)
  ```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7** and **b_format_dev_version** is set to **s1**.

- year()

  Description: Extracts the year part of the date and time and returns the result.

  Parameter: The input parameter is the date and time to be extracted, which can be an expression of the time type, or of the text, datetime, date, or time type.

  Return type: int.

  Example:

  ```
  gaussdb=# SELECT year('23-05-22');
   year
  ------
   2023
  (1 row)
  ```

  NOTE

  This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7** and **b_format_dev_version** is set to **s1**.

- yearweek(date[, mode])

  Description: Returns the year and week of a date.

  Parameters:

  - **date**: specifies the date and time, which can be of the DATE, DATETIME, TIMESTAMP, TIME, TEXT, or NUMERIC type.
  - **Table 1-66** describes the optional parameter **mode**. The default value is **0**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-66 Description of mode

mode	First Day of a Week	Range	Which Week is the First Week
0	Sunday	0–53	Week containing Sunday in this year
1	Monday	0–53	Week containing four or more days in this year
2	Sunday	1–53	Week containing Sunday in this year
3	Monday	1–53	Week containing four or more days in this year
4	Sunday	0–53	Week containing four or more days in this year
5	Monday	0–53	Week containing Monday in this year
6	Sunday	1–53	Week containing four or more days in this year
7	Monday	1–53	Week containing Monday in this year

Return type: bigint

Example:

```
gaussdb=# SELECT yearweek(datetime'2000-01-01', 3);
 yearweek
----------
   199952
(1 row)
```

NOTE

This function is valid only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7** and **b_format_dev_version** is set to **s1**.

Table 1-67 shows the templates for truncating date/time values.

Table 1-67 Truncating date/time values

Item	Format	Description
Microsecond	MICROSECON	Truncates date/time values, accurate to the microsecond (000000–999999).
	US	
	USEC	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Item	Format	Description
	USECOND	
Millisecond	MILLISECON	Truncates date/time values, accurate to the millisecond (000–999).
	MS	
	MSEC	
	MSECOND	
Second	S	Truncates date/time values, accurate to the second (00–59).
	SEC	
	SECOND	
Minute	M	Truncates date/time values, accurate to the minute (00–59).
	MI	
	MIN	
	MINUTE	
Hour	H	Truncates date/time values, accurate to the hour (00–23).
	HH	
	HOUR	
	HR	
Day	D	Truncates date/time values, accurate to the day (01-01 to 12-31).
	DAY	
	DD	
	DDD	
	J	
Week	W	Truncates date/time values, accurate to the week (the first day of the current week).
	WEEK	
Month	MM	Truncates date/time values, accurate to the month (the first day of the current month).
	MON	
	MONTH	
Quarter	Q	Truncates date/time values, accurate to the quarter (the first day of the current quarter).
	QTR	
	QUARTER	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Item	Format	Description
Year	Y	Truncates date/time values, accurate to the year (the first day of the current year).
	YEAR	
	YR	
	YYYY	
Decade	DEC	Truncates date/time values, accurate to the decade (the first day of the current decade).
	DECADE	
Century	C	Truncates date/time values, accurate to the century (the first day of the current century).
	CC	
	CENT	
	CENTURY	
Millennium	MIL	Truncates date/time values, accurate to the millennium (the first day of the current millennium).
	MILLENNIA	
	MILLENNIUM	

Table 1-68 Parameters for time truncation and rounding

Item	Format	Description
Minute	M	Truncates or rounds off date/time values, accurate to the minute (00–59).
	MI	
	MIN	
	MINUTE	
Hour	H	Truncates or rounds off date/time values, accurate to the hour (00–23).
	HH	
	HOUR	
	HR	
	HH12	
	HH24	
Day	DD	Truncates or rounds off date/time values, accurate to the day (01-01 to 12-31).
	DDD	
	J	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Item	Format	Description
ISO week	IW	Truncated or rounded off, accurate to week (the first day of the week is Monday).
Week	DAY	Truncated or rounded off, accurate to week (the first day of the week is Sunday).
	DY	
	D	
Week of the month	W	Truncated or rounded off, accurate to week (the first day of the week is the first day of the month).
Week of the year	WW	Truncated or rounded off, accurate to week (the first day of the week is the first day of the year).
Month	MM	Truncated or rounded off, accurate to month (the first day of the month).
	MON	
	MONTH	
	RM	
Quarter	Q	Truncated or rounded off, accurate to quarter (the first day of the quarter).
	QTR	
	QUARTER	
Year	Y	Truncated or rounded off, accurate to year (the first day of the current year).
	YEAR	
	YR	
	YYYY	
	SYYYY	
	YYY	
	YY	
	SYEAR	
Decade	DEC	Truncated or rounded off, accurate to decade (the first day of the current decade).
	DECADE	
Century	C	Truncated or rounded off, accurate to the century (the first day of the century).
	CC	
	CENT	
	CENTURY	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Item	Format	Description
	SCC	
Millennium	MIL	Truncated or rounded off, accurate to millennium (the first day of the millennium).
	MILLENNIA	
	MILLENNIUM	

NOTE

The behaviors of **Table 1-68** are valid only when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s1** in an ORA-compatible database.

- timestamp_diff(text, timestamp, timestamp)

  Description: Calculates the difference between two timestamps and truncates the difference to the precision specified by text.

  Return type: bigint

  Example:

```
gaussdb=# SELECT timestamp_diff('year','2018-01-01','2020-04-01');
 timestamp_diff
----------------
              2
(1 row)
gaussdb=# SELECT timestamp_diff('month','2018-01-01','2020-04-01');
 timestamp_diff
----------------
             27
(1 row)
gaussdb=# SELECT timestamp_diff('quarter','2018-01-01','2020-04-01');
 timestamp_diff
----------------
              9
(1 row)
gaussdb=# SELECT timestamp_diff('week','2018-01-01','2020-04-01');
 timestamp_diff
----------------
            117
(1 row)
gaussdb=# SELECT timestamp_diff('day','2018-01-01','2020-04-01');
 timestamp_diff
----------------
            821
(1 row)
gaussdb=# SELECT timestamp_diff('hour','2018-01-01 10:10:10','2018-01-01 12:12:12');
 timestamp_diff
----------------
              2
(1 row)
gaussdb=# SELECT timestamp_diff('minute','2018-01-01 10:10:10','2018-01-01 12:12:12');
 timestamp_diff
----------------
            122
(1 row)
gaussdb=# SELECT timestamp_diff('second','2018-01-01 10:10:10','2018-01-01 10:12:12');
 timestamp_diff
----------------
            122
(1 row)
gaussdb=# SELECT timestamp_diff('microsecond','2018-01-01 10:10:10','2018-01-01 10:12:12');
 timestamp_diff
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
----------------
    122000000
(1 row)
```

TIMESTAMPDIFF

TIMESTAMPDIFF(unit , timestamp_expr1, timestamp_expr2)

Description: Returns the result of **timestamp_expr2** – **timestamp_expr1** in the specified unit. This function is equivalent to **timestamp_diff(text, timestamp, timestamp)**.

Parameters: **timestamp_expr1** and **timestamp_expr2** are of the time expression, text, datetime, date, or time type. **unit** specifies the unit of the difference between two dates.

Return type: bigint

NOTE

This function is valid only when the MySQL-compatible database is used (that is, **sql_compatibility** is set to **'MYSQL'**).

- When **b_format_version** is set to **5.7** and **b_format_dev_version** is set to **s1**, the actually called function is b_timestampdiff.
- If the parameters **b_format_version** and **b_format_dev_version** are not set, the actually called function is timestamp_diff. You can run commands such as **\df b_timestampdiff** to query the detailed input parameters and return values of the function.

- year

  Year.

  ```
  gaussdb=# SELECT TIMESTAMPDIFF(YEAR, '2018-01-01', '2020-01-01');
   timestamp_diff
  ----------------
               2
  (1 row)
  ```

- quarter

  Quarter.

  ```
  gaussdb=# SELECT TIMESTAMPDIFF(QUARTER, '2018-01-01', '2020-01-01');
   timestamp_diff
  ----------------
               8
  (1 row)
  ```

- month

  Month.

  ```
  gaussdb=# SELECT TIMESTAMPDIFF(MONTH, '2018-01-01', '2020-01-01');
   timestamp_diff
  ----------------
              24
  (1 row)
  ```

- week

  Week.

  ```
  gaussdb=# SELECT TIMESTAMPDIFF(WEEK, '2018-01-01', '2020-01-01');
   timestamp_diff
  ----------------
             104
  (1 row)
  ```

- day

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Day.

```
gaussdb=# SELECT TIMESTAMPDIFF(DAY, '2018-01-01', '2020-01-01');
 timestamp_diff
----------------
            730
(1 row)
```

- hour

  Hour.

  ```
  gaussdb=# SELECT TIMESTAMPDIFF(HOUR, '2020-01-01 10:10:10', '2020-01-01 11:11:11');
   timestamp_diff
  ----------------
                1
  (1 row)
  ```

- minute

  Minute.

  ```
  gaussdb=# SELECT TIMESTAMPDIFF(MINUTE, '2020-01-01 10:10:10', '2020-01-01 11:11:11');
   timestamp_diff
  ----------------
               61
  (1 row)
  ```

- second

  Second.

  ```
  gaussdb=# SELECT TIMESTAMPDIFF(SECOND, '2020-01-01 10:10:10', '2020-01-01 11:11:11');
   timestamp_diff
  ----------------
             3661
  (1 row)
  ```

- microseconds

  The second field, including fractional parts, is multiplied by 1,000,000.

  ```
  gaussdb=# SELECT TIMESTAMPDIFF(MICROSECOND, '2020-01-01 10:10:10.000000', '2020-01-01
  10:10:10.111111');
   timestamp_diff
  ----------------
           111111
  (1 row)
  ```

EXTRACT

EXTRACT(field FROM source)

Description: The extract function retrieves fields such as year or hour from date/time values.

Parameters:

- **source** is a value expression of type timestamp, time, or interval. (Expressions of type date can be converted to timestamp and can therefore be used as well.)
- **field** is an identifier or string that selects what field to extract from the source value.

Return type: double precision

The following are valid **field** names:

- century

  Century.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The first century starts at 0001-01-01 00:00:00 A.D. This definition applies to all Gregorian calendar countries. There is no century numbered 0. The century goes from the 1st century B.C. to the 1st century A.D.

```
gaussdb=# SELECT EXTRACT(CENTURY FROM TIMESTAMP '2000-12-16 12:21:13');
 date_part
-----------
        20
(1 row)
```

- day
  - For **timestamp** values, the day (of the month) field (1–31)

    ```
    gaussdb=# SELECT EXTRACT(DAY FROM TIMESTAMP '2001-02-16 20:38:40');
     date_part
    -----------
            16
    (1 row)
    ```

  - For **interval** values, the number of days

    ```
    gaussdb=# SELECT EXTRACT(DAY FROM INTERVAL '40 days 1 minute');
     date_part
    -----------
            40
    (1 row)
    ```

- decade

  Year divided by 10

  ```
  gaussdb=# SELECT EXTRACT(DECADE FROM TIMESTAMP '2001-02-16 20:38:40');
   date_part
  -----------
        200
  (1 row)
  ```

- dow

  Day of the week as Sunday (**0**) to Saturday (**6**)

  ```
  gaussdb=# SELECT EXTRACT(DOW FROM TIMESTAMP '2001-02-16 20:38:40');
   date_part
  -----------
           5
  (1 row)
  ```

- doy

  Day of the year (1–365 or 366)

  ```
  gaussdb=# SELECT EXTRACT(DOY FROM TIMESTAMP '2001-02-16 20:38:40');
   date_part
  -----------
          47
  (1 row)
  ```

- epoch
  - For **timestamp with time zone** values, the number of seconds since 1970-01-01 00:00:00-00 UTC (can be negative).

    For **date** and **timestamp** values, the number of seconds since 1970-01-01 00:00:00-00 local time.

    For **interval** values, the total number of seconds in the interval.

    ```
    gaussdb=# SELECT EXTRACT(EPOCH FROM TIMESTAMP WITH TIME ZONE '2001-02-16
    20:38:40.12-08');
      date_part
    --------------
     982384720.12
    (1 row)
    gaussdb=# SELECT EXTRACT(EPOCH FROM INTERVAL '5 days 3 hours');
     date_part
    -----------
    ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
   442800
(1 row)
```

  - Way to convert an epoch value back to a timestamp

```
gaussdb=# SELECT TIMESTAMP WITH TIME ZONE 'epoch' + 982384720.12 * INTERVAL '1
second' AS RESULT;
          result
----------------------------
 2001-02-17 12:38:40.12+08
(1 row)
```

- hour

  Hour (0–23).

```
gaussdb=# SELECT EXTRACT(HOUR FROM TIMESTAMP '2001-02-16 20:38:40');
 date_part
-----------
        20
(1 row)
```

- isodow

  Day of a week (1–7). Monday is 1 and Sunday is 7.

  NOTE

  The value of **isodow** is identical to **dow** except for Sunday.

```
gaussdb=# SELECT EXTRACT(ISODOW FROM TIMESTAMP '2001-02-18 20:38:40');
 date_part
-----------
         7
(1 row)
```

- isoyear

  The ISO 8601 year that the date falls in (not applicable to intervals).

  The first week of the ISO year starts on Monday and must contain January 4. Therefore, the ISO year corresponding to the date in early January or late December may be different from the calendar year. For details, see **week**.

```
gaussdb=# SELECT EXTRACT(ISOYEAR FROM DATE '2006-01-01');
 date_part
-----------
      2005
(1 row)
gaussdb=# SELECT EXTRACT(WEEK FROM TIMESTAMP '2006-01-01 00:00:40');
 date_part
-----------
        52
(1 row)
gaussdb=# SELECT EXTRACT(ISOYEAR FROM DATE '2006-01-02');
 date_part
-----------
      2006
(1 row)
gaussdb=# SELECT EXTRACT(WEEK FROM TIMESTAMP '2006-01-02 00:00:40');
 date_part
-----------
         1
(1 row)
```

- microseconds

  The second field, including fractional parts, is multiplied by 1,000,000.

```
gaussdb=# SELECT EXTRACT(MICROSECONDS FROM TIME '17:12:28.5');
 date_part
-----------
  28500000
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- millennium

  Millennium.

  Years in the 1900s are in the second millennium. The third millennium started from January 1, 2001.

  ```
  gaussdb=# SELECT EXTRACT(MILLENNIUM FROM TIMESTAMP '2001-02-16 20:38:40');
   date_part
  -----------
           3
  (1 row)
  ```

- milliseconds

  Second field, including fractional parts, is multiplied by 1000. Note that this includes full seconds.

  ```
  gaussdb=# SELECT EXTRACT(MILLISECONDS FROM TIME '17:12:28.5');
   date_part
  -----------
       28500
  (1 row)
  ```

- minute

  Minute (0–59).

  ```
  gaussdb=# SELECT EXTRACT(MINUTE FROM TIMESTAMP '2001-02-16 20:38:40');
   date_part
  -----------
          38
  (1 row)
  ```

- month

  If **source** is **timestamp**, it represents the number of the month within the year (1–12).

  ```
  gaussdb=# SELECT EXTRACT(MONTH FROM TIMESTAMP '2001-02-16 20:38:40');
   date_part
  -----------
           2
  (1 row)
  ```

  If **source** is **interval**, it represents the number of months, modulo 12 (0–11).

  ```
  gaussdb=# SELECT EXTRACT(MONTH FROM INTERVAL '2 years 13 months');
   date_part
  -----------
           1
  (1 row)
  ```

- quarter

  The quarter of the year (1–4) that the day is in.

  ```
  gaussdb=# SELECT EXTRACT(QUARTER FROM TIMESTAMP '2001-02-16 20:38:40');
   date_part
  -----------
           1
  (1 row)
  ```

- second

  The seconds field, including fractional parts (0–59).

  ```
  gaussdb=# SELECT EXTRACT(SECOND FROM TIME '17:12:28.5');
   date_part
  -----------
        28.5
  (1 row)
  ```

- timezone

  Time zone offset from UTC, measured in seconds. Positive values correspond to time zones east of UTC, negative values to zones west of UTC.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- timezone_hour

  Hour component of the time zone offset.

- timezone_minute

  Minute component of the time zone offset.

- week

  Week number of the date in the year to which the date belongs. By definition (ISO 8601), the first week of a year contains January 4 of that year. (The ISO-8601 week starts on Monday.) In other words, the first Thursday of a year is in week 1 of that year.

  Because of this, it is possible for early January dates to be part of the 52nd or 53rd week of the previous year, and late December dates to be part of the 1st week of the next year. For example, 2006-01-01 is the 52nd week of 2005, and 2006-01-02 is the first week of 2006. You are advised to use the columns **isoyear** and **week** together to ensure consistency.

  ```
  gaussdb=# SELECT EXTRACT(ISOYEAR FROM DATE '2006-01-01');
   date_part
  -----------
       2005
  (1 row)
  gaussdb=# SELECT EXTRACT(WEEK FROM TIMESTAMP '2006-01-01 00:00:40');
   date_part
  -----------
         52
  (1 row)
  gaussdb=# SELECT EXTRACT(ISOYEAR FROM DATE '2006-01-02');
   date_part
  -----------
       2006
  (1 row)
  gaussdb=# SELECT EXTRACT(WEEK FROM TIMESTAMP '2006-01-02 00:00:40');
   date_part
  -----------
          1
  (1 row)
  ```

- year

  Year field.

  ```
  gaussdb=# SELECT EXTRACT(YEAR FROM TIMESTAMP '2001-02-16 20:38:40');
   date_part
  -----------
       2001
  (1 row)
  ```

  NOTE

  When **sql_compatibility** is set to **'ORA'**:

  - When the GUC parameter **mapping_date_to_datea** is set to **on**, the database parses the date type in the input parameter to the datea type. In this case, the unit can only be year, month, or day. If other units are specified, an error is reported.
  - When the input parameter is of the datea type, the unit can only be year, month, or day regardless of whether the GUC parameter **mapping_date_to_datea** is set to **on** or **off**. If other units are specified, an error is reported.

date_part

date_part('field', source)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: The date_part function is implemented based on the traditional Ingres function (this function is equivalent to the SQL standard function extract).

Parameter: The **field** parameter is a character string instead of a name. The valid field names are the same as those for **extract**. For details, see **EXTRACT**.

Example:

```
gaussdb=# SELECT date_part('day', TIMESTAMP '2001-02-16 20:38:40');
 date_part
-----------
        16
(1 row)
gaussdb=# SELECT date_part('hour', INTERVAL '4 hours 3 minutes');
 date_part
-----------
         4
(1 row)
```

Table 1-69 specifies the formats for formatting date and time values.

Table 1-69 Formats for formatting date and time

Category	Format	Description
Hour	HH	Hour of a day (01–12).
	HH12	Hour of a day (01–12).
	HH24	Hour of a day (00–23).
Minute	MI	Minute (00–59).
Second	SS	Second (00–59).
	FF	Microsecond (000000–999999).
	FF1	Microsecond (0–9).
	FF2	Microsecond (00–99).
	FF3	Microsecond (000–999).
	FF4	Microsecond (0000–9999).
	FF5	Microsecond (00000–99999).
	FF6	Microsecond (000000–999999).
	SSSSS	Second after midnight (0–86399).
Morning and afternoon	AM or A.M.	Morning identifier
	PM or P.M.	Afternoon identifier
Year	Y,YYY	Year with a comma (with four digits or more)
	SYYYY	Year with four digits BC
	YYYY	Year (with four digits or more)
	YYY	Last three digits of a year

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Format	Description
	YY	Last two digits of a year
	Y	Last one digit of a year
	IYYY	ISO year (with four digits or more)
	IYY	Last three digits of an ISO year
	IY	Last two digits of an ISO year
	I	Last one digit of an ISO year
	RR	Last two digits of a year (A year of the 20th century can be stored in the 21st century.)
	RRRR	Capable of receiving a year with four digits or two digits. If there are 2 digits, the value is the same as the returned value of RR. If there are 4 digits, the value is the same as YYYY.
	• BC or B.C. • AD or A.D.	Era indicator (**BC** or **AD**).
Month	MONTH	Full name of a month in uppercase. (9 characters are filled in if the value is empty.)
	MON	Month in abbreviated format in uppercase (with three characters).
	MM	Month number (01–12).
	RM	Month in Roman numerals (I–XII; I=JAN) and uppercase.
Day	DAY	Full name of a date in uppercase. (9 characters are filled in if the value is empty.)
	DY	Day in abbreviated format in uppercase (with three characters).
	DDD	Day of a year (001–366).
	DD	Day of a month (01–31).
	D	Day of a week (1–7), where **1** represents Sunday.
Week	W	Week of a month (1–5). (The first week starts from the first day of the month.)
	WW	Week of a year (1–53). (The first week starts from the first day of the year.)
	IW	Week in an ISO year (The first Thursday is in the first week.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Format	Description
Century	CC	Century (with two digits) (The 21st century starts from 2001-01-01.)
Julian date	J	Julian day (number of days after January 1, 4713 B.C.)
Quarter	Q	Quarter

When **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1** in an ORA-compatible database, the date and time will be formatted in the specified format.

Table 1-70 New formats for formatting date and time

Item	Format	Description
Century	SCC	Century. A hyphen (-) will be displayed before BC years.
Year	SYYYY	Returns a numeric year. A hyphen (-) will be displayed before BC years.
	RR	Returns the two-digit year of a date.
	RRRR	Returns the four-digit year of a date.
	YEAR	Returns the year of the character type.
	SYEAR	Returns the year of the character type. A hyphen (-) will be displayed before BC years.
Date Format	DL	Returns the date in the specified long date format.
	DS	Returns the date in the specified short date format.
	TS	Returns the time in the specified time format.
Second	FF7	Microsecond (0000000–9999990).
	FF8	Microsecond (00000000–99999900).
	FF9	Microsecond (000000000–999999000).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When **sql_compatibility** is set to **'ORA'**:
  - When the GUC parameter **mapping_date_to_datea** is set to **on**, the database parses the date type in the input parameter to the datea type. In this case, the format can only be year, month, or day. If other formats are specified, an error is reported.
  - When the input parameter is of the datea type, the format can only be year, month, or day regardless of whether the GUC parameter **mapping_date_to_datea** is set to **on** or **off**. If other formats are specified, an error is reported.
- In the scenario where this function is in an ORA-compatible database, the value of **a_format_version** is **10c**, and that of **a_format_dev_version** is **s1**:
  - The **to_date** and **to_timestamp** functions support the FX pattern (the input strictly corresponds to a pattern) and the X pattern (decimal point).
  - The input pattern can appear only once, indicating that the patterns of the same information cannot appear at the same time. For example, SYYYY and BC cannot be used together.
  - Formats are case-insensitive.
  - You are advised to use a separator between the input and the format. Otherwise, the behavior may not be compatible with database O.
  - GaussDB uses a strict matching policy. Each format element in the **fmt** string must have a corresponding content in the input string. Otherwise, an error is reported. Oracle Database uses a loose matching policy for optional elements (such as FF) in the format model when converting a string to a date. If the input string does not contain the optional elements, the system automatically skips them and continues to match subsequent elements. This difference affects the behavior of the to_date, to_timestamp, and to_timestamp_tz functions when **fmt** is not specified or contains optional format elements such as FF.
- In the table, the rules for RR to calculate years are as follows:
  - If the range of the input two-digit year is between 00 and 49:

    If the last two digits of the current year are between 00 and 49, the first two digits of the returned year are the same as the first two digits of the current year.

    If the last two digits of the current year are between 50 and 99, the first two digits of the returned year are equal to the first two digits of the current year plus 1.
  - If the range of the input two-digit year is between 50 and 99:

    If the last two digits of the current year are between 00 and 49, the first two digits of the returned year are equal to the first two digits of the current year minus 1.

    If the last two digits of the current year are between 50 and 99, the first two digits of the returned year are the same as the first two digits of the current year.

time_window

time_window(window_width interval, time timestamptz, [origin timestamptz])

Description: Performs modulo operation on the difference between the specified timestamp column and the original timestamp based on the specified time window size.

Parameters: See **Table 1-71**.

Return type: timestamptz

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-71 time_window parameters

Parameter	Type	Description
window_width	interval	Time window.
time	timestamp(tz)	Timestamp column.
origin	timestamp(tz)	Original timestamp. The default value is **2000-01-03 00:00:00**.

NOTE

- The unit of **window_width** can be microsecond, millisecond, second, minute, hour, day, week, or a positive number. The default unit is day.
- The values of **window_width** and **time** are correct and cannot be empty.

Example:

```
gaussdb=# SELECT
    time_window(interval '1 min', time),
    time_window(interval '3 min', time,  '2019-07-11 00:02:05'::timestamptz)
        FROM
    (VALUES ('2019-07-12 00:00:00'::timestamptz),
           ('2019-07-12 00:01:01'::timestamptz)) v(time);

     time_window       |      time_window
-------------------------+-------------------------
 2019-07-12 00:00:00+08 | 2019-07-11 23:59:05+08
 2019-07-12 00:01:00+08 | 2019-07-11 23:59:05+08
(2 rows)
```

1.6.9 Type Conversion Functions

Function Type Conversion

- cash_words(money)

  Description: Converts the money type into the text type and then outputs the result.

  Example:

  ```
  gaussdb=# SELECT cash_words('1.23');
              cash_words
  -----------------------------------
   One dollar and twenty three cents
  (1 row)
  ```

- convert(expr, type)

  Description: Converts **expr** into the type specified by **type**.

  Parameters: The first parameter is an arbitrary value, and the second parameter is a type name.

  Return type: the input type.

  Example:

  ```
  gaussdb=# SELECT convert(12.5, text);
   text
  ------
   12.5
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> This function takes effect only in MySQL-compatible mode.

- cast(x as y [DEFAULT z ON CONVERSION ERROR][,fmt])

  Description: Converts **x** into the type specified by **y**. When **sql_compatibility** is **'MYSQL'**, setting **b_format_version** to **'5.7'** and **b_format_dev_version** to **'s1'** will result in **x** being converted into the varchar type when **y** is of the char type.

  Parameters:

  – **x** represents the data to be converted. Its type and value must be convertible into the target type **y**.

  – DEFAULT z ON CONVERSION ERROR, an optional syntax for the first parameter, specifies that if **x** fails to be converted into the type specified by **y**, **z** will be converted into that type instead.

  – The optional parameter **fmt** can be specified when **y** is any of the following types:

    ▪ int1/int2/int4/int8/int16/float4/float8/numeric: The **fmt** parameter has the same effect as the to_number(expr [,fmt]) function.

    ▪ date/timestamp/timestamp with time zone/datea (effective only in ORA-compatible mode): The **fmt** parameter has the same effect as the to_date(string [,fmt]), to_timestamp(string [,fmt]), and to_timestamp_tz(string [,fmt]) functions.

  Example:

```
gaussdb=# SELECT cast('22-oct-1997' as timestamp);
     timestamp
---------------------
 1997-10-22 00:00:00
(1 row)

gaussdb=# SELECT cast('22-ocX-1997' as timestamp DEFAULT '22-oct-1997' ON CONVERSION
ERROR, 'DD-Mon-YYYY');
     timestamp
---------------------
 1997-10-22 00:00:00
(1 row)

gaussdb=# CREATE DATABASE gaussdb_m  WITH dbcompatibility  'MYSQL';
gaussdb=# \c gaussdb_m
-- Set compatible version control parameters.
gaussdb_m=# SET b_format_version='5.7';
gaussdb_m=# SET b_format_dev_version='s1';
gaussdb_m=# SELECT cast('aaa' as char);
 varchar
---------
 aaa
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This function supports the optional syntax DEFAULT z ON CONVERSION ERROR and optional parameter **fmt** when **a_format_version** is **10c** and **a_format_dev_version** is **s1**.
- In ORA-compatible databases where the **mapping_date_to_datea** parameter is enabled and **fmt** is not specified, **x** will be converted into the datea type based on **nls_date_format**.
- In ORA-compatible databases where the **mapping_date_to_datea** parameter is enabled and **fmt** is not specified, the datea type will be converted into the character type based on **nls_date_format**.
- When **y** is of the datea type, **x** will only accept the following types: text, bpchar, varchar, clob, nvarchar2, timestamp, and timestamp with time zone.

- cast(x AS {SIGNED | UNSIGNED} [INT | INTEGER])

  Description: Converts **x** into the BIGINT SIGNED or BIGINT UNSIGNED type.

  Parameter: **x** represents the data to be converted. Its type and value must be convertible into the target type INT.

  Return type: BIGINT SIGNED or BIGINT UNSIGNED.

  Example:

```
gaussdb=# SELECT CAST(12 AS UNSIGNED);
 uint8
-------
 12
(1 row)
```

- hextoraw(text)

  Description: Converts a hexadecimal character string into the raw type.

  Return type: raw.

  Example:

```
gaussdb=# SELECT hextoraw('7D');
 hextoraw
----------
 7D
(1 row)
```

- numtoday(numeric)

  Description: Converts a numeric value into a timestamp in the specified format.

  Return type: timestamp.

  Example:

```
gaussdb=# SELECT numtoday(2);
 numtoday
----------
 2 days
(1 row)
```

- rawtohex(string)

  Description: Converts a string in binary format into hexadecimal format. The result is the ASCII code of the input characters in hexadecimal format.

  Return type: varchar.

  Example:

```
gaussdb=# SELECT rawtohex('1234567');
    rawtohex
----------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 31323334353637
(1 row)
```

- to_blob(raw)

  Description: Converts the RAW type into the BLOB type.

  Return type: blob.

  Example:

  ```
  gaussdb=# SELECT to_blob('0AADD343CDBBD'::RAW(10));
     to_blob
  ----------------
   00AADD343CDBBD
  (1 row)
  ```

  NOTE

  The to_blob function is supported when **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- to_bigint(varchar)

  Description: Converts the character type into the bigint type.

  Return type: bigint.

  Example:

  ```
  gaussdb=# SELECT to_bigint('123364545554455');
     to_bigint
  ----------------
   123364545554455
  (1 row)
  ```

- to_binary_double(expr)

  Description: Converts **expr** into the float8 type.

  Parameter: **expr** accepts numeric values of the number, float4, or float8 type, as well as character strings that can be implicitly converted into numeric values.

  Return type: float8.

  Example:

  ```
  gaussdb=# SELECT to_binary_double('12345678');
   to_binary_double
  ------------------
          12345678
  (1 row)
  ```

  NOTE

  The to_binary_double function is supported when **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- to_binary_double(expr, fmt)

  Description: Converts **expr** into the float8 type in the format specified by **fmt**.

  Parameters: **expr** and **fmt** accept character strings of the char, nchar, varchar2, or nvarchar2 type. **expr** also accepts numeric values that can be implicitly converted into character strings.

  Return type: float8.

  Example:

  ```
  gaussdb=# SELECT to_binary_double('1,2,3', '9,9,9');
   to_binary_double
  ------------------
                123
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> NOTE
>
> The to_binary_double function is supported when **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- to_binary_double(expr DEFAULT return_value ON CONVERSION ERROR)

  Description: Converts **expr** into the float8 type. On failure, the default value **return_value** will be returned.

  Parameter: **expr** accepts numeric values of the number, float4, or float8 type, as well as numeric values that can be implicitly converted into character strings. If **expr** is not a numeric or string type, an error will be reported.

  Return type: float8.

  Example:

```
gaussdb=# SELECT to_binary_double(1e2 DEFAULT 12 ON CONVERSION ERROR);
 to_binary_double
------------------
              100
(1 row)

gaussdb=# SELECT to_binary_double('aa' DEFAULT 12 ON CONVERSION ERROR);
 to_binary_double
------------------
               12
(1 row)
```

> NOTE
>
> The to_binary_double function is supported when **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- to_binary_double(expr DEFAULT return_value ON CONVERSION ERROR, fmt)

  Description: Converts **expr** into the float8 type in the format specified by **fmt**. On failure, the default value **return_value** will be returned.

  Parameters: **expr** and **fmt** accept character strings of the char, nchar, varchar2, or nvarchar2 type. **expr** also accepts numeric values that can be implicitly converted into character strings.

  Return type: float8.

  Example:

```
gaussdb=# SELECT to_binary_double('12-' DEFAULT 10 ON CONVERSION ERROR, '99S');
 to_binary_double
------------------
              -12
(1 row)

gaussdb=# SELECT to_binary_double('aa-' DEFAULT 12 ON CONVERSION ERROR, '99S');
 to_binary_double
------------------
               12
(1 row)
```

> NOTE
>
> The to_binary_double function is supported when **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- to_binary_float(expr)

  Description: Converts **expr** into the float4 type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter: **expr** accepts numeric values of the number, float4, or float8 type, as well as character strings that can be implicitly converted into numeric values.

Return type: float4.

Example:

```
gaussdb=# SELECT to_binary_float('12345678');
 to_binary_float
------------------
     1.23457e+07
(1 row)
```

NOTE

The to_binary_float function is supported when **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- to_binary_float(expr, fmt)

  Description: Converts **expr** into the float4 type in the format specified by **fmt**.

  Parameters: **expr** and **fmt** accept character strings of the char, nchar, varchar2, or nvarchar2 type. **expr** also accepts numeric values that can be implicitly converted into character strings.

  Return type: float4.

  Example:

```
gaussdb=# SELECT to_binary_float('1,2,3', '9,9,9');
 to_binary_float
------------------
             123
(1 row)
```

NOTE

The to_binary_float function is supported when **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- to_binary_float(expr DEFAULT return_value ON CONVERSION ERROR)

  Description: Converts **expr** into the float4 type. On failure, the default value **return_value** will be returned.

  Parameter: **expr** accepts numeric values of the number, float4, or float8 type, as well as numeric values that can be implicitly converted into character strings. If **expr** is not a numeric or string type, an error will be reported.

  Return type: float4.

  Example:

```
gaussdb=# SELECT to_binary_float(1e2 DEFAULT 12 ON CONVERSION ERROR);
 to_binary_float
------------------
             100
(1 row)

gaussdb=# SELECT to_binary_float('aa' DEFAULT 12 ON CONVERSION ERROR);
 to_binary_float
------------------
              12
(1 row)
```

NOTE

The to_binary_float function is supported when **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- to_binary_float(expr DEFAULT return_value ON CONVERSION ERROR, fmt)

  Description: Converts **expr** into the float4 type in the format specified by **fmt**. On failure, the default value **return_value** will be returned.

  Parameters: **expr** and **fmt** accept character strings of the char, nchar, varchar2, or nvarchar2 type. **expr** also accepts numeric values that can be implicitly converted into character strings.

  Return type: float4.

  Example:

```
gaussdb=# SELECT to_binary_float('12-' DEFAULT 10 ON CONVERSION ERROR, '99S');
 to_binary_float
------------------
            -12
(1 row)

gaussdb=# SELECT to_binary_float('aa-' DEFAULT 12 ON CONVERSION ERROR, '99S');
 to_binary_float
------------------
             12
(1 row)
```

  NOTE

  The to_binary_float function is supported when **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- to_char(datetime/interval [, fmt])

  Description: Converts a DATETIME or INTERVAL value of the DATE, DATEA (effective only in ORA-compatible mode), TIMESTAMP, TIMESTAMP WITH TIME ZONE, or TIMESTAMP WITH LOCAL TIME ZONE type into the TEXT type in the format specified by **fmt** (a template used to format date and time values).

  – The optional parameter **fmt** accepts the following types: date, time, week, quarter, and century. Each type comes with its own distinct template. These templates can be combined as needed. Commonly used templates include HH, MI, SS, YYYY, MM, and DD. For details, see **Table 1-69**.

  – **fmt** can include a modifier, typically "FM", to help identify leading zeros or trailing spaces.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_char(current_timestamp,'HH12:MI:SS');
 to_char
----------
 10:19:26
(1 row)
gaussdb=# SELECT to_char(current_timestamp,'FMHH12:FMMI:FMSS');
 to_char
----------
 10:19:46
(1 row)
-- When mapping_date_to_datea is enabled in ORA-compatible mode, current_date returns the DATEA type.
test_db_a=# SELECT to_char(current_date);
  to_char
------------
 05-11-2024
(1 row)
test_db_a=# SELECT to_char(current_date, 'YYYY-MM-DD HH24:MI:SS AD');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
      to_char
------------------------
 2024-11-05 14:07:21 AD
(1 row)
```

> **NOTE**
>
> In ORA-compatible databases where the **mapping_date_to_datea** parameter is enabled and **fmt** is not specified, the datea type will be converted into the text type based on **nls_date_format**.

- to_char(double precision/real, text)

  Description: Converts a floating-point value into a character string in the specified format.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_char(125.8::real, '999D99');
 to_char
---------
  125.80
(1 row)
```

- to_char(numeric/smallint/integer/bigint/double precision/real[, fmt])

  Description: Converts an integer or floating-point value into a character string in the specified format.

  - The optional parameter **fmt** accepts the following types: decimal character, digit grouping separator, plus/minus sign, and currency symbol. Each type comes with its own distinct template. These templates can be combined as needed. Commonly used templates include 9, 0, thousands separator (,), and decimal point (.). For details, see **Table 1-72**.
  - A template can include a modifier like "FM". Be aware that if a 0 is specified in the template, "FM" will not be able to identify the 0 in the output.
  - To convert an integer value into a hexadecimal string, you can use the template X or x.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_char(1485,'9,999');
 to_char
---------
  1,485
(1 row)
gaussdb=# SELECT to_char( 1148.5,'9,999.999');
  to_char
------------
  1,148.500
(1 row)
gaussdb=# SELECT to_char(148.5,'990999.909');
   to_char
-------------
    0148.500
(1 row)
gaussdb=# SELECT to_char(123,'XXX');
 to_char
---------
   7B
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-72 Formats for the number type

Format	Description
, (comma)	Returns the digit grouping (thousands) separator.
. (period)	Returns the decimal point.
$	Returns the US dollar sign ($) in the specified position.
0	Returns leading zeros.
9	Returns a value with the specified number of digits.
B	Returns blanks when the integer part is 0.
C	Returns the currency symbol (depending on the locale settings).
D	Returns the decimal character (depending on the locale settings).
EEEE	Returns a value in scientific notation.
G	Returns the digit grouping separator (depending on the locale settings).
L	Returns the currency symbol (depending on the locale settings).
MI	Returns the minus sign in the specified position (if the number is less than 0).
PR	Returns a negative value in <angle brackets>.
RN	Returns a Roman numeral (for any input between 1 and 3,999).
S	Returns a signed number (depending on the locale settings).
TM	Returns a value in standard or scientific notation.
TM9	Returns a value in standard or scientific notation.
TME	Returns a value in standard or scientific notation.
U	Returns the dual currency symbol (depending on the locale settings).
V	Shifts the specified number of digits (decimal).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Format	Description
PL	Returns the plus sign in the specified position (if the number is greater than 0).
SG	Returns the plus/minus sign in the specified position.
TH / th	Returns the suffix for an ordinal number.

NOTE

This function supports the $, C, TM, TM9, TME, and U formats when **a_format_version** is **10c** and **a_format_dev_version** is **s1**. However, it does not support TH, PL, or SG.

- to_char(interval, text)

  Description: Converts a time interval value into a character string in the specified format.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_char(interval '15h 2m 12s', 'HH24:MI:SS');
 to_char
----------
 15:02:12
(1 row)
```

- to_char(integer, text)

  Description: Converts an integer value into a character string in the specified format.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_char(125, '999');
 to_char
---------
  125
(1 row)
```

- to_char(set)

  Description: Converts a value of the SET type into a character string. The SET type is currently unsupported in distributed mode.

  Return type: text.

- to_char(numeric, text)

  Description: Converts a numeric value into a character string in the specified format.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_char(-125.8, '999D99S');
 to_char
---------
 125.80-
(1 row)
```

- to_char(string)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Converts the CHAR, VARCHAR, VARCHAR2, or CLOB type into the TEXT type. If the value to be converted from the CLOB type exceeds the acceptable value range for the target type, an error will be reported.

Return type: text.

Example:

```
gaussdb=# SELECT to_char('01110');
 to_char
---------
 01110
(1 row)
```

- to_nvarchar2(numeric)

  Description: Converts the input into the nvarchar2 type.

  Return type: nvarchar2.

- to_char(timestamp, text)

  Description: Converts a timestamp value into a character string in the specified format.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT to_char(current_timestamp, 'HH12:MI:SS');
   to_char
  ----------
   10:55:59
  (1 row)
  ```

  NOTE

  - When **a_format_version** is **10c** and **a_format_dev_version** is **s1**, the to_char function will report an error for any incorrect **fmt**.
  - In non-compatible mode, the to_char function will output any unmatched digit in the given **fmt** value as it is. For example, if **fmt** is **FF10**, the function will convert based on the matched **FF1** and output the unmatched digit **0** as it is.

- to_nchar(datetime/interval [, fmt])

  Description: Converts a DATETIME or INTERVAL value of the DATE, DATEA (effective only in ORA-compatible mode), TIMESTAMP, TIMESTAMP WITH TIME ZONE, or TIMESTAMP WITH LOCAL TIME ZONE type into the TEXT type in the format specified by **fmt**.

  – The optional parameter **fmt** accepts the following types: date, time, week, quarter, and century. Each type comes with its own distinct template. These templates can be combined as needed. Commonly used templates include HH, MI, SS, YYYY, MM, and DD. For details, see **Table 1-69**.

  – A template can include a modifier, typically "FM", to help identify leading zeros or trailing spaces.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT to_nchar(current_timestamp,'HH12:MI:SS');
   to_nchar
  ----------
   10:19:26
  (1 row)
  gaussdb=# SELECT to_nchar(current_timestamp,'FMHH12:FMMI:FMSS');
   to_nchar
  ----------
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 10:19:46
(1 row)
```

NOTE

- The to_nchar function is supported when **a_format_version** is **10c** and **a_format_dev_version** is **s2**.
- In ORA-compatible databases where the **mapping_date_to_datea** parameter is enabled and **fmt** is not specified, the datea type will be converted into the text type based on **nls_date_format**.

- to_nchar(double precision/real, text)

  Description: Converts a floating-point value into a character string in the specified format.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_nchar(125.8::real, '999D99');
 to_nchar
----------
  125.80
(1 row)
```

- to_nchar(numeric/smallint/integer/bigint/double precision/real[, fmt])

  Description: Converts an integer or floating-point value into a character string in the specified format.

  – The optional parameter **fmt** accepts the following types: decimal character, digit grouping separator, plus/minus sign, and currency symbol. Each type comes with its own distinct template. These templates can be combined as needed. Commonly used templates include 9, 0, thousands separator (,), and decimal point (.). For details, see **Table 1-72**.

  – A template can include a modifier like "FM". Be aware that if a 0 is specified in the template, "FM" will not be able to identify the 0 in the output.

  – To convert an integer value into a hexadecimal string, you can use the template X or x.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_nchar(1485,'9,999');
 to_nchar
----------
  1,485
(1 row)
gaussdb=# SELECT to_nchar( 1148.5,'9,999.999');
  to_nchar
------------
  1,148.500
(1 row)
gaussdb=# SELECT to_nchar(148.5,'990999.909');
   to_nchar
-------------
    0148.500
(1 row)
gaussdb=# SELECT to_nchar(123,'XXX');
 to_nchar
----------
   7B
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> This function takes effect only in ORA-compatible mode (**sql_compatibility = 'ORA'**) when **a_format_version** is **10c** and **a_format_dev_version** is **s2**. When the parameter is enabled, the $, C, TM, TM9, TME, and U formats are supported. However, it does not support TH, PL, or SG.

- to_nchar(interval, text)

  Description: Converts a time interval value into a character string in the specified format.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_nchar(interval '15h 2m 12s', 'HH24:MI:SS');
 to_nchar
----------
 15:02:12
(1 row)
```

- to_nchar(integer, text)

  Description: Converts an integer value into a character string in the specified format.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_nchar(125, '999');
 to_nchar
----------
  125
(1 row)
```

- to_nchar(set)

  Description: Converts a value of the SET type into a character string. The SET type is currently unsupported in distributed mode.

  Return type: text.

- to_nchar(numeric, text)

  Description: Converts a numeric value into a character string in the specified format.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_nchar(-125.8, '999D99S');
 to_nchar
----------
 125.80-
(1 row)
```

- to_nchar(string)

  Description: Converts the CHAR, VARCHAR, VARCHAR2, or CLOB type into the TEXT type. If the value to be converted from the CLOB type exceeds the acceptable value range for the target type, an error will be reported.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_nchar('01110');
 to_nchar
----------
 01110
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- to_nchar(timestamp, text)

  Description: Converts a timestamp value into a character string in the specified format.

  Return type: text.

  Example:

```
gaussdb=# SELECT to_nchar(current_timestamp, 'HH12:MI:SS');
 to_nchar
----------
 10:55:59
(1 row)
```

  > NOTE
  >
  > The to_nchar function is supported when **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- to_clob(char/nchar/varchar/varchar2/nvarchar2/text/raw)

  Description: Converts the RAW type or a text character set type (CHAR, NCHAR, VARCHAR, VARCHAR2, NVARCHAR2, or TEXT) into the CLOB type.

  Return type: CLOB.

  Example:

```
gaussdb=# SELECT to_clob('ABCDEF'::RAW(10));
 to_clob
---------
 ABCDEF
(1 row)
gaussdb=# SELECT to_clob('hello111'::CHAR(15));
 to_clob
----------
 hello111
(1 row)
gaussdb=# SELECT to_clob('gauss123'::NCHAR(10));
 to_clob
----------
 gauss123
(1 row)
gaussdb=# SELECT to_clob('gauss234'::VARCHAR(10));
 to_clob
----------
 gauss234
(1 row)
gaussdb=# SELECT to_clob('gauss345'::VARCHAR2(10));
 to_clob
----------
 gauss345
(1 row)
gaussdb=# SELECT to_clob('gauss456'::NVARCHAR2(10));
 to_clob
----------
 gauss456
(1 row)
gaussdb=# SELECT to_clob('World222!'::TEXT);
  to_clob
-----------
 World222!
(1 row)
```

- to_date(text)

  Description: Converts a text value into a timestamp in the specified format. If **a_format_version** is set to **10c**, **a_format_dev_version** is set to **s1**, and **mapping_date_to_datea** is set to **off**, the conversion is performed based on the format specified by the **nls_timestamp_format** parameter. If

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

a_format_version and **a_format_dev_version** are not specified, the conversion is performed based on the fixed format **'YYYY-MM-DD HH24:MI:SS'**.

- Format 1: Date without separators, for example, 20150814. The value must contain the complete year, month, and day.
- Format 2: Date with separators, for example, 2014-08-14. The separator can be any non-digit character.

Return type: timestamp without time zone or datea (effective only in ORA-compatible mode).

Example:

```
-- In ORA-compatible mode, ensure that the input text type adheres to nls_date_format. Otherwise, an error will be reported.
gaussdb=# SELECT to_date('2015-08-14');
      to_date
---------------------
 2015-08-14 00:00:00
(1 row)
```

NOTE

- When **sql_compatibility** is **'ORA'** and the GUC parameter **mapping_date_to_datea** is **on**, this function returns a value of the datea type.
- The to_date function is of the stable rather than immutable type. Therefore, it cannot be used as generated columns/expression indexes.

- to_date(text, text)

  Description: Converts a character-string value into a date in the specified format.

  Return type: timestamp without time zone or datea (effective only in ORA-compatible mode).

  Example:

  ```
  gaussdb=# SELECT to_date('05 Dec 2000', 'DD Mon YYYY');
        to_date
  ---------------------
   2000-12-05 00:00:00
  (1 row)
  ```

  NOTE

  - When **sql_compatibility** is **'ORA'** and the GUC parameter **mapping_date_to_datea** is **on**, this function returns a value of the datea type.
  - The to_date function is of the stable rather than immutable type. Therefore, it cannot be used as generated columns/expression indexes.

- to_date(text [DEFAULT return_value ON CONVERSION ERROR [, fmt]])

  Description: Converts **text** into the DATE type in the format specified by **fmt**. If **fmt** is not specified, the conversion will adhere to **nls_timestamp_format** instead, provided that **a_format_version** is **10c** and **a_format_dev_version** is **s1**. However, if **a_format_version** and **a_format_dev_version** are not specified, the conversion will default to the fixed format **'YYYY-MM-DD HH24:MI:SS'**.

  Parameters:

  - **text**: any expression that can be calculated as CHAR, VARCHAR2, NCHAR, NVARCHAR2, or TEXT string. If the input is null, the function will return null.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **DEFAULT return_value ON CONVERSION ERROR**: This optional parameter specifies that **return_value** will be returned when **text** fails to be converted into the DATE type, provided that **a_format_version** is **10c** and **a_format_dev_version** is **s1**. **return_value** can be an expression or a bind variable that must be convertible into the CHAR, VARCHAR2, NCHAR, NVARCHAR2, or TEXT type. It can also be null. The method of converting **return_value** into the DATE type is the same as converting **text** into the DATE type. If **return_value** fails to be converted into the DATE type, the function will report an error.
- **fmt**: optional parameter. It specifies the date and time format for **string**. If it is not specified, **text** must adhere to the default date format. However, if **fmt** is set to **J**, **text** must be an integer.

Return type: timestamp without time zone or datea (effective only in ORA-compatible mode).

Example:

```
-- In ORA-compatible mode, if the input consists solely of a text type, ensure that the text type
adheres to nls_date_format. Otherwise, an error will be reported.
gaussdb=# SELECT to_date('2015-08-14');
      to_date
---------------------
 2015-08-14 00:00:00
(1 row)
gaussdb=# SELECT to_date('05 Dec 2000', 'DD Mon YYYY');
      to_date
---------------------
 2000-12-05 00:00:00
(1 row)
gaussdb=# SET a_format_version='10c';
SET
gaussdb=# SET a_format_dev_version='s1';
SET
gaussdb=# SHOW nls_timestamp_format;
   nls_timestamp_format
----------------------------
 DD-Mon-YYYY HH:MI:SS.FF AM
(1 row)
gaussdb=# SELECT to_date('12-jan-2022' DEFAULT '12-apr-2022' ON CONVERSION ERROR);
      to_date
---------------------
 2022-01-12 00:00:00
(1 row)
gaussdb=# SELECT to_date('12-ja-2022' DEFAULT '12-apr-2022' ON CONVERSION ERROR);
      to_date
---------------------
 2022-04-12 00:00:00
(1 row)
gaussdb=# SELECT to_date('2022-12-12' DEFAULT '2022-01-01' ON CONVERSION ERROR, 'yyyy-mm-
dd');
      to_date
---------------------
 2022-12-12 00:00:00
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When **a_format_version** is **10c** and **a_format_dev_version** is **s1**, inputting a calendar year above 9999 does not necessarily trigger an error. For example, executing to_date('99999-12-12', 'YYYY-MM-DD HH24:MI:SS') will yield 9999-09-12 12:00:00. Any numbers that come after 9999 (the maximum allowed year) will be interpreted as the next **fmt**. This limitation also extends to the to_timestamp function.
- When **sql_compatibility** is **'ORA'** and the GUC parameter **mapping_date_to_datea** is **on**, this function returns a value of the DATEA type. In this case, to_date converts **text** into the DATEA type in the format specified by **fmt**. If **fmt** is not specified, the conversion will adhere to **nls_date_format** instead.
- The to_date function is of the stable rather than immutable type. Therefore, it cannot be used as generated columns/expression indexes.

- to_number(expr [, fmt])

  Description: Converts **expr** into the NUMBER type in the specified format. For details about type conversion formats, see **Table 1-75**. During hexadecimal-to-decimal conversion, only hexadecimal strings with a maximum of 16 bytes can be converted into unsigned numbers. It is important to note that the format value must contain only x or X characters. Otherwise, an error will be reported.

  Return type: number.

  Example:

```
gaussdb=# SELECT to_number('12,454.8-', '99G999D9S');
 to_number
-----------
  -12454.8
(1 row)
```

- to_number(text, text)

  Description: Converts a character-string value into a number in the specified format.

  Return type: numeric.

  Example:

```
gaussdb=# SELECT to_number('12,454.8-', '99G999D9S');
 to_number
-----------
  -12454.8
(1 row)
```

- to_number(expr [DEFAULT return_value ON CONVERSION ERROR [, fmt]])

  Description: Converts **expr** into a numeric value in the format specified by **fmt**. If **fmt** is not specified, **expr** must be directly convertible into a numeric value, such as '123' and '1e2'. For details about type conversion formats, see **Table 1-76**.

  Parameters:

  - **expr**: an expression that can be converted into a CHAR, VARCHAR2, NCHAR, NVARCHAR2, TEXT, INT, or FLOAT string. If the input is null, the function will return null.
  - **DEFAULT return_value ON CONVERSION ERROR**: This optional parameter specifies that **return_value** will be returned when **expr** fails to be converted into the numeric type, provided that **a_format_version** is **10c** and **a_format_dev_version** is **s1**. Similar to **expr**, **return_value** accepts any type that can be converted into a character string. Similar to

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

expr, **return_value** is converted based on **fmt**. If the conversion fails, the function will report an error.

– **fmt**: optional parameter. Specifies the conversion format of **expr**. If any input parameter is null, the function will return null.

Return type: numeric.

Example:

```
gaussdb=# SET a_format_version='10c';
gaussdb=# SET a_format_dev_version='s1';

gaussdb=# SELECT to_number('1e2');
 to_number
-----------
       100
(1 row)

gaussdb=# SELECT to_number('123.456');
 to_number
-----------
   123.456
(1 row)

gaussdb=# SELECT to_number('123', '999');
 to_number
-----------
       123
(1 row)

gaussdb=# SELECT to_number('123-', '999MI');
 to_number
-----------
      -123
(1 row)

gaussdb=# SELECT to_number('123' DEFAULT '456-' ON CONVERSION ERROR, '999MI');
 to_number
-----------
      -456
(1 row)
```

- to_timestamp(double precision)

  Description: Converts a Unix century into a timestamp.

  Return type: timestamp with time zone.

  Example:

  ```
  gaussdb=# SELECT to_timestamp(1284352323);
      to_timestamp
  ------------------------
   2010-09-13 12:32:03+08
  (1 row)
  ```

- to_timestamp(string [,fmt])

  Description: Converts **string** into a timestamp value in the format specified by **fmt**. If **fmt** is not specified, the conversion will adhere to **nls_timestamp_format** instead. In **to_timestamp** in GaussDB:

  – If the input year **YYYY** is 0, the function will report an error.

  – If the input year **YYYY** is less than 0 and **fmt** is set to **SYYYY**, the function will correctly output the absolute value *n* of the BC year.

  Characters in **fmt** must match data/time formats. Otherwise, an error will be reported.

  Return type: timestamp without time zone.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SHOW nls_timestamp_format;
    nls_timestamp_format
-----------------------------
 DD-Mon-YYYY HH:MI:SS.FF AM
(1 row)

gaussdb=# SELECT to_timestamp('12-sep-2014');
    to_timestamp
---------------------
 2014-09-12 00:00:00
(1 row)
gaussdb=# SELECT to_timestamp('12-Sep-10 14:10:10.123000','DD-Mon-YY HH24:MI:SS.FF');
      to_timestamp
--------------------------
 2010-09-12 14:10:10.123
(1 row)
gaussdb=# SELECT to_timestamp('-1','SYYYY');
      to_timestamp
------------------------
 0001-01-01 00:00:00 BC
(1 row)
gaussdb=# SELECT to_timestamp('98','RR');
    to_timestamp
---------------------
 1998-01-01 00:00:00
(1 row)
gaussdb=# SELECT to_timestamp('01','RR');
    to_timestamp
---------------------
 2001-01-01 00:00:00
(1 row)
```

NOTE

- When **a_format_version** is **10c** and **a_format_dev_version** is **s1**, **fmt** supports FF[7-9]. When FF[7-9] is applied, the number of characters that can be converted from the target position in the string is limited by the number following FF, with only 6 characters ultimately being retained in the conversion result.
- The result returned by the current_timestamp function cannot be used as a string parameter.
- In PG-compatible mode, if a string exceeds the maximum length allowed by **fmt**, as seen in the example with to_timestamp('2024-01-05 14:52:06.694752', 'yyyy-MM-dd hh24:mi:ss'), the string will be automatically truncated to fit within the maximum length allowed by **fmt**, and any excess part will be discarded.
- In PG-compatible databases where the GUC parameter **format_pg_to_timestamp** is enabled, any unsupported formats will be skipped. The current version does not recognize formats such as FF, FF7, FF8, FF9, RR, RRRR, SYYYY, x, and X. However, it supports dates prior to 294277-01-01 00:00:00 and negative years (converted into BC). Milliseconds exceeding precision will be rounded off in FF1 to FF6. When conflicting time and date formats arise, the correctly formatted one will be returned without generating an error. For details about such conflicts, see **Table 1-73**.

Table 1-73 Conflicting formats (fmt) for to_timestamp

Conflict Scenario	Description
SSSSS conflicts with time.	Retain minutes and hours of time and the seconds part of SSSSS.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Conflict Scenario	Description
W and J conflict with time and date.	Priority: Overwrite J, W, and mmdd in sequence. W: Wednesday unless otherwise specified. Default behavior: Overwrite mmdd with W.
SSSSS conflicts with SS.	Retain SS and discard SSSSS.
DDD conflicts with a date.	Discard DDD.

- to_timestamp(text [DEFAULT return_value ON CONVERSION ERROR [, fmt]])

  Description: Converts **text** into the TIMESTAMP type in the format specified by **fmt**. If **fmt** is not specified, the conversion will adhere to **nls_timestamp_format** instead, provided that **a_format_version** is **10c** and **a_format_dev_version** is **s1**. However, if **a_format_version** and **a_format_dev_version** are not specified, the conversion will default to the fixed format '**yyyy-mm-dd hh24-mi-ss**'.

  Parameters:

  - **text**: any expression that can be calculated as CHAR, VARCHAR2, NCHAR, NVARCHAR2, or TEXT string. If the input is null, the function will return null.
  - **DEFAULT return_value ON CONVERSION ERROR**: This optional parameter specifies that **return_value** will be returned when **text** fails to be converted into the TIMESTAMP type, provided that **a_format_version** is **10c** and **a_format_dev_version** is **s1**. **return_value** can be an expression or a bind variable that must be convertible into the CHAR, VARCHAR2, NCHAR, NVARCHAR2, or TEXT type. It can also be null. The method of converting **return_value** into the TIMESTAMP type is the same as converting **text** into the TIMESTAMP type. If **return_value** fails to be converted into the TIMESTAMP type, the function will report an error.
  - **fmt**: optional parameter. It specifies the date and time format for **string**. If it is not specified, **text** must adhere to the default date format. However, if **fmt** is set to **J**, **text** must be an integer.

  Return type: timestamp without time zone.

  Example:

```
gaussdb=# SET a_format_version='10c';
SET
gaussdb=# SET a_format_dev_version='s1';
SET
gaussdb=# SELECT to_timestamp('11-Sep-11' DEFAULT '12-Sep-10 14:10:10.123000' ON
CONVERSION ERROR,'DD-Mon-YY HH24:MI:SS.FF');
    to_timestamp
---------------------
 2011-09-11 00:00:00
(1 row)
gaussdb=# SELECT to_timestamp('12-Sep-10 14:10:10.123000','DD-Mon-YY HH24:MI:SSXFF');
      to_timestamp
-------------------------
 2010-09-12 14:10:10.123
(1 row)
```

- to_timestamp(text, text)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Converts a character-string value into a timestamp in the specified format.

Return type: timestamp.

Example:

```
gaussdb=# SELECT to_timestamp('05 Dec 2000', 'DD Mon YYYY');
    to_timestamp
---------------------
 2000-12-05 00:00:00
(1 row)
```

- to_timestamp_tz(string [DEFAULT return_value ON CONVERSION ERROR] [,fmt])

  Description: Converts **string** into the timestamp with time zone type in the format specified by **fmt**. If **fmt** is not specified, the conversion will adhere to **nls_timestamp_tz_format** instead.

  Parameters:

  – **DEFAULT return_value ON CONVERSION ERROR**: optional parameter. If **string** fails to be converted into the timestamp with time zone type, **return_value** will be converted into that type instead.

  – **fmt**: optional parameter. It specifies the date and time format for **string**.

  Return type: timestamp with time zone.

  Example:

```
gaussdb=# SELECT to_timestamp_tz('05 DeX 2000' DEFAULT '05 Dec 2001' ON CONVERSION ERROR,
'DD Mon YYYY');
    to_timestamp_tz
---------------------------
 2001-12-05 00:00:00+08:00
(1 row)
```

  NOTE

  This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s1**.

- to_timestamp_tz(string [DEFAULT return_value ON CONVERSION ERROR], fmt, nlsparam)

  Description: Converts **string** into the timestamp with time zone type in the format specified by **fmt**. If **string** fails to be converted into the timestamp with time zone type, **return_value** will be converted into that type instead.

  Parameters: See **Table 1-74**.

  Return type: timestamp with time zone.

  Table 1-74 Parameters of to_timestamp_tz(string [DEFAULT return_value ON CONVERSION ERROR], fmt, nlsparam)

Parameter	Type	Description
string	text	Character string to be converted into the timestamp with time zone type.
return_value	text	If **string** fails to be converted into the timestamp with time zone type, **return_value** will be converted into that type instead.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
fmt	text	Date and time format for the **string** parameter.
nlsparam	text	Language for month and day names in the specified **string** in the format of '**nls_date_language=***language*'. Currently, *language* can only be **ENGLISH** or **AMERICAN**. Using the **nlsparam** parameter correctly yields the same result as omitting it.

Example:

```
gaussdb=# SELECT to_timestamp_tz('05 DeX 2000' DEFAULT '05 Dec 2001' ON CONVERSION ERROR,
'DD Mon YYYY','nls_date_language=AMERICAN');
    to_timestamp_tz
---------------------------
 2001-12-05 00:00:00+08:00
(1 row)
```

NOTE

This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s4**.

- to_dsinterval(text)

  Description: Converts characters into the interval type. SQL-compatible and ISO formats are supported.

  Return type: interval.

  Example:

  ```
  gaussdb=# SELECT to_dsinterval('12 1:2:3.456');
     to_dsinterval
  ----------------------
   12 days 01:02:03.456
  (1 row)

  gaussdb=# SELECT to_dsinterval('P3DT4H5M6S');
    to_dsinterval
  -----------------
   3 days 04:05:06
  (1 row)
  ```

  NOTE

  This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

- to_yminterval(text)

  Description: Converts characters into the interval type. SQL-compatible and ISO formats are supported.

  Return type: interval.

  Example:

  ```
  gaussdb=# SELECT to_yminterval('1-1');
   to_yminterval
  -----------------
   1 year 1 mon
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT to_yminterval('P13Y3M4DT4H2M5S');
 to_yminterval
-----------------
 13 years 3 mons
(1 row)
```

NOTE

This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2**.

Table 1-75 Numeric formats

Format	Description
9	Returns a value with the specified number of digits.
0	Returns leading zeros.
. (period)	Returns the decimal point.
, (comma)	Returns the digit grouping (thousands) separator.
PR	Returns a negative value in <angle brackets>.
S	Returns a signed number (depending on the locale settings).
L	Returns the currency symbol (depending on the locale settings).
D	Returns the decimal character (depending on the locale settings).
G	Returns the digit grouping separator (depending on the locale settings).
MI	Returns the minus sign in the specified position (if the number is less than 0).
PL	Returns the plus sign in the specified position (if the number is greater than 0).
SG	Returns the plus/minus sign in the specified position.
RN	Returns a Roman numeral (for any input between 1 and 3,999).
TH / th	Returns the suffix for an ordinal number.
V	Shifts the specified number of digits (decimal).
x / X	Converts a decimal number into its hexadecimal equivalent.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-76 Numeric formats for to_number

Format	Description
9	Matches a digit. The digit "9" can occur an equal or greater number of times than the digits in the specified position in **expr**.
0	Strictly matches a digit. The digit "0" must occur the same number of times as the digits in **expr**.
5	Matches the digit "0" or "5".
. (period)	Returns a decimal point in the specified position.
, (comma)	Returns a digit grouping (thousands) separator in the specified position. You can specify multiple commas in **fmt**.
B	Returns leading blanks.
PR	Specifies a negative value in angle brackets (<>), or a positive value with no leading sign.
S	Specifies a negative value with the leading minus sign (–) or a positive value with the leading plus sign (+).
MI	Specifies a negative value with the leading minus sign (–) or a positive value with no leading sign.
$	Returns a value with a leading US dollar sign ($).
L	Returns the local currency symbol.
C	Returns the ISO currency symbol in the specified position.
U	Returns the dual currency symbol.
D	Returns the decimal character (depending on the locale settings).
G	Returns the digit grouping separator (complying with the ISO standard). You can specify multiple commas in **fmt**.
RN / rn	Returns a Roman numeral (for any input between 1 and 3,999). The to_number function does not support this format.
V	The to_number function does not support this format.
X / x	Converts a hexadecimal number into its decimal equivalent.
TM	The to_number function does not support this format.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Format	Description
FM	Removes any leading blanks from the beginning of **fmt** while retaining the placeholder 0. This does not alter the result of the to_number function.
EEEE	Returns a value in scientific notation.

NOTE

Table 1-76 describes the usage of **fmt** in cases where **a_format_version** is **10c** and **a_format_dev_version** is **s1**, while Table 1-75 outlines the usage of **fmt** in all other cases. In Table 1-76, the usage of ISO-related formats is influenced by the **lc_monetary** and **lc_numeric** parameters. Specifically, **lc_monetary** determines the currency symbol, and **lc_numeric** affects the display rules for the decimal point and thousands separator.

- cast_varchar2_to_raw_for_histogram(varchar2)
  Description: Converts the varchar2 type into the raw type.
  Return type: raw.
- abstime_text(abstime)
  Description: Converts the abstime type into the text type.
  Return type: text.
- abstime_to_smalldatetime(abstime)
  Description: Converts the abstime type into the smalldatetime type.
  Return type: smalldatetime.
- bigint_tid(bigint)
  Description: Converts the bigint type into the tid type.
  Return type: tid.
- bool_int1(boolean)
  Description: Converts the Boolean type into the int1 type.
  Return type: tinyint.
- bool_int2(boolean)
  Description: Converts the Boolean type into the int2 type.
  Return type: smallint.
- bool_int8(boolean)
  Description: Converts the Boolean type into the int8 type.
  Return type: bigint.
- bpchar_date(character)
  Description: Converts a character string into a date.
  Return type: date.
- bpchar_float4(character)
  Description: Converts a character string into the float4 type.
  Return type: real.
- bpchar_float8(character)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Converts a character string into the float8 type.

Return type: double precision.

- bpchar_int4(character)

  Description: Converts a character string into the int4 type.

  Return type: integer.

- bpchar_int8(character)

  Description: Converts a character string into the int8 type.

  Return type: bigint.

- bpchar_numeric(character)

  Description: Converts a character string into the numeric type.

  Return type: numeric.

- bpchar_timestamp(character)

  Description: Converts a character string into a timestamp.

  Return type: timestamp without time zone.

- bpchar_to_smalldatetime(character)

  Description: Converts a character string into the smalldatetime type.

  Return type: smalldatetime.

- complex_array_in(cstring, oid, int2vector)

  Description: Converts an external complex_array type into the internal anyarray type.

  Return type: anyarray.

- date_bpchar(date)

  Description: Converts the date type into the bpchar type.

  Return type: character.

- date_text(date)

  Description: Converts the date type into the text type.

  Return type: text.

- date_varchar(date)

  Description: Converts the date type into the varchar type.

  Return type: character varying.

- f4toi1(real)

  Description: Forcibly converts the float4 type into the uint8 type.

  Return type: tinyint.

- f8toi1(double precision)

  Description: Forcibly converts the float8 type into the uint8 type.

  Return type: tinyint.

- float4_bpchar(real)

  Description: Converts the float4 type into the bpchar type.

  Return type: character.

- float4_text(real)

  Description: Converts the float4 type into the text type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: text.

- float4_varchar(real)
  Description: Converts the float4 type into the varchar type.
  Return type: character varying.
- float8_bpchar(double precision)
  Description: Converts the float8 type into the bpchar type.
  Return type: character.
- float8_interval(double precision)
  Description: Converts the float8 type into the interval type.
  Return type: interval.
- float8_text(double precision)
  Description: Converts the float8 type into the text type.
  Return type: text.
- float8_varchar(double precision)
  Description: Converts the float8 type into the varchar type.
  Return type: character varying.
- i1tof4(tinyint)
  Description: Converts the uint8 type into the float4 type.
  Return type: real.
- i1tof8(tinyint)
  Description: Converts the uint8 type into the float8 type.
  Return type: double precision.
- i1toi2(tinyint)
  Description: Converts the uint8 type into the int16 type.
  Return type: smallint.
- i1toi4(tinyint)
  Description: Converts the uint8 type into the int32 type.
  Parameter: tinyint.
  Return type: integer.
- i1toi8(tinyint)
  Description: Converts the uint8 type into the int64 type.
  Return type: bigint.
- i2toi1(smallint)
  Description: Converts the int16 type into the uint8 type.
  Return type: tinyint.
- i4toi1(integer)
  Description: Converts the int32 type into the uint8 type.
  Parameter: integer.
  Return type: tinyint.
- i8toi1(bigint)
  Description: Converts the int64 type into the uint8 type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: tinyint.

- int1_avg_accum(bigint[], tinyint)

  Description: Adds the second parameter of the uint8 type to the first parameter. The first parameter is an array of the bigint type.

  Return type: bigint[].

- int1_bool(tinyint)

  Description: Converts the uint8 type into the Boolean type.

  Return type: Boolean.

- int1_bpchar(tinyint)

  Description: Converts the uint8 type into the bpchar type.

  Return type: character.

- int1_mul_cash(tinyint, money)

  Description: Returns the product of an int8 parameter and a money parameter. The return type is money.

  Return type: money.

- int1_numeric(tinyint)

  Description: Converts the uint8 type into the numeric type.

  Return type: numeric.

- int1_nvarchar2(tinyint)

  Description: Converts the uint8 type into the nvarchar2 type.

  Return type: nvarchar2.

- int1_text(tinyint)

  Description: Converts the uint8 type into the text type.

  Return type: text.

- int1_varchar(tinyint)

  Description: Converts the uint8 type into the varchar type.

  Return type: character varying.

- int1in(cstring)

  Description: Converts a character string into an unsigned 1-byte integer.

  Return type: tinyint.

- int1out(tinyint)

  Description: Converts an unsigned 1-byte integer into a character string.

  Return type: cstring.

- int1up(tinyint)

  Description: Converts an integer into an unsigned 1-byte integer.

  Return type: tinyint.

- int2_bool(smallint)

  Description: Converts a signed two-byte integer into the Boolean type.

  Return type: Boolean.

- int2_bpchar(smallint)

  Description: Converts a signed two-byte integer into the bpchar type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: character.

- int2_text(smallint)
  Description: Converts a signed two-byte integer into the text type.
  Return type: text.
- int2_varchar(smallint)
  Description: Converts a signed two-byte integer into the varchar type.
  Return type: character varying.
- int4_bpchar(integer)
  Description: Converts a signed four-byte integer into the bpchar type.
  Return type: character.
- int4_text(integer)
  Description: Converts a signed four-byte integer into the text type.
  Return type: text.
- int4_varchar(integer)
  Description: Converts a signed four-byte integer into the varchar type.
  Return type: character varying.
- int8_bool(bigint)
  Description: Converts a signed eight-byte integer into the Boolean type.
  Return type: Boolean.
- int8_bpchar(bigint)
  Description: Converts a signed eight-byte integer into the bpchar type.
  Return type: character.
- int8_text(bigint)
  Description: Converts a signed eight-byte integer into the text type.
  Return type: text.
- int8_varchar(bigint)
  Description: Converts a signed eight-byte integer into the varchar type.
  Return type: character varying.
- intervaltonum(interval)
  Description: Converts an internal numeric date into the numeric type.
  Return type: numeric.
- numeric_bpchar(numeric)
  Description: Converts the numeric type into the bpchar type.
  Return type: character.
- numeric_int1(numeric)
  Description: Converts the numeric type into the signed 1-byte integer type.
  Return type: tinyint.
- numeric_text(numeric)
  Description: Converts the numeric type into the text type.
  Return type: text.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- numeric_varchar(numeric)
  Description: Converts the numeric type into the varchar type.
  Return type: character varying.
- nvarchar2in(cstring, oid, integer)
  Description: Converts a cstring into the varchar type.
  Return type: nvarchar2.
- nvarchar2out(nvarchar2)
  Description: Converts the text type into a cstring.
  Return type: cstring.
- nvarchar2send(nvarchar2)
  Description: Converts the varchar type into the binary format.
  Return type: bytea.
- oidvectorin_extend(cstring)
  Description: Converts a character string into the oidvector type.
  Return type: oidvector_extend.
- oidvectorout_extend(oidvector_extend)
  Description: Converts the oidvector type into a character string.
  Return type: cstring.
- oidvectorsend_extend(oidvector_extend)
  Description: Converts the oidvector type into a character string.
  Return type: bytea.
- reltime_text(reltime)
  Description: Converts the reltime type into the text type.
  Return type: text.
- text_date(text)
  Description: Converts the text type into the date type.
  Return type: date.
- text_float4(text)
  Description: Converts the text type into the float4 type.
  Return type: real.
- text_float8(text)
  Description: Converts the text type into the float8 type.
  Return type: double precision.
- text_int1(text)
  Description: Converts the text type into the int1 type.
  Return type: tinyint.
- text_int2(text)
  Description: Converts the text type into the int2 type.
  Return type: smallint.
- text_int4(text)
  Description: Converts the text type into the int4 type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: integer.

- text_int8(text)
  Description: Converts the text type into the int8 type.
  Return type: bigint.
- text_numeric(text)
  Description: Converts the text type into the numeric type.
  Return type: numeric.
- text_timestamp(text)
  Description: Converts the text type into the timestamp type.
  Return type: timestamp without time zone.
- time_text(time without time zone)
  Description: Converts the time type into the text type.
  Return type: text.
- timestamp_text(timestamp without time zone)
  Description: Converts the timestamp type into the text type.
  Return type: text.
- timestamp_to_smalldatetime(timestamp without time zone)
  Description: Converts the timestamp type into the smalldatetime type.
  Return type: smalldatetime.
- timestamp_varchar(timestamp without time zone)
  Description: Converts the timestamp type into the varchar type.
  Return type: character varying.
- timestamptz_to_smalldatetime(timestamp with time zone)
  Description: Converts the timestamptz type into the smalldatetime type.
  Return type: smalldatetime.
- timestampzone_text(timestamp with time zone)
  Description: Converts the timestamptz type into the text type.
  Return type: text.
- timetz_text(time with time zone)
  Description: Converts the timetz type into the text type.
  Return type: text.
- to_integer(character varying)
  Description: Converts the input into the integer type.
  Return type: integer.
- to_interval(character varying)
  Description: Converts the input into the interval type.
  Return type: interval.
- to_numeric(character varying)
  Description: Converts the input into the numeric type.
  Return type: numeric.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- to_text(smallint)
  Description: Converts the input into the text type.
  Return type: text.
- to_ts(character varying)
  Description: Converts the input into the timestamptz type.
  Return type: timestamp without time zone.
- to_varchar2(timestamp without time zone)
  Description: Converts the input into the varchar2 type.
  Return type: character varying.
- varchar_blob(character varying)
  Description: Converts the varchar type into the blob type.
  Return type: blob.

  NOTICE

  This function is supported only in Oracle-compatible mode. Using it in other modes will result in an error.

- varchar_date(character varying)
  Description: Converts the varchar type into the date type.
  Return type: date.
- varchar_float4(character varying)
  Description: Converts the varchar type into the float4 type.
  Return type: real.
- varchar_float8(character varying)
  Description: Converts the varchar type into the float8 type.
  Return type: double precision.
- varchar_int4(character varying)
  Description: Converts the varchar type into the int4 type.
  Return type: integer.
- varchar_int8(character varying)
  Description: Converts the varchar type into the int8 type.
  Return type: bigint.
- varchar_numeric(character varying)
  Description: Converts the varchar type into the numeric type.
  Return type: numeric.
- varchar_timestamp(character varying)
  Description: Converts the varchar type into the timestamp type.
  Return type: timestamp without time zone.
- varchar2_to_smlldatetime(character varying)
  Description: Converts the varchar2 type into the smlldatetime type.
  Return type: smalldatetime.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- xidout4(xid32)
  Description: Converts the xid type into a 4-byte number.
  Return type: cstring.
- xidsend4(xid32)
  Description: Converts the xid type into the binary format.
  Return type: bytea.
- bytea_blob(any)
  Description: Converts the bytea type into the blob type. Currently, this conversion is limited to the bytea type.
  Return type: blob.
- datea_text(datea)
  Description: Converts the datea type into the text type.
  Return type: text.
- datea_clob(datea)
  Description: Converts the datea type into the clob type.
  Return type: CLOB.
- datea_varchar(datea)
  Description: Converts the datea type into the character varying type.
  Return type: character varying.
- datea_nvarchar2(datea)
  Description: Converts the datea type into the nvarchar2 type.
  Return type: nvarchar2.
- datea_bpchar(datea)
  Description: Converts the datea type into the bpchar type.
  Return type: character.
- text_datea(text)
  Description: Converts the text type into the datea type.
  Return type: datea.
- clob_datea(clob)
  Description: Converts the clob type into the datea type.
  Return type: datea.
- varchar_datea(character varying)
  Description: Converts the character varying type into the datea type.
  Return type: datea.
- bpchar_datea(character)
  Description: Converts the character type into the datea type.
  Return type: datea.
- datea_timestamp(datea)
  Description: Converts the datea type into the timestamp type.
  Return type: timestamp.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- timestamp_datea(timestamp)

  Description: Converts the timestamp type into the datea type.

  Return type: datea.
- datea_timestamptz(datea)

  Description: Converts the datea type into the timestamptz type.

  Return type: timestamptz.
- timestamptz_datea(timestamptz)

  Description: Converts the timestamptz type into the datea type.

  Return type: datea.
- nvarchar2_datea(nvarchar2)

  Description: Converts the nvarchar2 type into the datea type.

  Return type: datea.

  NOTE

  The datea-related conversion functions can be used only in ORA-compatible mode.
- chartorowid(character/character varying/nvarchar2/text)

  Description: Converts the character/character varying/nvarchar2/text type into the rowid type.

  Return type: rowid.
- rowidtobpchar(rowid)

  Description: Converts the rowid type into the character type.

  Return type: character.
- rowidtochar(rowid)

  Description: Converts the rowid type into the character varying type.

  Return type: character varying.
- rowidtonchar(rowid)

  Description: Converts the rowid type into the nvarchar2 type.

  Return type: nvarchar2.
- rowidtotext(rowid)

  Description: Converts the rowid type into the text type.

  Return type: text.

  NOTE

  The rowid-related conversion functions can be used only in Oracle-compatible mode.
- treat(expr AS [JSON | REF] schema.type)

  Description: Converts **expr** into the type (JSON or user-defined type) specified by the keyword after **AS**.

  Return type: JSON or user-defined type.

  Example:

```
gaussdb=# CREATE TABLE json_doc(data CLOB);
gaussdb=# INSERT INTO json_doc VALUES('{"name":"a"}');
gaussdb=# SELECT treat(data as json) FROM json_doc;
    json
--------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 {"name":"a"}
(1 row)
gaussdb=# DROP TABLE json_doc;
DROP TABLE
```

- nesttable_to_array(anynesttable)

  Description: Converts a nest-table collection type into an array type with the same elements.

  Return type: anyarray.

  Example:

```
gaussdb=# CREATE OR REPLACE PROCEDURE p1 IS
gaussdb$#     type t1 IS table of int;
gaussdb$#     v1 t1 := t1(1, 2, 3);
gaussdb$#     v2 int[] := cast(v1 as int[]);
gaussdb$# BEGIN
gaussdb$#     raise info '%', v2;
gaussdb$# END;
gaussdb$# /
CREATE PROCEDURE
gaussdb=# CALL p1();
INFO:  {1,2,3}
 p1
----

(1 row)

gaussdb=# CREATE TYPE t1 is table of int;
CREATE TYPE
gaussdb=# SELECT cast(t1(1, 2, 3) AS int[]) result;
 result
---------
 {1,2,3}
(1 row)
gaussdb=# DROP PROCEDURE p1;
DROP PROCEDURE

gaussdb=# DROP TYPE t1;
DROP TYPE
```

- indexbytableint_to_array(anyindexbytable)

  Description: Converts a collection whose index type is integer into an array type with the same elements.

  Return type: anyarray.

  Example:

```
gaussdb=# CREATE OR REPLACE PACKAGE pkg1 IS
gaussdb$#     type t1 IS table of int index by int;
gaussdb$#     PROCEDURE p1();
gaussdb$# END pkg1;
gaussdb$# /
CREATE PACKAGE
gaussdb=#
gaussdb=# CREATE OR REPLACE PACKAGE BODY pkg1 IS
gaussdb$#     PROCEDURE p1() IS
gaussdb$#         v1 t1 := t1(1 => 1, 2 => 2, 3 => 3);
gaussdb$#         v2 int[];
gaussdb$#     BEGIN
gaussdb$#         v2 := cast(v1 as int[]);
gaussdb$#         RAISE INFO '%', v2;
gaussdb$#     END;
gaussdb$# END pkg1;
gaussdb$# /
CREATE PACKAGE BODY
gaussdb=#
gaussdb=# CALL pkg1.p1();
INFO:  {1,2,3}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 p1
----

(1 row)

gaussdb=# SELECT indexbytableint_to_array(pkg1.t1(1 => 1, 2 => 2, 3 => 3));
 indexbytableint_to_array
--------------------------
 {1,2,3}
(1 row)

gaussdb=# DROP PACKAGE pkg1;
DROP PACKAGE
```

Encoding Type Conversion

convert_to_nocase(text, text)

Description: Converts a string into a specified encoding type and then outputs the result.

Return type: bytea.

Example:

```
gaussdb=# SELECT convert_to_nocase('12345', 'GBK');
 convert_to_nocase
-------------------
 \x3132333435
(1 row)
```

1.6.10 Geometric Functions and Operators

Geometric Operators

- +

  Description: Translation

  Example:

  ```
  gaussdb=# SELECT box '((0,0),(1,1))' + point '(2.0,0)' AS RESULT;
     result
  -------------
   (3,1),(2,0)
  (1 row)
  ```

- -

  Description: Translation

  Example:

  ```
  gaussdb=# SELECT box '((0,0),(1,1))' - point '(2.0,0)' AS RESULT;
     result
  ---------------
   (-1,1),(-2,0)
  (1 row)
  ```

- *

  Description: Scaling out/Rotation

  Example:

  ```
  gaussdb=# SELECT box '((0,0),(1,1))' * point '(2.0,0)' AS RESULT;
     result
  -------------
   (2,2),(0,0)
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- /

  Description: Scaling in/Rotation

  Example:

```
gaussdb=# SELECT box '((0,0),(2,2))' / point '(2.0,0)' AS RESULT;
   result
-------------
 (1,1),(0,0)
(1 row)
```

- #

  Description: Intersection of two figures

  Example:

```
gaussdb=# SELECT box '((1,-1),(-1,1))' # box '((1,1),(-2,-2))' AS RESULT;
 result
---------------
 (1,1),(-1,-1)
(1 row)
```

- #

  Description: Number of paths or polygon vertexes

  Example:

```
gaussdb=# SELECT # path'((1,0),(0,1),(-1,0))' AS RESULT;
 result
--------
      3
(1 row)
```

- @-@

  Description: Length or circumference

  Example:

```
gaussdb=# SELECT @-@ path '((0,0),(1,0))' AS RESULT;
 result
--------
      2
(1 row)
```

- @@

  Description: Center of box

  Example:

```
gaussdb=# SELECT @@ circle '((0,0),10)' AS RESULT;
 result
--------
 (0,0)
(1 row)
```

- <->

  Description: Distance between the two figures

  Example:

```
gaussdb=# SELECT circle '((0,0),1)' <-> circle '((5,0),1)' AS RESULT;
 result
--------
      3
(1 row)
```

- &&

  Description: Overlaps? (One point in common makes this true.)

  Example:

```
gaussdb=# SELECT box '((0,0),(1,1))' && box '((0,0),(2,2))' AS RESULT;
 result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------
 t
(1 row)
```

- <<

  Description: Is strictly left of (no common horizontal coordinate)?

  Example:

```
gaussdb=# SELECT circle '((0,0),1)' << circle '((5,0),1)' AS RESULT;
 result
--------
 t
(1 row)
```

- >>

  Description: Is strictly right of (no common horizontal coordinate)?

  Example:

```
gaussdb=# SELECT circle '((5,0),1)' >> circle '((0,0),1)' AS RESULT;
 result
--------
 t
(1 row)
```

- &<

  Description: Does not extend to the right of?

  Example:

```
gaussdb=# SELECT box '((0,0),(1,1))' &< box '((0,0),(2,2))' AS RESULT;
 result
--------
 t
(1 row)
```

- &>

  Description: Does not extend to the left of?

  Example:

```
gaussdb=# SELECT box '((0,0),(3,3))' &> box '((0,0),(2,2))' AS RESULT;
 result
--------
 t
(1 row)
```

- <<|

  Description: Is strictly below (no common horizontal coordinate)?

  Example:

```
gaussdb=# SELECT box '((0,0),(3,3))' <<| box '((3,4),(5,5))' AS RESULT;
 result
--------
 t
(1 row)
```

- |>>

  Description: Is strictly above (no common horizontal coordinate)?

  Example:

```
gaussdb=# SELECT box '((3,4),(5,5))' |>> box '((0,0),(3,3))' AS RESULT;
 result
--------
 t
(1 row)
```

- &<|

  Description: Does not extend above?

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT box '((0,0),(1,1))' &<| box '((0,0),(2,2))' AS RESULT;
 result
--------
 t
(1 row)
```

- |&>

  Description: Does not extend below?

  Example:

```
gaussdb=# SELECT box '((0,0),(3,3))' |&> box '((0,0),(2,2))' AS RESULT;
 result
--------
 t
(1 row)
```

- <^

  Description: Is below (allows touching)?

  Example:

```
gaussdb=# SELECT box '((0,0),(-3,-3))' <^ box '((0,0),(2,2))' AS RESULT;
 result
--------
 t
(1 row)
```

- >^

  Description: Is above (allows touching)?

  Example:

```
gaussdb=# SELECT box '((0,0),(2,2))' >^ box '((0,0),(-3,-3))'  AS RESULT;
 result
--------
 t
(1 row)
```

- ?#

  Description: Intersect?

  Example:

```
gaussdb=# SELECT lseg '((-1,0),(1,0))' ?# box '((-2,-2),(2,2))' AS RESULT;
 result
--------
 t
(1 row)
```

- ?-

  Description: Is horizontal?

  Example:

```
gaussdb=# SELECT ?- lseg '((-1,0),(1,0))' AS RESULT;
 result
--------
 t
(1 row)
```

- ?-

  Description: Are horizontally aligned?

  Example:

```
gaussdb=# SELECT point '(1,0)' ?- point '(0,0)' AS RESULT;
 result
--------
 t
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- ?|

  Description: Is vertical?

  Example:

```
gaussdb=# SELECT ?| lseg '((-1,0),(1,0))' AS RESULT;
 result
--------
 f
(1 row)
```

- ?|

  Description: Are vertically aligned?

  Example:

```
gaussdb=# SELECT point '(0,1)' ?| point '(0,0)' AS RESULT;
 result
--------
 t
(1 row)
```

- ?-|

  Description: Are perpendicular?

  Example:

```
gaussdb=# SELECT lseg '((0,0),(0,1))' ?-| lseg '((0,0),(1,0))' AS RESULT;
 result
--------
 t
(1 row)
```

- ?||

  Description: Are parallel?

  Example:

```
gaussdb=# SELECT lseg '((-1,0),(1,0))' ?|| lseg '((-1,2),(1,2))' AS RESULT;
 result
--------
 t
(1 row)
```

- @>

  Description: Contains?

  Example:

```
gaussdb=# SELECT circle '((0,0),2)' @> point '(1,1)' AS RESULT;
 result
--------
 t
(1 row)
```

- <@

  Description: Contained in or on?

  Example:

```
gaussdb=# SELECT point '(1,1)' <@ circle '((0,0),2)' AS RESULT;
 result
--------
 t
(1 row)
```

- ~=

  Description: Specifies whether two figures are the same.

  Example:

```
gaussdb=# SELECT polygon '((0,0),(1,1))' ~= polygon '((1,1),(0,0))' AS RESULT;
 result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------
 t
(1 row)
```

Geometric Functions

- area(object)

  Description: Area calculation

  Return type: double precision

  Example:

  ```
  gaussdb=# SELECT area(box '((0,0),(1,1))') AS RESULT;
   result
  --------
       1
  (1 row)
  ```

- center(object)

  Description: Figure center calculation

  Return type: point

  Example:

  ```
  gaussdb=# SELECT center(box '((0,0),(1,2))') AS RESULT;
   result
  ---------
   (0.5,1)
  (1 row)
  ```

- diameter(circle)

  Description: Circle diameter calculation

  Return type: double precision

  Example:

  ```
  gaussdb=# SELECT diameter(circle '((0,0),2.0)') AS RESULT;
   result
  --------
       4
  (1 row)
  ```

- height(box)

  Description: Vertical size of box

  Return type: double precision

  Example:

  ```
  gaussdb=# SELECT height(box '((0,0),(1,1))') AS RESULT;
   result
  --------
       1
  (1 row)
  ```

- isclosed(path)

  Description: A closed path?

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT isclosed(path '((0,0),(1,1),(2,0))') AS RESULT;
   result
  --------
   t
  (1 row)
  ```

- isopen(path)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: An open path?

Return type: Boolean

Example:

```
gaussdb=# SELECT isopen(path '[(0,0),(1,1),(2,0)]') AS RESULT;
 result
--------
 t
(1 row)
```

- length(object)

  Description: Length calculation

  Return type: double precision

  Example:

```
gaussdb=# SELECT length(path '((-1,0),(1,0))') AS RESULT;
 result
--------
      4
(1 row)
```

- npoints(path)

  Description: Number of points in path

  Return type: int

  Example:

```
gaussdb=# SELECT npoints(path '[(0,0),(1,1),(2,0)]') AS RESULT;
 result
--------
      3
(1 row)
```

- npoints(polygon)

  Description: Number of points in polygon

  Return type: int

  Example:

```
gaussdb=# SELECT npoints(polygon '((1,1),(0,0))') AS RESULT;
 result
--------
      2
(1 row)
```

- pclose(path)

  Description: Converts path to closed.

  Return type: path

  Example:

```
gaussdb=# SELECT pclose(path '[(0,0),(1,1),(2,0)]') AS RESULT;
       result
---------------------
 ((0,0),(1,1),(2,0))
(1 row)
```

- popen(path)

  Description: Converts path to open.

  Return type: path

  Example:

```
gaussdb=# SELECT popen(path '((0,0),(1,1),(2,0))') AS RESULT;
       result
---------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 [(0,0),(1,1),(2,0)]
(1 row)
```

- radius(circle)

  Description: Circle diameter calculation

  Return type: double precision

  Example:

```
gaussdb=# SELECT radius(circle '((0,0),2.0)') AS RESULT;
 result
--------
      2
(1 row)
```

- width(box)

  Description: Horizontal size of box

  Return type: double precision

  Example:

```
gaussdb=# SELECT width(box '((0,0),(1,1))') AS RESULT;
 result
--------
      1
(1 row)
```

Geometric Type Conversion Functions

- box(circle)

  Description: Circle to box

  Return type: box

  Example:

```
gaussdb=# SELECT box(circle '((0,0),2.0)') AS RESULT;
                                result
-----------------------------------------------------------------------
 (1.41421356237309,1.41421356237309),(-1.41421356237309,-1.41421356237309)
(1 row)
```

- box(point, point)

  Description: Points to box

  Return type: box

  Example:

```
gaussdb=# SELECT box(point '(0,0)', point '(1,1)') AS RESULT;
   result
-------------
 (1,1),(0,0)
(1 row)
```

- box(polygon)

  Description: Polygon to box

  Return type: box

  Example:

```
gaussdb=# SELECT box(polygon '((0,0),(1,1),(2,0))') AS RESULT;
   result
-------------
 (2,1),(0,0)
(1 row)
```

- circle(box)

  Description: Box to circle

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: circle

Example:

```
gaussdb=# SELECT circle(box '((0,0),(1,1))') AS RESULT;
            result
-------------------------------
 <(0.5,0.5),0.707106781186548>
(1 row)
```

- circle(point, double precision)

  Description: Center and radius to circle

  Return type: circle

  Example:

  ```
  gaussdb=# SELECT circle(point '(0,0)', 2.0) AS RESULT;
    result
  -----------
   <(0,0),2>
  (1 row)
  ```

- circle(polygon)

  Description: Polygon to circle

  Return type: circle

  Example:

  ```
  gaussdb=# SELECT circle(polygon '((0,0),(1,1),(2,0))') AS RESULT;
                  result
  ------------------------------------------
   <(1,0.333333333333333),0.924950591148529>
  (1 row)
  ```

- lseg(box)

  Description: Box diagonal to line segment

  Return type: lseg

  Example:

  ```
  gaussdb=# SELECT lseg(box '((-1,0),(1,0))') AS RESULT;
      result
  ----------------
   [(1,0),(-1,0)]
  (1 row)
  ```

- lseg(point, point)

  Description: Points to line segment

  Return type: lseg

  Example:

  ```
  gaussdb=# SELECT lseg(point '(-1,0)', point '(1,0)') AS RESULT;
      result
  ----------------
   [(-1,0),(1,0)]
  (1 row)
  ```

- slope(point, point)

  Description: Calculates the slope of a straight line formed by two points.

  Return type: double

  Example:

  ```
  gaussdb=# SELECT slope(point '(1,1)', point '(0,0)') AS RESULT;
   result
  --------
       1
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- path(polygon)

  Description: Polygon to path

  Return type: path

  Example:

```
gaussdb=# SELECT path(polygon '((0,0),(1,1),(2,0))') AS RESULT;
      result
--------------------
 ((0,0),(1,1),(2,0))
(1 row)
```

- point(double precision, double precision)

  Description: Points

  Return type: point

  Example:

```
gaussdb=# SELECT point(23.4, -44.5) AS RESULT;
   result
--------------
 (23.4,-44.5)
(1 row)
```

- point(box)

  Description: Center of box

  Return type: point

  Example:

```
gaussdb=# SELECT point(box '((-1,0),(1,0))') AS RESULT;
 result
--------
 (0,0)
(1 row)
```

- point(circle)

  Description: Center of circle

  Return type: point

  Example:

```
gaussdb=# SELECT point(circle '((0,0),2.0)') AS RESULT;
 result
--------
 (0,0)
(1 row)
```

- point(lseg)

  Description: Center of a line segment

  Return type: point

  Example:

```
gaussdb=# SELECT point(lseg '((-1,0),(1,0))') AS RESULT;
 result
--------
 (0,0)
(1 row)
```

- point(polygon)

  Description: Center of a polygon

  Return type: point

  Example:

```
gaussdb=# SELECT point(polygon '((0,0),(1,1),(2,0))') AS RESULT;
      result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
----------------------
 (1,0.333333333333333)
(1 row)
```

- polygon(box)

  Description: Box to 4-point polygon

  Return type: polygon

  Example:

```
gaussdb=# SELECT polygon(box '((0,0),(1,1))') AS RESULT;
          result
---------------------------
 ((0,0),(0,1),(1,1),(1,0))
(1 row)
```

- polygon(circle)

  Description: Circle to 12-point polygon

  Return type: polygon

  Example:

```
gaussdb=# SELECT polygon(circle '((0,0),2.0)') AS RESULT;

result

------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------
 ((-2,0),(-1.73205080756888,1),(-1,1.73205080756888),(-1.22464679914735e-16,2),
(1,1.73205080756888),(1.73205080756888,1),(2,2.44929359829471e-16),
(1.73205080756888,-0.999999999999999),(1,-1.73205080756888),(3.67394039744206e-16,-2),
(-0.999999999999999,-1.73205080756888),(-1.73205080756888,-1))
(1 row)
```

- polygon(npts, circle)

  Description: Circle to npts-point polygon

  Return type: polygon

  Example:

```
gaussdb=# SELECT polygon(12, circle '((0,0),2.0)') AS RESULT;

result

------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------
 ((-2,0),(-1.73205080756888,1),(-1,1.73205080756888),(-1.22464679914735e-16,2),
(1,1.73205080756888),(1.73205080756888,1),(2,2.44929359829471e-16),
(1.73205080756888,-0.999999999999999),(1,-1.73205080756888),(3.67394039744206e-16,-2),
(-0.999999999999999,-1.73205080756888),(-1.73205080756888,-1))
(1 row)
```

- polygon(path)

  Description: Path to polygon

  Return type: polygon

  Example:

```
gaussdb=# SELECT polygon(path '((0,0),(1,1),(2,0))') AS RESULT;
      result
--------------------
 ((0,0),(1,1),(2,0))
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.6.11 Network Address Functions and Operators

cidr and inet Operators

The operators **<<**, **<<=**, **>>**, and **>>=** test for subnet inclusion. These operators compare only the network parts of the two addresses (ignoring any host parts) and determine whether one network is identical to or a subnet of the other.

- <

  Description: Is less than

  Example:

```
gaussdb=# SELECT inet '192.168.1.5' < inet '192.168.1.6' AS RESULT;
 result
--------
 t
(1 row)
```

- <=

  Description: Is less than or equals

  Example:

```
gaussdb=# SELECT inet '192.168.1.5' <= inet '192.168.1.5' AS RESULT;
 result
--------
 t
(1 row)
```

- =

  Description: Equals

  Example:

```
gaussdb=# SELECT inet '192.168.1.5' = inet '192.168.1.5' AS RESULT;
 result
--------
 t
(1 row)
```

- >=

  Description: Is greater than or equals

  Example:

```
gaussdb=# SELECT inet '192.168.1.5' >= inet '192.168.1.5' AS RESULT;
 result
--------
 t
(1 row)
```

- >

  Description: Is greater than

  Example:

```
gaussdb=# SELECT inet '192.168.1.5' > inet '192.168.1.4' AS RESULT;
 result
--------
 t
(1 row)
```

- <>

  Description: Does not equal

  Example:

```
gaussdb=# SELECT inet '192.168.1.5' <> inet '192.168.1.4' AS RESULT;
 result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------
 t
(1 row)
```

- <<

  Description: Is contained in

  Example:

```
gaussdb=# SELECT inet '192.168.1.5' << inet '192.168.1/24' AS RESULT;
 result
--------
 t
(1 row)
```

- <<=

  Description: Is contained in or equals

  Example:

```
gaussdb=# SELECT inet '192.168.1/24' <<= inet '192.168.1/24' AS RESULT;
 result
--------
 t
(1 row)
```

- >>

  Description: Contains

  Example:

```
gaussdb=# SELECT inet '192.168.1/24' >> inet '192.168.1.5' AS RESULT;
 result
--------
 t
(1 row)
```

- >>=

  Description: Contains or equals

  Example:

```
gaussdb=# SELECT inet '192.168.1/24' >>= inet '192.168.1/24' AS RESULT;
 result
--------
 t
(1 row)
```

- ~

  Description: Bitwise NOT

  Example:

```
gaussdb=# SELECT ~ inet '192.168.1.6' AS RESULT;
    result
---------------
 63.87.254.249
(1 row)
```

- &

  Description: Performs an AND operation on each bit of the two network addresses.

  Example:

```
gaussdb=# SELECT inet '192.168.1.6' & inet '10.0.0.0' AS RESULT;
 result
---------
 0.0.0.0
(1 row)
```

- |

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Performs an OR operation on each bit of the two network addresses.

Example:

```
gaussdb=# SELECT inet '192.168.1.6' | inet '10.0.0.0' AS RESULT;
   result
-------------
 202.168.1.6
(1 row)
```

- +

  Description: Addition

  Example:

```
gaussdb=# SELECT inet '192.168.1.6' + 25 AS RESULT;
    result
--------------
 192.168.1.31
(1 row)
```

- -

  Description: Subtraction

  Example:

```
gaussdb=# SELECT inet '192.168.1.43' - 36 AS RESULT;
   result
-------------
 192.168.1.7
(1 row)
```

- -

  Description: Subtraction

  Example:

```
gaussdb=# SELECT inet '192.168.1.43' - inet '192.168.1.19' AS RESULT;
 result
--------
     24
(1 row)
```

cidr and inet Functions

The abbrev, host, and text functions are primarily intended to offer alternative display formats.

- abbrev(inet)

  Description: Abbreviated display format as text

  Return type: text

  Example:

```
gaussdb=# SELECT abbrev(inet '10.1.0.0/16') AS RESULT;
   result
-------------
 10.1.0.0/16
(1 row)
```

- abbrev(cidr)

  Description: Abbreviated display format as text

  Return type: text

  Example:

```
gaussdb=# SELECT abbrev(cidr '10.1.0.0/16') AS RESULT;
 result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
---------
 10.1/16
(1 row)
```

- broadcast(inet)

  Description: Broadcast address for networks

  Return type: inet

  Example:

```
gaussdb=# SELECT broadcast('192.168.1.5/24') AS RESULT;
      result
------------------
 192.168.1.255/24
(1 row)
```

- family(inet)

  Description: Extracts family of address; **4** for IPv4, **6** for IPv6

  Return type: int

  Example:

```
gaussdb=# SELECT family('127.0.0.1') AS RESULT;
 result
--------
      4
(1 row)
```

- host(inet)

  Description: Extracts IP addresses as text.

  Return type: text

  Example:

```
gaussdb=# SELECT host('192.168.1.5/24') AS RESULT;
   result
-------------
 192.168.1.5
(1 row)
```

- hostmask(inet)

  Description: Constructs the host mask for a network.

  Return type: inet

  Example:

```
gaussdb=# SELECT hostmask('192.168.23.20/30') AS RESULT;
 result
---------
 0.0.0.3
(1 row)
```

- masklen(inet)

  Description: Extracts subnet mask length.

  Return type: int

  Example:

```
gaussdb=# SELECT masklen('192.168.1.5/24') AS RESULT;
 result
--------
     24
(1 row)
```

- netmask(inet)

  Description: Constructs the subnet mask for a network.

  Return type: inet

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT netmask('192.168.1.5/24') AS RESULT;
    result
---------------
 255.255.255.0
(1 row)
```

- network(inet)

  Description: Extracts the network part of an address.

  Return type: cidr

  Example:

  ```
  gaussdb=# SELECT network('192.168.1.5/24') AS RESULT;
      result
  ----------------
   192.168.1.0/24
  (1 row)
  ```

- set_masklen(inet, int)

  Description: Sets subnet mask length for the **inet** value.

  Return type: inet

  Example:

  ```
  gaussdb=# SELECT set_masklen('192.168.1.5/24', 16) AS RESULT;
      result
  ----------------
   192.168.1.5/16
  (1 row)
  ```

- set_masklen(cidr, int)

  Description: Sets subnet mask length for the **cidr** value.

  Return type: cidr

  Example:

  ```
  gaussdb=# SELECT set_masklen('192.168.1.0/24'::cidr, 16) AS RESULT;
      result
  ----------------
   192.168.0.0/16
  (1 row)
  ```

- text(inet)

  Description: Extracts IP addresses and subnet mask length as text.

  Return type: text

  Example:

  ```
  gaussdb=# SELECT text(inet '192.168.1.5') AS RESULT;
      result
  ----------------
   192.168.1.5/32
  (1 row)
  ```

Any **cidr** value can be cast to **inet** implicitly or explicitly; therefore, the functions shown above as operating on **inet** also work on **cidr** values. An **inet** value can be cast to **cidr**. After the conversion, any bits to the right of the subnet mask are silently zeroed to create a valid **cidr** value. In addition, you can cast a text string to **inet** or **cidr** using normal casting syntax. For example, **inet(expression)** or **colname::cidr**.

macaddr Functions

trunc(macaddr)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Returns a MAC address with the last three bytes set to zero.

Return type: macaddr

Example:

```
gaussdb=# SELECT trunc(macaddr '12:34:56:78:90:ab') AS RESULT;
     result
-------------------
 12:34:56:00:00:00
(1 row)
```

NOTE

The macaddr type also supports the standard relational operators (such as **>** and **<=**) for lexicographical ordering, and the bitwise arithmetic operators (**~**, **&** and **|**) for NOT, AND and OR.

1.6.12 Text Search Functions and Operators

Text Search Operators

- @@

  Description: Specifies whether the tsvector-typed words match the tsquery-typed words.

  Example:

  ```
  gaussdb=# SELECT to_tsvector('fat cats ate rats') @@ to_tsquery('cat & rat') AS RESULT;
   result
  --------
   t
  (1 row)
  ```

- @@@

  Description: Synonym for @@

  Example:

  ```
  gaussdb=# SELECT to_tsvector('fat cats ate rats') @@@ to_tsquery('cat & rat') AS RESULT;
   result
  --------
   t
  (1 row)
  ```

- ||

  Description: Connects two tsvector-typed words.

  Example:

  ```
  gaussdb=# SELECT 'a:1 b:2'::tsvector || 'c:1 d:2 b:3'::tsvector AS RESULT;
            result
  ----------------------------
   'a':1 'b':2,5 'c':3 'd':4
  (1 row)
  ```

- &&

  Description: Performs the AND operation on two tsquery-typed words.

  Example:

  ```
  gaussdb=# SELECT 'fat | rat'::tsquery && 'cat'::tsquery AS RESULT;
            result
  ----------------------------
   ( 'fat' | 'rat' ) & 'cat'
  (1 row)
  ```

- ||

Description: Performs the OR operation on two tsquery-typed words.

Example:

```
gaussdb=# SELECT 'fat | rat'::tsquery || 'cat'::tsquery AS RESULT;
        result
-----------------------
 'fat' | 'rat' | 'cat'
(1 row)
```

- !!

  Description: Not a tsquery-typed word.

  Example:

  ```
  gaussdb=# SELECT !! 'cat'::tsquery AS RESULT;
   result
  --------
   !'cat'
  (1 row)
  ```

- <->

  Description: Performs the PHRASE operation on two tsquery-typed words.

  Example:

  ```
  gaussdb=# SELECT 'fat'::tsquery <-> 'cat'::tsquery AS RESULT;
      result
  ------------------
   'fat' <-> 'cat'
  (1 row)
  ```

- @>

  Description: Specifies whether a tsquery-typed word contains another tsquery-typed word.

  Example:

  ```
  gaussdb=# SELECT 'cat'::tsquery @> 'cat & rat'::tsquery AS RESULT;
   result
  --------
   f
  (1 row)
  ```

- <@

  Description: Specifies whether a tsquery-typed word is contained in another tsquery-typed word.

  Example:

  ```
  gaussdb=# SELECT 'cat'::tsquery <@ 'cat & rat'::tsquery AS RESULT;
   result
  --------
   t
  (1 row)
  ```

In addition to the preceding operators, the ordinary B-tree comparison operators (including = and <) are defined for types tsvector and tsquery.

Text Search Functions

- get_current_ts_config()

  Description: Obtains default text search configurations.

  Return type: regconfig

  Example:

  ```
  gaussdb=# SELECT get_current_ts_config();
   get_current_ts_config
  -----------------------
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 english
(1 row)
```

- length(tsvector)

  Description: Specifies the number of tsvector-typed words.

  Return type: integer

  Example:

```
gaussdb=# SELECT length('fat:2,4 cat:3 rat:5A'::tsvector);
 length
--------
      3
(1 row)
```

- numnode(tsquery)

  Description: Specifies the number of tsquery-typed words plus operators.

  Return type: integer

  Example:

```
gaussdb=# SELECT numnode('(fat & rat) | cat'::tsquery);
 numnode
---------
       5
(1 row)
```

- plainto_tsquery([config regconfig,] query text)

  Description: Generates tsquery-typed words without punctuations. The AND operator (&) is used as the connection operator.

  Return type: tsquery

  Example:

```
gaussdb=# SELECT plainto_tsquery('english', 'The Fat Rats');
 plainto_tsquery
-----------------
 'fat' & 'rat'
(1 row)
```

- phraseto_tsquery([config regconfig,] query text)

  Description: Generates tsquery-typed words without punctuations. The PHRASE operator (<->) is used as the connection operator. **config** is the token configuration dictionary. For details, see **Dictionaries**.

  Return type: tsquery

  Example:

```
gaussdb=# SELECT phraseto_tsquery('english', 'The Fat Rats');
 phraseto_tsquery
------------------
 'fat' <-> 'rat'
(1 row)
```

- tsquery_phrase(query1 tsquery, query2 tsquery [, distance regdistance])

  Description: Connects tsquery types using the PHRASE operator. **query1** and **query2** can use data of the text type, which will be converted to the tsquery type. **distance** can be used to specify the distance between **query1** and **query2**.

  Return type: tsquery

  Example:

```
-- If the distance is not specified, use the default value 1.
gaussdb=# SELECT tsquery_phrase(to_tsquery('fat'), to_tsquery('cat'));
 tsquery_phrase
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-----------------
 'fat' <-> 'cat'
(1 row)

-- Specify the distance.
gaussdb=# SELECT tsquery_phrase(to_tsquery('fat'), to_tsquery('cat'), 2);
 tsquery_phrase
-----------------
 'fat' <2> 'cat'
(1 row)
```

- querytree(query tsquery)

  Description: Obtains the indexable part of a tsquery-typed word.

  Return type: text

  Example:

```
gaussdb=# SELECT querytree('foo & ! bar'::tsquery);
 querytree
-----------
 'foo'
(1 row)
```

- setweight(tsvector, "char")

  Description: Assigns weight to each tsvector. The weight value can be **'A'**, **'B'**, **'C'**, or **'D'**.

  Return type: tsvector

  Example:

```
gaussdb=# SELECT setweight('fat:2,4 cat:3 rat:5B'::tsvector, 'A');
           setweight
-------------------------------
 'cat':3A 'fat':2A,4A 'rat':5A
(1 row)
```

- strip(tsvector)

  Description: Removes positions and weights from tsvector.

  Return type: tsvector

  Example:

```
gaussdb=# SELECT strip('fat:2,4 cat:3 rat:5A'::tsvector);
       strip
-------------------
 'cat' 'fat' 'rat'
(1 row)
```

- to_tsquery([config regconfig ,] query text)

  Description: Normalizes words and converts them to tsquery. For details, see **Parsing Queries**.

  Return type: tsquery

  Example:

```
gaussdb=# SELECT to_tsquery('english', 'The & Fat & Rats');
  to_tsquery
---------------
 'fat' & 'rat'
(1 row)
```

- to_tsvector([config regconfig ,] document text)

  Description: Standardizes the document content, removes stop words, sorts the content, deduplicates it, and then regenerates it to the tsvector type.

  Return type: tsvector

  Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT to_tsvector('english', 'The Fat Rats');
   to_tsvector
-----------------
 'fat':2 'rat':3
(1 row)
```

- to_tsvector_for_batch([config regconfig ,] document text)

  Description: Standardizes the document content, removes stop words, sorts the content, deduplicates it, and then regenerates it to the tsvector type.

  Return type: tsvector

  Example:

```
gaussdb=# SELECT to_tsvector_for_batch('english', 'The Fat Rats');
   to_tsvector
-----------------
 'fat':2 'rat':3
(1 row)
```

- ts_headline([config regconfig,] document text, query tsquery [, options text])

  Description: Highlights a query match. For details, see **Highlighting Search Results**.

  Return type: text

  Example:

```
gaussdb=# SELECT ts_headline('x y z', 'z'::tsquery);
 ts_headline
--------------
 x y <b>z</b>
(1 row)
```

- ts_rank([weights float4[],] vector tsvector, query tsquery [, normalization integer])

  Description: Ranks document for query. For details, see **Ranking Search Results**.

  Return type: float4

  Example:

```
gaussdb=# SELECT ts_rank('hello world'::tsvector, 'world'::tsquery);
 ts_rank
----------
 .0607927
(1 row)
```

- ts_rank_cd([weights float4[],] vector tsvector, query tsquery [, normalization integer])

  Description: Ranks document for query using cover density. For details, see **Ranking Search Results**.

  Return type: float4

  Example:

```
gaussdb=# SELECT ts_rank_cd('hello world'::tsvector,'world'::tsquery);
 ts_rank_cd
------------
          0
(1 row)
```

- ts_rewrite(query tsquery, target tsquery, substitute tsquery)

  Description: Replaces tsquery-typed word. For details, see **Rewriting Queries**.

  Return type: tsquery

  Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT ts_rewrite('a & b'::tsquery, 'a'::tsquery, 'foo|bar'::tsquery);
     ts_rewrite
------------------------
 'b' & ( 'foo' | 'bar' )
(1 row)
```

- ts_rewrite(query tsquery, select text)

  Description: Replaces tsquery data in the target with the result of a **SELECT** command. For details, see **Rewriting Queries**.

  Return type: tsquery

  Example:

```
gaussdb=# SELECT ts_rewrite('world'::tsquery, 'select ''world''::tsquery, ''hello''::tsquery');
 ts_rewrite
------------
 'hello'
(1 row)
```

Text Search Debugging Functions

NOTICE

This function is for internal use only. You are advised not to use it.

- ts_debug([config regconfig,] document text, OUT alias text, OUT description text, OUT token text, OUT dictionaries regdictionary[], OUT dictionary regdictionary, OUT lexemes text[])

  Description: Tests a configuration.

  Return type: SETOF record

  Example:

```
gaussdb=# SELECT ts_debug('english', 'The Brightest supernovaes');
                                ts_debug
-------------------------------------------------------------------------------
 (asciiword,"Word, all ASCII",The,{english_stem},english_stem,{})
 (blank,"Space symbols"," ",{},,)
 (asciiword,"Word, all ASCII",Brightest,{english_stem},english_stem,{brightest})
 (blank,"Space symbols"," ",{},,)
 (asciiword,"Word, all ASCII",supernovaes,{english_stem},english_stem,{supernova})
(5 rows)
```

- ts_lexize(dict regdictionary, token text)

  Description: Tests a data dictionary.

  Return type: text[]

  Example:

```
gaussdb=# SELECT ts_lexize('english_stem', 'stars');
 ts_lexize
-----------
 {star}
(1 row)
```

- ts_parse(parser_name text, document text, OUT tokid integer, OUT token text)

  Description: Tests a parser.

  Return type: SETOF record

  Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT ts_parse('default', 'foo - bar');
 ts_parse
-----------
 (1,foo)
 (12," ")
 (12,"- ")
 (1,bar)
(4 rows)
```

- ts_parse(parser_oid oid, document text, OUT tokid integer, OUT token text)

  Description: Tests a parser.

  Return type: SETOF record

  Example:

```
gaussdb=# SELECT ts_parse(3722, 'foo - bar');
 ts_parse
-----------
 (1,foo)
 (12," ")
 (12,"- ")
 (1,bar)
(4 rows)
```

- ts_token_type(parser_name text, OUT tokid integer, OUT alias text, OUT description text)

  Description: Obtains token types defined by a parser.

  Return type: SETOF record

  Example:

```
gaussdb=# SELECT ts_token_type('default');
                      ts_token_type
-----------------------------------------------------------------
 (1,asciiword,"Word, all ASCII")
 (2,word,"Word, all letters")
 (3,numword,"Word, letters and digits")
 (4,email,"Email address")
 (5,url,URL)
 (6,host,Host)
 (7,sfloat,"Scientific notation")
 (8,version,"Version number")
 (9,hword_numpart,"Hyphenated word part, letters and digits")
 (10,hword_part,"Hyphenated word part, all letters")
 (11,hword_asciipart,"Hyphenated word part, all ASCII")
 (12,blank,"Space symbols")
 (13,tag,"XML tag")
 (14,protocol,"Protocol head")
 (15,numhword,"Hyphenated word, letters and digits")
 (16,asciihword,"Hyphenated word, all ASCII")
 (17,hword,"Hyphenated word, all letters")
 (18,url_path,"URL path")
 (19,file,"File or path name")
 (20,float,"Decimal notation")
 (21,int,"Signed integer")
 (22,uint,"Unsigned integer")
 (23,entity,"XML entity")
(23 rows)
```

- ts_token_type(parser_oid oid, OUT tokid integer, OUT alias text, OUT description text)

  Description: Obtains token types defined by a parser.

  Return type: SETOF record

  Example:

```
gaussdb=# SELECT ts_token_type(3722);
                      ts_token_type
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
------------------------------------------------------------
 (1,asciiword,"Word, all ASCII")
 (2,word,"Word, all letters")
 (3,numword,"Word, letters and digits")
 (4,email,"Email address")
 (5,url,URL)
 (6,host,Host)
 (7,sfloat,"Scientific notation")
 (8,version,"Version number")
 (9,hword_numpart,"Hyphenated word part, letters and digits")
 (10,hword_part,"Hyphenated word part, all letters")
 (11,hword_asciipart,"Hyphenated word part, all ASCII")
 (12,blank,"Space symbols")
 (13,tag,"XML tag")
 (14,protocol,"Protocol head")
 (15,numhword,"Hyphenated word, letters and digits")
 (16,asciihword,"Hyphenated word, all ASCII")
 (17,hword,"Hyphenated word, all letters")
 (18,url_path,"URL path")
 (19,file,"File or path name")
 (20,float,"Decimal notation")
 (21,int,"Signed integer")
 (22,uint,"Unsigned integer")
 (23,entity,"XML entity")
(23 rows)
```

- ts_stat(sqlquery text, [weights text,] OUT word text, OUT ndoc integer, OUT nentry integer)

  Description: Obtains statistics of a tsvector column.

  Return type: SETOF record

  Example:

```
gaussdb=# SELECT ts_stat('select ''hello world''::tsvector');
   ts_stat
-------------
 (world,1,1)
 (hello,1,1)
(2 rows)
```

1.6.13 JSON/JSONB Functions and Operators

For details about the JSON/JSONB data type, see **JSON/JSONB Types**. For details about the operator information, see **Table 1-77** and **Table 1-78**.

Table 1-77 Common JSON/JSONB operators

Operators	Left Operand Type	Right Operand Type	Return Type	Description	Example
->	Array-json(b)	int	json(b)	Obtains the **array-json** element. If the index does not exist, **NULL** is returned.	SELECT '[{"a":"foo"},{"b":"bar"},{"c":"baz"}]'::json->2; ?column? ------------- {"c":"baz"} (1 row)
->	object-json(b)	text	json(b)	Obtains the value by a key. If no record is found, **NULL** is returned.	SELECT '{"a": {"b":"foo"}}'::json->'a'; ?column? ------------- {"b":"foo"} (1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operators	Left Operand Type	Right Operand Type	Return Type	Description	Example
->>	Array-json(b)	int	text	Obtains JSON array elements. If the index does not exist, **NULL** is returned.	SELECT '[1,2,3]'::json->>2; ?column? ---------- 3 (1 row)
->>	object-json(b)	text	text	Obtains the value by a key. If no record is found, **NULL** is returned.	SELECT '{"a":1,"b":2}'::json->>'b'; ?column? ---------- 2 (1 row)
#>	container-json (b)	text[]	json(b)	Obtains the JSON object in the specified path. If the path does not exist, **NULL** is returned.	SELECT '{"a": {"b":{"c": "foo"}}}'::json #>'{a,b}'; ?column? -------------- {"c": "foo"} (1 row)
#>>	container-json (b)	text[]	text	Obtains the JSON object in the specified path. If the path does not exist, **NULL** is returned.	SELECT '{"a":[1,2,3],"b":[4,5,6]}'::json #>>'{a,2}'; ?column? ---------- 3 (1 row)

CAUTION

For the #> and #>> operators, if no data is found in the specified path, they will return null instead of reporting an error.

Table 1-78 Additional JSONB support for operators

Operators	Right Operand Type	Description	Example
@>	jsonb	Specifies whether the top layer of the JSON on the left contains all items of the top layer of the JSON on the right.	'{"a":1, "b":2}'::jsonb @> '{"b":2}'::jsonb

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operators	Right Operand Type	Description	Example
<@	jsonb	Specifies whether all items in the JSON file on the left exist at the top layer of the JSON file on the right.	'{"b":2}'::jsonb <@ '{"a":1, "b":2}'::jsonb
?	text	Specifies whether the string of the key or element exists at the top layer of the JSON value.	'{"a":1, "b":2}'::jsonb ? 'b'
?|	text[]	Specifies whether any of these array strings exists as top-layer keys.	'{"a":1, "b":2, "c":3}'::jsonb ?| array['b', 'c']
?&	text[]	Specifies whether all these array strings exist as top-layer keys.	'["a", "b"]'::jsonb ?& array['a', 'b']
=	jsonb	Determines the size between two **JSONB** files, which is the same as the **jsonb_eq** function.	/
<>	jsonb	Determines the size between two **JSONB** files, which is the same as the **jsonb_ne** function.	/
<	jsonb	Determines the size between two **JSONB** files, which is the same as the **jsonb_lt** function.	/
>	jsonb	Determines the size between two **JSONB** files, which is the same as the **jsonb_gt** function.	/

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operators	Right Operand Type	Description	Example
<=	jsonb	Determines the size between two **JSONB** files, which is the same as the **jsonb_le** function.	/
>=	jsonb	Determines the size between two **JSONB** files, which is the same as the **jsonb_ge** function.	/

NOTE

GIN indexes can be used to effectively search for keys or key-value pairs that appear in a large number of JSONB documents (datums). Two GIN operator classes (jsonb_ops and jsonb_hash_ops) are provided for different performance and flexibility choices.

- The default GIN operator class jsonb_ops supports queries using the @>, ?, ?&, and ?| operators.
- The non-default GIN operator class jsonb_hash_ops supports indexing the @> operator only.

Functions Supported by JSON/JSONB

- array_to_json(anyarray [, pretty_bool])

  Description: Returns the array as JSON. It combines a multi-dimensional array into a JSON array. Line feeds will be added between one-dimensional elements if **pretty_bool** is **true**.

  Return type: json

  Example:

```
gaussdb=# SELECT array_to_json('{{1,5},{99,100}}'::int[]);
  array_to_json
------------------
 [[1,5],[99,100]]
(1 row)
```

- row_to_json(record [, pretty_bool])

  Description: Returns the row as JSON. Line feeds will be added between level-1 elements if **pretty_bool** is **true**.

  Return type: json

  Example:

```
gaussdb=# SELECT row_to_json(row(1,'foo'));
    row_to_json
---------------------
 {"f1":1,"f2":"foo"}
(1 row)
```

- json_array_element(array-json, integer), jsonb_array_element(array-jsonb, integer)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Same as the operator `->`, which returns the element with the specified index in the array.

Return type: json, jsonb

Example:

```
gaussdb=# SELECT json_array_element('[1,true,[1,[2,3]],null]',2);
 json_array_element
--------------------
 [1,[2,3]]
(1 row)
```

- json_array_element_text(array-json, integer), jsonb_array_element_text(array-jsonb, integer)

  Description: Same as the operator `->>`, which returns the element with the specified index in the array.

  Return type: text, text

  Example:

  ```
  gaussdb=# SELECT json_array_element_text('[1,true,[1,[2,3]],null]',2);
  json_array_element_text
  ------------------------
   [1,[2,3]]
  (1 row)
  ```

- json_object_field(object-json, text), jsonb_object_field(object-jsonb, text)

  Description: Same as the operator **`->`**, which returns the value of a specified key in an object.

  Return type: json, json

  Example:

  ```
  gaussdb=# SELECT json_object_field('{"a": {"b":"foo"}}','a');
   json_object_field
  -------------------
   {"b":"foo"}
  (1 row)
  ```

- json_object_field_text(object-json, text), jsonb_object_field_text(object-jsonb, text)

  Description: Same as the operator **`->>`**, which returns the value of a specified key in an object.

  Return type: text, text

  Example:

  ```
  gaussdb=# SELECT json_object_field_text('{"a": {"b":"foo"}}','a');
  json_object_field_text
  ----------------------
   {"b":"foo"}
  (1 row)
  ```

- json_extract_path(json, VARIADIC text[]), jsonb_extract_path((jsonb, VARIADIC text[])

  Description: It is equivalent to the operator `#>`, searches for **json** based on the path specified by the second parameter, and returns the result.

  Return type: json, jsonb

  Example:

  ```
  gaussdb=# SELECT json_extract_path('{"f2":{"f3":1},"f4":{"f5":99,"f6":"stringy"}}', 'f4','f6');
   json_extract_path
  -------------------
   "stringy"
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- json_extract_path_op(json, text[]), jsonb_extract_path_op(jsonb, text[])

  Description: It is equivalent to the operator `#>`, searches for **json** based on the path specified by the second parameter, and returns the result.

  Return type: json, jsonb

  Example:

```
gaussdb=# SELECT json_extract_path_op('{"f2":{"f3":1},"f4":{"f5":99,"f6":"stringy"}}', ARRAY['f4','f6']);
 json_extract_path_op
----------------------
 "stringy"
(1 row)
```

- json_extract_path_text(json, VARIADIC text[]), jsonb_extract_path_text((jsonb, VARIADIC text[])

  Description: It is equivalent to the operator `#>>`, searches for **json** based on the path specified by the second parameter, and returns the result.

  Return type: text, text

  Example:

```
gaussdb=# SELECT json_extract_path_text('{"f2":{"f3":1},"f4":{"f5":99,"f6":"stringy"}}', 'f4','f6');
 json_extract_path_text
------------------------
 stringy
(1 row)
```

- json_extract_path_text_op(json, text[]), jsonb_extract_path_text_op(jsonb, text[])

  Description: It is equivalent to the operator `#>>`, searches for **json** based on the path specified by the second parameter, and returns the result.

  Return type: text, text

  Example:

```
gaussdb=# SELECT json_extract_path_text_op('{"f2":{"f3":1},"f4":{"f5":99,"f6":"stringy"}}',
ARRAY['f4','f6']);
 json_extract_path_text_op
---------------------------
 stringy
(1 row)
```

- json_array_elements(array-json), jsonb_array_elements(array-jsonb)

  Description: Splits an array. Each element returns a row.

  Return type: json, jsonb

  Example:

```
gaussdb=# SELECT json_array_elements('[1,true,[1,[2,3]],null]');
 json_array_elements
---------------------
 1
 true
 [1,[2,3]]
 null
(4 rows)
```

- json_array_elements_text(array-json), jsonb_array_elements_text(array-jsonb)

  Description: Splits an array. Each element returns a row.

  Return type: text, text

  Example:

```
gaussdb=# SELECT * FROM  json_array_elements_text('[1,true,[1,[2,3]],null]');
   value
-----------
 1
 true
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 [1,[2,3]]

(4 rows)
```

- json_array_length(array-json), jsonb_array_length(array-jsonb)

  Description: Returns the array length.

  Return type: integer

  Example:

```
gaussdb=# SELECT json_array_length('[1,2,3,{"f1":1,"f2":[5,6]},4,null]');
 json_array_length
-------------------
                 6
(1 row)
```

- json_each(object-json), jsonb_each(object-jsonb)

  Description: Splits each key-value pair of an object into one row and two columns.

  Return type: setof(key text, value json), setof(key text, value jsonb)

  Example:

```
gaussdb=# SELECT * FROM  json_each('{"f1":[1,2,3],"f2":{"f3":1},"f4":null}');
 key |  value
-----+----------
 f1  | [1,2,3]
 f2  | {"f3":1}
 f4  | null
(3 rows)
```

- json_each_text(object-json), jsonb_each_text(object-jsonb)

  Description: Splits each key-value pair of an object into one row and two columns.

  Return type: setof(key text, value text), setof(key text, value text)

  Example:

```
gaussdb=# SELECT * FROM  json_each_text('{"f1":[1,2,3],"f2":{"f3":1},"f4":null}');
 key |  value
-----+----------
 f1  | [1,2,3]
 f2  | {"f3":1}
 f4  |
(3 rows)
```

- json_object_keys(object-json), jsonb_object_keys(object-jsonb)

  Description: Returns all keys at the top layer of the object.

  Return type: SETOF text

  Example:

```
gaussdb=# SELECT json_object_keys('{"f1":"abc","f2":{"f3":"a", "f4":"b"}, "f1":"abcd"}');
 json_object_keys
------------------
 f1
 f2
 f1
(3 rows)
```

```
--The jsonb_object_keys function involves deduplication.
gaussdb=# SELECT jsonb_object_keys('{"f1":"abc","f2":{"f3":"a", "f4":"b"}, "f1":"abcd"}');
 jsonb_object_keys
-------------------
 f1
 f2
(2 rows)
```

- json_populate_record(anyelement, object-json [, bool]), jsonb_populate_record(anyelement, object-jsonb [, bool])

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Splits each key-value pair in the second parameter, uses the key as the column name, matches the column name in the first parameter (which must be of the composite type), and fills the value in the format of the first parameter.

Return type: anyelement, anyelement

Example:

```
gaussdb=# CREATE TYPE jpop AS (a text, b int, c bool);
CREATE TYPE
gaussdb=# SELECT * FROM json_populate_record(null::jpop,'{"a":"blurfl","x":43.2}');
   a    | b | c
--------+---+---
 blurfl |   |
(1 row)
```

```
gaussdb=# SELECT * FROM json_populate_record((1,1,null)::jpop,'{"a":"blurfl","x":43.2}');
   a    | b | c
--------+---+---
 blurfl | 1 |
(1 row)
gaussdb=# DROP TYPE jpop;
DROP TYPE
```

- json_populate_recordset(anyelement, array-json [, bool]), jsonb_populate_recordset(anyelement, array-jsonb [, bool])

  Description: Processes each element in the array of the second parameter by utilizing the same method as the json_populate_record and jsonb_populate_record functions. Therefore, each element in the array of the second parameter must be of the object-json type.

  Return type: setof anyelement, setof anyelement

  Example:

```
gaussdb=# CREATE TYPE jpop AS (a text, b int, c bool);
CREATE TYPE
gaussdb=# SELECT * FROM json_populate_recordset(null::jpop, '[{"a":1,"b":2},{"a":3,"b":4}]');
 a | b | c
---+---+---
 1 | 2 |
 3 | 4 |
(2 rows)
gaussdb=# DROP TYPE jpop;
DROP TYPE
```

- json_typeof(json), jsonb_typeof(jsonb)

  Description: Checks the JSON type.

  Return type: text, text

  Example:

```
gaussdb=# SELECT value, json_typeof(value)  FROM (values (json '123.4'), (json '"foo"'), (json 'true'),
(json 'null'), (json '[1, 2, 3]'), (json '{"x":"foo", "y":123}'), (NULL::json))  AS data(value);
         value          | json_typeof
------------------------+-------------
 123.4                  | number
 "foo"                  | string
 true                   | boolean
 null                   | null
 [1, 2, 3]              | array
 {"x":"foo", "y":123} | object
                        |
(7 rows)
```

- json_build_array([VARIADIC "any"])

  Description: Constructs a JSON array from a variable parameter list.

  Return type: array-json

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT json_build_array('a',1,'b',1.2,'c',true,'d',null,'e',json '{"x": 3, "y": [1,2,3]}','');
                          json_build_array
-------------------------------------------------------------------------------
 ["a", 1, "b", 1.2, "c", true, "d", null, "e", {"x": 3, "y": [1,2,3]}, null]
(1 row)
```

- json_build_object([VARIADIC "any"])

  Description: Constructs a JSON object from a variable parameter list. The number of input parameters must be an even number. Every two input parameters form a key-value pair. Note that the key value cannot be **NULL**.

  Return type: object-json

  Example:

```
gaussdb=# SELECT json_build_object(1,2);
 json_build_object
-------------------
 {"1" : 2}
(1 row)
```

- jsonb_build_object([VARIADIC "any"])

  Description: Constructs a JSONB object from a variable parameter list. The number of input parameters must be an even number. Every two input parameters form a key-value pair. Note that the key value cannot be **NULL**.

  Return type: object-jsonb

NOTICE

- When an element in the variable parameter list contains an empty string (''), if the SQL compatibility mode is Oracle-compatible, the return result of the corresponding element is **NULL**; if the SQL compatibility mode of the database is PG-compatible, the return result of the corresponding element is an empty string. This is because empty strings are treated as **NULL** in ORA-compatible mode.
- If an element in the variable parameter list is of the DATE type, and **mapping_date_to_datea** is disabled in ORA-compatible mode, the DATE type is replaced by the TIMESTAMP (0) WITHOUT TIME ZONE type. When **mapping_date_to_datea** is enabled, the DATE type is replaced by the DATEA type. Therefore, the format of the returned result of the corresponding element position is affected by whether **mapping_date_to_datea** is enabled. In PG-compatible mode, the returned result of the corresponding element position does not contain hour, minute, or second.

Example:

```
gaussdb=# SELECT jsonb_build_object('a',1,'b',1.2,'c',true,'d',null,'e',json '{"x": 3, "y": [1,2,3]}');
                         jsonb_build_object
-------------------------------------------------------------------------------
 {"a": 1, "b": 1.2, "c": true, "d": null, "e": {"x": 3, "y": [1, 2, 3]}}
(1 row)

gaussdb=# SELECT jsonb_build_object();
 jsonb_build_object
--------------------
 {}
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- json_to_record(object-json, bool), json_to_record(object-json)

  Description: Like all functions that return **record**, the caller must explicitly define the structure of the record with an AS clause. The key-value pairs in **object-json** are split and reassembled. The keys serve as column names to be matched with and populated into the structure of the record explicitly specified by the AS clause. The values of the keys must follow the structure of the relevant data type. The date type is equivalent to TIMESTAMP(0) WITHOUT TIME ZONE in ORA-compatible mode when **mapping_date_to_datea** is disabled. Compared with that in PG-compatible mode, the value also displays the 00:00:00 (hour, minute, and second) character string.

  Parameter: The Boolean input parameter of the first overloaded function specifies whether object nesting is supported. The value **true** indicates that object nesting is supported, and the value **false** indicates that object nesting is not supported. If there is only one input parameter, nesting is supported by default. In this case, the nested JSON object can be converted into a row type and parsed as an array enclosed in square brackets.

  Return type: record

  Example:

  ```
  gaussdb=# SELECT * FROM json_to_record('{"a":1,"b":"foo","c":"bar"}',true) AS x(a int, b text, d text);
    a |  b  | d
   ---+-----+---
    1 | foo |
   (1 row)
  gaussdb=# SELECT * FROM json_to_record('{"a": {"x": 1, "y": 2},"b":"foo","c":[1, 2]}') AS x(a json, b
  text, c int[]);
          a         | b |   c
   -----------------+-----+-------
    {"x": 1, "y": 2} | foo | {1,2}
   (1 row)
  ```

- json_to_recordset(array-json, bool)

  Description: Executes the preceding function on each element in the array by referring to the json_to_record function. Therefore, each element in the array must be object-json. For **bool**, refer to the json_to_record function. **true** indicates that nested objects can be parsed, and **false** indicates that nested objects cannot be parsed, that is, whether the value of an element in a JSON object can be a JSON object.

  Return type: SETOF record

  Example:

  ```
  gaussdb=# SELECT * FROM json_to_recordset('[{"a":1,"b":"foo","d":false},
  {"a":2,"b":"bar","c":true}]',false) AS x(a int, b text, c boolean);
   a |  b  | c
  ---+-----+---
   1 | foo |
   2 | bar | t
  (2 rows)
  ```

- json_object(text[]), json_object(text[], text[])

  Description: Constructs an **object-json** from a text array. This is an overloaded function. When the input parameter is a text array, the array length must be an even number, and members are considered as alternate key-value pairs. When two text arrays are used, the first array is considered as a key, and the second array a value. The lengths of the two arrays must be the same. The value of a key cannot be **NULL**.

  Return type: object-json

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT json_object('{a,1,b,2,3,NULL,"d e f","a b c"}');
                    json_object
-----------------------------------------------------------
 {"a" : "1", "b" : "2", "3" : null, "d e f" : "a b c"}
(1 row)
gaussdb=# SELECT json_object('{a,b,"a b c"}', '{a,1,1}');
              json_object
---------------------------------------
 {"a" : "a", "b" : "1", "a b c" : "1"}
(1 row)
```

- json_object([VARIADIC "any"])

  Description: Constructs a JSON object from a variable parameter list. The number of input parameters must be an even number. Every two input parameters form a key-value pair. If the key-value pair is null, an error is reported. If the parameter is in the odd format, an error is reported.

  Parameter: variable parameter list. The input parameter is a combination of one or more key-value pairs.

  Return type: json

  Example:

```
gaussdb=# SELECT json_object('d',2,'c','name','b',true,'a',2,'a',NULL,'d',1);
               json_object
-----------------------------------------
 {"a": 2, "b": true, "c": "name", "d": 2}
(1 row)

gaussdb=# SELECT json_object('d',2,true,'name','b',true,'a',2,'aa', current_timestamp);
                                json_object
---------------------------------------------------------------------------
 {"1": "name", "a": 2, "b": true, "d": 2, "aa": "2023-08-12 11:28:13.385958"}
(1 row)
```

  NOTE

  This function takes effect in a MySQL-compatible database when **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s1'**. When this function takes effect, the original json_object behavior is replaced.

- json_append/json_array_append(json, [VARIADIC "any"])

  Description: Constructs several **json_path**-**value** pairs from a variable-parameter list, appends a value to the path specified in **json**, and returns the modified **json**. json_append is the same as json_array_append. If any parameter is null, **null** is returned. If the JSON format is incorrect, **json_path** is an invalid path expression, or **json_path** contains * or **, an error is reported. If the variable parameter list contains null values and has format error, the exceptions are processed based on the exception sequence. If a null value exists first, **null** is returned first. If a format error occurs first, an error is reported first.

  Parameters: For details, see **Table 1-79**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-79 Parameters

Parameter	Type	Description	Value Range
json	JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, NVARCHAR2	JSON value to be modified. If other non-restricted types are passed, an error will be reported during execution.	The value must be in JSON format.
[VARIADIC "any"]	Variable any array	A variable parameter list, including variable json_path-value pairs. The **json_path**-**value** pair will be constructed from the variable parameter list.	**json_path** must be a valid path expression, and **value** can be any value.

Return type: json

Example:

```
gaussdb=# SELECT json_array_append('[1, [2, 3]]', '$[1]', 4, '$[0]', false, '$[0]', null, '$[0]',
current_timestamp);
                    json_array_append
----------------------------------------------------------------
 [[1, false, null, "2023-08-12 14:27:16.142355+08"], [2, 3, 4]]
(1 row)
```

- json_array([VARIADIC "any"])

  Description: Constructs an array from a variable parameter list and returns a JSON array. If the function does not have any parameters, an empty JSON array is returned.

  Parameter: The input parameter is a variable parameter list. The values in the list can be of any type.

  Return type: json

  Example:

  ```
  -- If no input parameter is entered, an empty JSON array is returned.
  gaussdb=# SELECT json_array();
  ))::numeric >= $0) json_array
  ------------
   []
  (1 row)

  -- The input parameter can be of any type.
  gaussdb=# SELECT json_array(TRUE, FALSE, NULL, 114, 'text', current_timestamp);
                            json_array
  ------------------------------------------------------------------
   [true, false, null, 114, "text", "2023-08-12 15:17:34.979294+08"]
  (1 row)
  ```

- json_array_insert(json, [VARIADIC "any"])

  Description: Constructs one or more **json_path**-**value** pairs from a variable parameter list, inserts a value into the array path specified by the **json_path**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

in **json**, and returns a new JSON value. If there is already an existing value on the specified path, the value is inserted on the path and the existing value is moved backward. If **json** or **json_path** is null, **null** is returned. If the JSON format is incorrect, **json_path** is an invalid path expression, or **json_path** contains * or **, an error is reported. If the variable parameter list contains null values and has format error, the exceptions are processed based on the exception sequence. If a null value exists first, **null** is returned first. If a format error occurs first, an error is reported first.

Parameters: See **Table 1-80**.

Table 1-80 Parameters

Parameter	Type	Description	Value Range
json	JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, NVARCHAR2	JSON value to be modified. If other non-restricted types are passed, an error will be reported during execution.	The value must be in JSON format.
[VARIADIC "any"]	Variable any array	Variable parameter list, containing one or more json_path-value pairs.	**json_path** must be a valid path expression, and **value** can be any value.

Return type: json

Example:

```
-- Example of containing one json_path-value pair
gaussdb=# SELECT json_array_insert('[1, [2, 3]]', '$[1]', 4);
 json_array_insert
-------------------
 [1, 4, [2, 3]]
(1 row)

-- Example of containing multiple json_path-value pairs
gaussdb=# SELECT json_array_insert('{"x": 1, "y": [1, 2]}', '$.y[0]', NULL, '$.y[0]', 123, '$.y[3]',
current_timestamp);
                        json_array_insert
------------------------------------------------------------------
 {"x": 1, "y": [123, null, 1, "2023-08-14 14:54:12.85087+08", 2]}
(1 row)
```

- json_contains(target_json, candidate_json[, json_path])

  Description: The input parameters are two JSON objects and an optional path specified by **json_path**. If **json_path** is not specified, the system checks whether **target_json** contains **candidate_json**. If **json_path** is specified, the system checks whether **candidate_json** is contained in the JSON path specified by **json_path** in **target_json**. If **json** or **json_path** is null, **null** is returned. If the JSON format is incorrect, **json_path** is an invalid path

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

expression, or **json_path** contains * or **, an error is reported. If the variable parameter list contains null values and has format error, the exceptions are processed based on the exception sequence. If a null value exists first, **null** is returned first. If a format error occurs first, an error is reported first.

Parameters: See **Table 1-81**.

Table 1-81 Parameters

Parameter	Type	Description	Value Range
target_json	JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, NVARCHAR2	Target JSON object, which is used to check whether **candidate_json** is contained in the JSON object. If other non-restricted types are passed, an error will be reported during execution.	The value must be in JSON format.
candidate_json	JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, NVARCHAR2	JSON subobject, which is used to check whether it is contained in **target_json**. If other non-restricted types are passed, an error will be reported during execution.	The value must be in JSON format.
json_path	text	Specified JSON path. This parameter is optional. If this parameter is specified, the path specified by **json_path** in **target_json** is used to determine the inclusion relationship.	**json_path** must be a valid path expression.

Return type: bigint. If there is an inclusion relationship, **1** is returned. If there is no inclusion relationship, **0** is returned.

Example:

```
-- No path is specified.
gaussdb=# SELECT json_contains('[1, 2, {"x": 3}]', '{"x":3}');
 json_contains
---------------
             1
(1 row)

-- A path is specified.
gaussdb=# SELECT json_contains('[1, 2, {"x": 3},[1,2,3,4]]', '2','$[1]');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 json_contains
---------------
             1
(1 row)

gaussdb=# SELECT json_contains('[1, 2, {"x": 3},[1,2,3,4]]', '1','$[1]');
 json_contains
---------------
             0
(1 row)
```

- json_contains_path(json, mode_str, [VARIADIC text])

  Description: Checks whether a JSON object has a value on the specified path. There can be multiple paths. The first input parameter is a JSON object, and the second input parameter can be **one** or **all**, specifying whether to check all paths. The third input parameter is a variable parameter list. All JSON paths are constructed from the variable parameter list. According to the mode, if a path value exists, **1** is returned. If no path value exists, **0** is returned. If **json** or **json_path** is null, **null** is returned. If the JSON format is incorrect, or **json_path** is an invalid path expression, an error is reported. If the variable parameter list contains null values and has format error, the exceptions are processed based on the exception sequence. If a null value exists first, **null** is returned first. If a format error occurs first, an error is reported first.

  Parameters: See **Table 1-82**.

Table 1-82 Parameters

Parameter	Type	Description	Value Range
json	JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, NVARCHAR2	JSON object to be specified. If other non-restricted types are passed, an error will be reported during execution.	The value must be in JSON format.
mode_str	text	The value can be **one** or **all**. If the value is **one** and one path exists, **1** is returned; otherwise, **0** is returned. If the value is **all** and all paths exist, **1** is returned; otherwise, **0** is returned.	The value must be **one** or **all**, which is case-insensitive.
[VARIADIC text]	Variable text array	Variable parameter list from which all JSON paths are constructed.	**json_path** must be a valid path expression.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: bigint

Example:

```
-- In the all scenario, all paths must exist.
gaussdb=# SELECT json_contains_path('[1, 2, {"x": 3}]', 'all', '$[0]', '$[1]', '$[2]');
 json_contains_path
--------------------
                  1
(1 row)

gaussdb=# SELECT json_contains_path('[1, 2, {"x": 3}]', 'all', '$[0]', '$[1]', '$[6]');
 json_contains_path
--------------------
                  0
(1 row)

-- In the one scenario, at least one path must exist.
gaussdb=# SELECT json_contains_path('[1, 2, {"x": 3}]', 'one', '$[0]', '$[1]', '$[5]');
 json_contains_path
--------------------
                  1
(1 row)
```

- json_depth(json)

  Description: Returns the maximum depth of the input JSON object. If the input parameter is null, **null** is returned.

  Parameter: The input parameter is a JSON object whose depth needs to be queried. Only the JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, and NVARCHAR2 types are supported. If other non-restricted types or an invalid JSON object is passed, an error will be reported during execution.

  Return type: int

  Example:

  ```
  -- The depth of an empty array is 1.
  gaussdb=# SELECT json_depth('[]');
   json_depth
  ------------
           1
  (1 row)

  gaussdb=# SELECT json_depth('{"s":1, "x":2,"y":[1]}');
   json_depth
  ------------
           3
  (1 row)
  ```

- json_extract(json, [VARIADIC text])

  Description: Extracts the data of JSON paths from a JSON object, merges this data into a JSON array, and returns the array. These JSON paths are built out of the variable-parameter list. Both the JSON paths and JSON object are specified in the input parameters. If **json** or **json_path** is null, **null** is returned. If the JSON format is incorrect, or **json_path** is an invalid path expression, an error is reported. If the variable parameter list contains null values and has format error, the exceptions are processed based on the exception sequence. If a null value exists first, **null** is returned first. If a format error occurs first, an error is reported first.

  Parameters: See **Table 1-83**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-83 Parameters

Parameter	Type	Description	Value Range
json	JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, NVARCHAR2	JSON value from which paths are to be extracted. If other non-restricted types are passed, an error will be reported during execution.	The value must be in JSON format.
[VARIADIC text]	Variable text array	Variable parameter list, containing one or more JSON paths.	**json_path** must be a valid path expression.

Return type: json

Example:

```
-- One path is extracted.
gaussdb=# SELECT json_extract('[1, 2, {"x": 3}]', '$[2]');
 json_extract
--------------
 {"x": 3}
(1 row)

-- Multiple paths are extracted.
gaussdb=# SELECT json_extract('["a", ["b", "c"], "d"]', '$[1]', '$[2]', '$[3]');
   json_extract
-------------------
 [["b", "c"], "d"]
(1 row)
```

- json_insert(json, [VARIADIC any])

  Description: Inserts a value at the position specified by **json_path** in a JSON object. The insertion operation is performed only when the path does not exist. The input parameter is a JSON object, several JSON paths, and the value to be inserted. Each JSON path must be paired with a value. If **json** or **json_path** is null, **null** is returned. In addition, **json_path** cannot contain wildcard characters * or **. Otherwise, an error is reported. If the variable parameter list contains null values and has format error, the exceptions are processed based on the exception sequence. If a null value exists first, **null** is returned first. If a format error occurs first, an error is reported first.

  Parameters: See **Table 1-84**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-84 Parameters

Parameter	Type	Description	Value Range
json	JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, NVARCHAR2	JSON value to be inserted. If other non-restricted types are passed, an error will be reported during execution.	The value must be in JSON format.
[VARIADIC any]	Variable any array	Variable parameter list, containing one or more json_path-value pairs.	**json_path** must be a valid path expression, and **value** can be any value.

Return type: json

Example:

```
gaussdb=# SELECT json_insert('[1, [2, 3], {"a": [4, 5]}]', '$[3]', 2);
          json_insert
-------------------------------
 [1, [2, 3], {"a": [4, 5]}, 2]
(1 row)

gaussdb=# SELECT json_insert('[1, [2, 3], {"a": [4, 5]}]', '$[10]', 10,'$[5]', 5);
            json_insert
-----------------------------------
 [1, [2, 3], {"a": [4, 5]}, 10, 5]
(1 row)
```

- json_keys(json[, json_path])

  Description: The input parameters are a JSON object and an optional JSON path. If no JSON path is specified, this function returns a JSON array of member key values of the top-layer object in the JSON object. If a JSON path is specified, this function returns the JSON array of the top-layer member key values in the JSON object corresponding to the path. If **json** or **json_path** is null, **null** is returned. If the JSON format is incorrect, **json_path** is an invalid path expression, or **json_path** contains * or **, an error is reported.

  Parameters: See **Table 1-85**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-85 Parameters

Parameter	Type	Description	Value Range
json	JSON, JSONB, CSTRING, TEXT, "char" , VARCHAR, BPCHAR, NVARCHAR2	A JSON value. If other non-restricted types are passed, an error will be reported during execution.	The value must be in JSON format.
json_path	text	A JSON path, which is optional.	**json_path** must be a valid path expression.

Return type: json

Example:

```
gaussdb=# SELECT json_keys('{"x": 1, "y": 2, "z": 3}');
    json_keys
-----------------
 ["x", "y", "z"]
(1 row)

gaussdb=# SELECT json_keys('[1,2,3,{"name":"Tom"}]','$[3]');
 json_keys
-----------
 ["name"]
(1 row)
```

- json_length(json[, json_path])

  Description: The input parameters are a JSON object and an optional JSON path. If no JSON path is specified, this function returns the length of the specified JSON object. If a JSON path is specified, this function returns the length of the JSON object corresponding to the path. If **json** or **json_path** is null, **null** is returned. If the JSON format is incorrect, **json_path** is an invalid path expression, or **json_path** contains * or **, an error is reported.

  Parameters: See **Table 1-86**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-86 Parameters

Parameter	Type	Description	Value Range
json	JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, NVARCHAR2	A JSON value. If other non-restricted types are passed, an error will be reported during execution.	The value must be in JSON format.
json_path	text	A JSON path, which is optional.	**json_path** must be a valid path expression.

Return type: int

Example:

```
gaussdb=# SELECT json_length('[1,2,3,4,5]');
 json_length
-------------
           5
(1 row)

gaussdb=# SELECT json_length('{"name":"Tom", "age":24, "like":"football"}');
 json_length
-------------
           3
(1 row)
```

- json_merge([VARIADIC any])

  Description: Combines all JSON objects passed to this function and returns the combined result. If the input parameter is null, **null** is returned. If the variable parameter list contains null values and has format error, the exceptions are processed based on the exception sequence. If a null value exists first, **null** is returned first. If a format error occurs first, an error is reported first.

  Parameter: a variable parameter list used to construct multiple JSON objects. The input parameter consists of multiple JSON objects. Only the JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, and NVARCHAR2 types are supported. If other non-restricted types are passed, an error will be reported during execution.

  Return type: json

  Example:

```
gaussdb=# SELECT json_merge('[1, 2]','[2]');
 json_merge
------------
 [1, 2, 2]
(1 row)

gaussdb=# SELECT json_merge('{"b":"2"}','{"a":"1"}','[1,2]');
          json_merge
------------------------------
 [{"a": "1", "b": "2"}, 1, 2]
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- json_quote(json)

  Description: Uses double quotation marks ("") to enclose the input parameter into a JSON string and returns the string.

  Parameter: The input parameter is a JSON object. Only the JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, and NVARCHAR2 types are supported. If other non-restricted types are passed, an error will be reported during execution.

  Return type: json

  Example:

  ```
  gaussdb=# SELECT json_quote('gauss');
   json_quote
  ------------
   "gauss"
  (1 row)
  ```

- json_unquote(json)

  Description: Cancels the enclosure of the input parameter using quotation marks and returns the character string.

  Parameter: The input parameter is a JSON object. Only the JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, and NVARCHAR2 types are supported. If other non-restricted types are passed, an error will be reported during execution.

  Return type: text

  Example:

  ```
  gaussdb=# SELECT json_unquote('"gauss"');
   json_unquote
  --------------
   gauss
  (1 row)
  ```

- json_remove(json, [VARIADIC text])

  Description: The input parameters are a JSON object and several JSON paths to be deleted. These JSON paths are constructed from the variable parameter list. This function deletes the value of **json_path** in the JSON object and returns the modified JSON object. If **json** or **json_path** is null, **null** is returned. If the JSON format is incorrect, **json_path** is an invalid path expression, or **json_path** contains * or **, an error is reported. If the variable parameter list contains null values and has format error, the exceptions are processed based on the exception sequence. If a null value exists first, **null** is returned first. If a format error occurs first, an error is reported first.

  Parameters: See **Table 1-87**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-87 Parameters

Parameter	Type	Description	Value Range
json	JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, NVARCHAR2	A JSON value. If other non-restricted types are passed, an error will be reported during execution.	The value must be in JSON format.
[VARIADIC text]	Variable text array	Variable parameter list, containing one or more JSON paths.	**json_path** must be a valid path expression.

Return type: json

Example:

```
gaussdb=# SELECT json_remove('[0, 1, 2, [3, 4]]', '$[0]', '$[2]');
 json_remove
-------------
 [1, 2]
(1 row)

gaussdb=# SELECT json_remove('[0, 1, 2, [3, 4]]', '$[0]', '$[0]','$[0]');
 json_remove
-------------
 [[3, 4]]
(1 row)
```

- json_replace(json, [VARIADIC any])

  Description: Replaces data in a path specified by **json_path** in a JSON object with a specified value and returns the modified JSON object. The input parameter is a JSON object, multiple JSON paths, and values to be replaced. The JSON paths and values must be in pairs. If **json** or **json_path** is null, **null** is returned. If the JSON format is incorrect, **json_path** is an invalid path expression, or **json_path** contains * or **, an error is reported. If the variable parameter list contains null values and has format error, the exceptions are processed based on the exception sequence. If a null value exists first, **null** is returned first. If a format error occurs first, an error is reported first.

  Parameters: See **Table 1-88**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-88 Parameters

Parameter	Type	Description	Value Range
json	JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, NVARCHAR2	A JSON value. If other non-restricted types are passed, an error will be reported during execution.	The value must be in JSON format.
[VARIADIC any]	Variable any input parameter	Variable parameter list, containing one or more json_path-value pairs.	**json_path** must be a valid path expression, and **value** can be any value.

Return type: json

Example:

```
gaussdb=# SELECT json_replace('{"x": 1}', '$.x', 'true');
 json_replace
---------------
 {"x": "true"}
(1 row)

gaussdb=# SELECT json_replace('{"x": 1}', '$.x', true, '$.x', 123, '$.x', 'asd', '$.x', null);
 json_replace
--------------
 {"x": null}
(1 row)
```

- json_search(json, mode_str, search_str, escape_char, json_path)

  Description: Returns the path of a given string in a JSON object. It returns a path string or an array of multiple paths. If any of **json**, **search_str**, and **json_path** is null, **null** is returned. If the JSON format is incorrect or **json_path** is not a valid path expression, an error is reported. If the variable parameter list contains null values and has format error, the exceptions are processed based on the exception sequence. If a null value exists first, **null** is returned first. If a format error occurs first, an error is reported first.

  Parameters: See **Table 1-89**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-89 Parameters

Parameter	Type	Description	Value Range
json	JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, NVARCHAR2	Target JSON object. If other non-restricted types are passed, an error will be reported during execution.	The value must be in JSON format.
mode_str	text	The value can be **one** or **all**. If the value is **one**, only the first matched path is obtained. If the value is **all**, all matched paths are obtained.	The value must be **one** or **all**, which is case-insensitive.
search_str	text	Character string to be searched for. In normal cases, the entire character string is matched. However, the wildcard % can be used to match any number of characters, and _ can be used to match any character.	-
escape_char	text	If **search_str** contains a wildcard, this parameter is defined as an escape character for the wildcard. This parameter is optional. The default value is **'\'**. If the character is added before the wildcard, the wildcard is not considered as a wildcard.	The value can be any single character.
json_path	text	If **json_path** is specified, the search is performed in the path. This parameter is optional.	**json_path** must be a valid path expression.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: text

Example:

```
-- In the all mode:
gaussdb=# SELECT json_search('{"a":"abc","b":"abc"}','all','abc');
 json_search
----------------
 ["$.a", "$.b"]
(1 row)

-- In the one mode:
gaussdb=# SELECT json_search('{"a":"abc","b":"abc"}','one','abc');
 json_search
-------------
 "$.a"
(1 row)

-- The default escape character is used.
gaussdb=# SELECT json_search('{"a":"abc","b":"a%c"}','one','a\%c');
 json_search
-------------
 "$.b"
(1 row)
```

- json_set(json, [VARIADIC any])

  Description: Uses the specified value to update the data in the path specified by **json_path** in a JSON object and returns the modified JSON object. If the specified path does not have any value, the value is inserted into the corresponding path. The input parameter is a JSON object, multiple JSON paths, and values to be set. The JSON paths and values must be in pairs. If **json** or **json_path** is null, **null** is returned. If the JSON format is incorrect, **json_path** is an invalid path expression, or **json_path** contains * or **, an error is reported. If the variable parameter list contains null values and has format error, the exceptions are processed based on the exception sequence. If a null value exists first, **null** is returned first. If a format error occurs first, an error is reported first.

  Parameters: See **Table 1-90**.

Table 1-90 Parameters

Parameter	Type	Description	Value Range
json	JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, NVARCHAR2	A JSON value. If other non-restricted types are passed, an error will be reported during execution.	The value must be in JSON format.
[VARIADIC any]	Variable any array	Variable parameter list, containing one or more json_path-value pairs.	**json_path** must be a valid path expression, and **value** can be any value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: json

Example:

```
gaussdb=# SELECT json_set('{"s":3}','$.s','d');
  json_set
------------
 {"s": "d"}
(1 row)

gaussdb=# SELECT json_set('{"s":3}','$.a','d','$.a','1');
      json_set
--------------------
 {"a": "1", "s": 3}
(1 row)
```

- json_type(json)

  Description: Returns a string representing the type of the given JSON object. If the input parameter is null, **null** is returned.

  Parameter: a JSON object. Only the JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, and NVARCHAR2 types are supported. If other non-restricted types are passed, an error will be reported during execution.

  Return type: text. For details, see **Table 1-91**.

  Table 1-91 Return value of json_type

JSON Element Type	Return Value
Array	ARRAY
Object	OBJECT
String	STRING
Numeric	NUMBER
Boolean	BOOLEAN
NULL	NULL

  Example:

```
gaussdb=# SELECT json_type('{"w":{"2":3},"2":4}');
 json_type
-----------
 OBJECT
(1 row)

gaussdb=# SELECT json_type('[1,2,2,3,3,4,4,4,4,4,4,4,4]');
 json_type
-----------
 ARRAY
(1 row)
```

- json_valid(json)

  Description: Returns **0** or **1** to indicate whether the given parameter is a valid JSON object. If the input parameter is null, **null** is returned.

  Parameter: The input parameter is a JSON object. Only the JSON, JSONB, CSTRING, TEXT, "char", VARCHAR, BPCHAR, and NVARCHAR2 types are supported. If a non-null value of a restricted type is passed, **0** is always returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: bigint

Example:

```
gaussdb=# SELECT json_valid('{"name":"Tom"}');
 json_valid
------------
          1
(1 row)


gaussdb=# SELECT json_valid('[1,23,4,5,5]');
 json_valid
------------
          1
(1 row)

gaussdb=# SELECT json_valid('[1,23,4,5,5]}');
 json_valid
------------
          0
(1 row)
```

- json_agg(any)

  Description: Aggregates values into a JSON array.

  Return type: array-json

  Example:

```
gaussdb=# CREATE TABLE classes(name varchar, score int);
CREATE TABLE
gaussdb=# INSERT INTO classes VALUES('A',2);
INSERT 0 1
gaussdb=# INSERT INTO classes VALUES('A',3);
INSERT 0 1
gaussdb=# INSERT INTO classes VALUES('D',5);
INSERT 0 1
gaussdb=# INSERT INTO classes VALUES('D',null);
INSERT 0 1
gaussdb=# SELECT * FROM classes;
 name | score
------+-------
 A    |     2
 A    |     3
 D    |     5
 D    |
(4 rows)

gaussdb=# SELECT name, json_agg(score) score FROM classes GROUP BY name ORDER BY name;
 name |   score
------+-----------
 A    | [2, 3]
 D    | [5, null]
(2 rows)

gaussdb=# DROP TABLE classes;
DROP TABLE
```

- json_object_agg(any, any)

  Description: Aggregates values into a JSON object.

  Return type: object-json

  Example:

```
gaussdb=# CREATE TABLE classes(name varchar, score int);
CREATE TABLE
gaussdb=# INSERT INTO classes VALUES('A',2);
INSERT 0 1
gaussdb=# INSERT INTO classes VALUES('A',3);
INSERT 0 1
gaussdb=# INSERT INTO classes VALUES('D',5);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
INSERT 0 1
gaussdb=# INSERT INTO classes VALUES('D',null);
INSERT 0 1
gaussdb=# SELECT * FROM classes;
 name | score
------+-------
 A    |     2
 A    |     3
 D    |     5
 D    |
(4 rows)

gaussdb=# SELECT json_object_agg(name, score) FROM classes GROUP BY name ORDER BY name;
     json_object_agg
-------------------------
 { "A" : 2, "A" : 3 }
 { "D" : 5, "D" : null }
(2 rows)

gaussdb=# DROP TABLE classes;
DROP TABLE
```

- jsonb_contained(jsonb, jsonb)

  Description: Same as the operator `<@`, determines whether all elements in the first parameter exist at the top layer of the second parameter.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT jsonb_contained('[1,2,3]', '[1,2,3,4]');
   jsonb_contained
  -----------------
    t
   (1 row)
  ```

- jsonb_contains(jsonb, jsonb)

  Description: Same as the operator `@>`, determines whether all elements in the second parameter exist at the top layer of the first parameter.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT jsonb_contains('[1,2,3,4]', '[1,2,3]');
   jsonb_contains
  ----------------
   t
  (1 row)
  ```

- jsonb_exists(jsonb, text)

  Description: Same as the operator `?`, checks whether the second parameter in the string exists at the top layer of the first parameter in the format of key/elem/scalar.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT jsonb_exists('["1",2,3]', '1');
   jsonb_exists
  --------------
   t
  (1 row)
  ```

- jsonb_exists_all(jsonb, text[])

  Description: Same as the operator `?&`, checks whether all elements of the second parameter in the string array exist at the top layer of the first parameter in the format of key/elem/scalar.

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT jsonb_exists_all('["1","2",3]', '{1, 2}');
 jsonb_exists_all
------------------
 t
(1 row)
```

- jsonb_exists_any(jsonb, text[])

  Description: Similar to the `?|` operator, it determines whether any element in parameter 2 exists at the top level of parameter 1 in the form of key/elem/scalar.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT jsonb_exists_any('["1","2",3]', '{1, 2, 4}');
 jsonb_exists_any
------------------
 t
(1 row)
```

- jsonb_cmp(jsonb, jsonb)

  Description: Compares values. A positive value indicates greater than, a negative value indicates less than, and **0** indicates equal.

  Return type: integer

  Example:

```
gaussdb=# SELECT jsonb_cmp('["a", "b"]', '{"a":1, "b":2}');
 jsonb_cmp
-----------
     -1
(1 row)
```

- jsonb_eq(jsonb, jsonb)

  Description: Same as the operator `=`, compares two values to check whether they are equal.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT jsonb_eq('["a", "b"]', '{"a":1, "b":2}');
 jsonb_eq
----------
 f
(1 row)
```

- jsonb_ne(jsonb, jsonb)

  Description: Same as the operator `<>`, compares two values to check whether they are not equal.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT jsonb_ne('["a", "b"]', '{"a":1, "b":2}');
 jsonb_ne
----------
 t
(1 row)
```

- jsonb_gt(jsonb, jsonb)

  Description: Same as the operator `>`, compares whether the value of the first parameter is greater than that of the second parameter.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT jsonb_gt('["a", "b"]', '{"a":1, "b":2}');
 jsonb_gt
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
----------
 f
(1 row)
```

- jsonb_ge(jsonb, jsonb)

  Description: Same as the operator `>=`, compares whether the value of the first parameter is greater than or equal to that of the second parameter.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT jsonb_ge('["a", "b"]', '{"a":1, "b":2}');
 jsonb_ge
----------
 f
(1 row)
```

- jsonb_lt(jsonb, jsonb)

  Description: Same as the operator `<`, compares whether the value of the first parameter is less than that of the second parameter.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT jsonb_lt('["a", "b"]', '{"a":1, "b":2}');
 jsonb_lt
----------
 t
(1 row)
```

- jsonb_le(jsonb, jsonb)

  Description: Same as the operator `<=`, compares whether the value of the first parameter is less than or equal to that of the second parameter.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT jsonb_le('["a", "b"]', '{"a":1, "b":2}');
 jsonb_le
----------
 t
(1 row)
```

- to_json(anyelement)

  Description: Converts the input parameter into the `json` type.

  Return type: json

  Example:

```
gaussdb=# SELECT to_json('{1,5}'::text[]);
  to_json
-----------
 ["1","5"]
(1 row)
```

- to_jsonb(anyelement)

  Description: Converts the input anyelement parameter to the JSONB type.

  Return type: jsonb

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- When the parameter is an empty string (''), if the SQL compatibility mode of the database is Oracle-compatible, **NULL** is returned; if the SQL compatibility mode of the database is PG-compatible, an empty string is returned. This is because empty strings are treated as NULL in ORA-compatible mode.
- When the parameter is of the DATE type, if the database SQL compatibility mode is Oracle-compatible, the returned result contains hour, minute, and second; if the database SQL compatibility mode is PG-compatible, the returned result does not contain hour, minute, and second. This is because DATE will be replaced by TIMESTAMP(0) WITHOUT TIME ZONE in ORA-compatible mode when **mapping_date_to_datea** is disabled.

Example:

```
gaussdb=# SELECT to_jsonb(ARRAY[1, 2, 3, 4]);
   to_jsonb
--------------
 [1, 2, 3, 4]
(1 row)
```

- jsonb_hash(jsonb)

  Description: Performs the hash operation on JSONB.

  Return type: integer

  Example:

```
gaussdb=# SELECT jsonb_hash('[1,2,3]');
 jsonb_hash
------------
 -559968547
(1 row)
```

- Other Functions

  Description: Internal functions used by JSON and JSONB aggregate functions.

```
json_agg_transfn
json_agg_finalfn
json_object_agg_transfn
json_object_agg_finalfn
```

1.6.14 HLL Functions and Operators

Hash Functions

- hll_hash_boolean(bool)

  Description: Hashes data of the Boolean type.

  Return type: hll_hashval

  Example:

```
gaussdb=# SELECT hll_hash_boolean(FALSE);
  hll_hash_boolean
---------------------
 -5451962507482445012
(1 row)
```

- hll_hash_boolean(bool, int32)

  Description: Configures a hash seed (that is, changes the hash policy) and hashes data of the Boolean type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: hll_hashval

Example:

```
gaussdb=# SELECT hll_hash_boolean(FALSE, 10);
 hll_hash_boolean
--------------------
 -1169037589280886076
(1 row)
```

- hll_hash_smallint(smallint)

  Description: Hashes data of the smallint type.

  Return type: hll_hashval

  Example:

  ```
  gaussdb=# SELECT hll_hash_smallint(100::smallint);
   hll_hash_smallint
  --------------------
   962727970174027904
  (1 row)
  ```

  NOTE

  If parameters with the same numeric value are hashed using different data types, the data will differ, because hash functions select different calculation policies for each type.

- hll_hash_smallint(smallint, int32)

  Description: Configures a hash seed (that is, changes the hash policy) and hashes data of the smallint type.

  Return type: hll_hashval

  Example:

  ```
  gaussdb=# SELECT hll_hash_smallint(100::smallint, 10);
   hll_hash_smallint
  --------------------
   -9056177146160443041
  (1 row)
  ```

- hll_hash_integer(integer)

  Description: Hashes data of the integer type.

  Return type: hll_hashval

  Example:

  ```
  gaussdb=# SELECT hll_hash_integer(0);
    hll_hash_integer
  ----------------------
   5156626420896634997
  (1 row)
  ```

- hll_hash_integer(integer, int32)

  Description: Hashes data of the integer type and configures a hash seed (that is, changes the hash policy).

  Return type: hll_hashval

  Example:

  ```
  gaussdb=# SELECT hll_hash_integer(0, 10);
   hll_hash_integer
  --------------------
   -5035020264353794276
  (1 row)
  ```

- hll_hash_bigint(bigint)

  Description: Hashes data of the bigint type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: hll_hashval

Example:

```
gaussdb=# SELECT hll_hash_bigint(100::bigint);
   hll_hash_bigint
---------------------
 -2401963681423227794
(1 row)
```

- hll_hash_bigint(bigint, int32)

  Description: Hashes data of the bigint type and configures a hash seed (that is, changes the hash policy).

  Return type: hll_hashval

  Example:

  ```
  gaussdb=# SELECT hll_hash_bigint(100::bigint, 10);
     hll_hash_bigint
  ---------------------
   -2305749404374433531
  (1 row)
  ```

- hll_hash_bytea(bytea)

  Description: Hashes data of the bytea type.

  Return type: hll_hashval

  Example:

  ```
  gaussdb=# SELECT hll_hash_bytea(E'\\x');
   hll_hash_bytea
  ----------------
   0
  (1 row)
  ```

- hll_hash_bytea(bytea, int32)

  Description: Hashes data of the bytea type and configures a hash seed (that is, changes the hash policy).

  Return type: hll_hashval

  Example:

  ```
  gaussdb=# SELECT hll_hash_bytea(E'\\x', 10);
     hll_hash_bytea
  ---------------------
   7233188113542599437
  (1 row)
  ```

- hll_hash_text(text)

  Description: Hashes data of the text type.

  Return type: hll_hashval

  Example:

  ```
  gaussdb=# SELECT hll_hash_text('AB');
     hll_hash_text
  ---------------------
   -5666002586880275174
  (1 row)
  ```

- hll_hash_text(text, int32)

  Description: Hashes data of the text type and configures a hash seed (that is, changes the hash policy).

  Return type: hll_hashval

  Example:

  ```
  gaussdb=# SELECT hll_hash_text('AB', 10);
  hll_hash_text
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
---------------------
-2215507121143724132
(1 row)
```

- hll_hash_any(anytype)

  Description: Hashes data of any type.

  Return type: hll_hashval

  Example:

```
gaussdb=# SELECT hll_hash_any(1);
    hll_hash_any
----------------------
 -1316670585935156930
(1 row)

gaussdb=# SELECT hll_hash_any('08:00:2b:01:02:03'::macaddr);
    hll_hash_any
----------------------
 -3719950434455589360
(1 row)
```

- hll_hash_any(anytype, int32)

  Description: Hashes data of any type and configures a hash seed (that is, changes the hash policy).

  Return type: hll_hashval

  Example:

```
gaussdb=# SELECT hll_hash_any(1, 10);
    hll_hash_any
----------------------
 7048553517657992351
(1 row)
```

- hll_hashval_eq(hll_hashval, hll_hashval)

  Description: Compares two pieces of data of the hll_hashval type to check whether they are the same.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT hll_hashval_eq(hll_hash_integer(1), hll_hash_integer(1));
 hll_hashval_eq
----------------
 t
(1 row)
```

- hll_hashval_ne(hll_hashval, hll_hashval)

  Description: Compares two pieces of data of the hll_hashval type to check whether they are different.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT hll_hashval_ne(hll_hash_integer(1), hll_hash_integer(1));
 hll_hashval_ne
----------------
 f
(1 row)
```

Log Functions

There are three HLL modes: explicit, sparse, and full. When the data size is small, the explicit mode is used. In this mode, distinct values are calculated without errors. As the number of distinct values increases, the HLL mode is switched to the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

sparse and full modes in sequence. The two modes have no difference in the calculation result, but vary in the calculation efficiency of HLL functions and the storage space of HLL objects. The following functions can be used to view some HLL parameters.

- hll_print(hll)

  Description: Prints some debugging parameters of an HLL.

  Example:

```
gaussdb=# SELECT hll_print(hll_empty());
                                hll_print
-------------------------------------------------------------------------------
 type=1(HLL_EMPTY), log2m=14, log2explicit=10, log2sparse=12, duplicatecheck=0
(1 row)
```

- hll_type(hll)

  Description: Checks the type of the current HLL.

  The return values are described as follows:

  – **0** indicates HLL_UNINIT, which is an uninitialized HLL object.
  – **1** indicates HLL_EMPTY, which is an empty HLL object.
  – **2** indicates HLL_EXPLICIT, which is an HLL object in explicit mode.
  – **3** indicates HLL_SPARSE, which is an HLL object in sparse mode.
  – **4** indicates HLL_FULL, which is an HLL object in full mode.
  – **5** indicates HLL_UNDEFINED, which is an invalid HLL object.

  Example:

```
gaussdb=# SELECT hll_type(hll_empty());
 hll_type
----------
        1
(1 row)
```

- hll_log2m(hll)

  Description: Checks the value of **log2m** in the current HLL data structure. **log2m** is the logarithm of the number of buckets. This value affects the error rate of calculating distinct values by HLL. The error rate = ±1.04/√(2^log2m). When the value of **log2m** is explicitly set to a value ranging from 10 to 16, HLL sets the number of buckets to 2^{log2m}. When the value of **log2explicit** is explicitly set to **–1**, the built-in default value is used.

  Example:

```
gaussdb=# SELECT hll_log2m(hll_empty());
 hll_log2m
-----------
        14
(1 row)

gaussdb=# SELECT hll_log2m(hll_empty(10));
 hll_log2m
-----------
        10
(1 row)

gaussdb=# SELECT hll_log2m(hll_empty(-1));
 hll_log2m
-----------
        14
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- hll_log2explicit(hll)

  Description: Queries the value of **log2explicit** in the current HLL data structure. Generally, the HLL changes from the explicit mode to the sparse mode and then to the full mode. This process is called the promotion hierarchy policy. You can change the value of **log2explicit** to change the policy. For example, if the value of **log2explicit** is **0**, the explicit mode will be skipped and the sparse mode is directly entered. When the value of **log2explicit** is explicitly set to a value ranging from 1 to 12, the HLL will switch to the sparse mode when the length of the data segment exceeds $2^{log2explicit}$. When the value of **log2explicit** is explicitly set to **–1**, the built-in default value is used.

  Example:

```
gaussdb=# SELECT hll_log2explicit(hll_empty());
 hll_log2explicit
------------------
               10
(1 row)

gaussdb=# SELECT hll_log2explicit(hll_empty(12, 8));
 hll_log2explicit
------------------
                8
(1 row)

gaussdb=# SELECT hll_log2explicit(hll_empty(12, -1));
 hll_log2explicit
------------------
               10
(1 row)
```

- hll_log2sparse(hll)

  Description: Queries the value of **log2sparse** in the current HLL data structure. Generally, the HLL changes from the explicit mode to the sparse mode and then to the full mode. This process is called the promotion hierarchy policy. You can adjust the value of **log2sparse** to change the policy. For example, if the value of **log2sparse** is **0**, the system skips the sparse mode and directly enters the full mode. If the value of **log2sparse** is explicitly set to a value ranging from 1 to 14, the HLL will switch to the full mode when the length of the data segment exceeds $2^{log2sparse}$. When the value of **log2sparse** is explicitly set to **–1**, the built-in default value is used.

  Example:

```
gaussdb=# SELECT hll_log2sparse(hll_empty());
 hll_log2sparse
----------------
             12
(1 row)

gaussdb=# SELECT hll_log2sparse(hll_empty(12, 8, 10));
 hll_log2sparse
----------------
             10
(1 row)

gaussdb=# SELECT hll_log2sparse(hll_empty(12, 8, -1));
 hll_log2sparse
----------------
             12
(1 row)
```

- hll_duplicatecheck(hll)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Specifies whether duplicate check is enabled. **0**: disable; **1**: enable. This function is disabled by default. If there are many duplicate values, you can enable this function to improve efficiency. When the value of **duplicatecheck** is explicitly set to **–1**, the built-in default value is used.

Example:

```
gaussdb=# SELECT hll_duplicatecheck(hll_empty());
 hll_duplicatecheck
--------------------
                  0
(1 row)

gaussdb=# SELECT hll_duplicatecheck(hll_empty(12, 8, 10, 1));
 hll_duplicatecheck
--------------------
                  1
(1 row)

gaussdb=# SELECT hll_duplicatecheck(hll_empty(12, 8, 10, -1));
 hll_duplicatecheck
--------------------
                  0
(1 row)
```

Functional Functions

- hll_empty()

  Description: Creates an empty HLL.

  Return type: HLL

  Example:

  ```
  gaussdb=# SELECT hll_empty();
                                                          hll_empty
  ------------------------------------------------------------------
   \x484c4c00000000002b050000000000000000000000000000000000000000
  (1 row)
  ```

- hll_empty(int32 log2m)

  Description: Creates an empty HLL and sets the **log2m** parameter. The parameter value ranges from 10 to 16. If the input is **–1**, the built-in default value is used.

  Return type: HLL

  Example:

  ```
  gaussdb=# SELECT hll_empty(10);
                                                          hll_empty
  ------------------------------------------------------------------
   \x484c4c00000000002b040000000000000000000000000000000000000000
  (1 row)

  gaussdb=# SELECT hll_empty(-1);
                                                          hll_empty
  ------------------------------------------------------------------
   \x484c4c00000000002b050000000000000000000000000000000000000000
  (1 row)
  ```

- hll_empty(int32 log2m, int32 log2explicit)

  Description: Creates an empty HLL and sets the **log2m** and **log2explicit** parameters in sequence. The value of **log2explicit** ranges from 0 to 12. The value **0** indicates that the explicit mode is skipped. This parameter is used to set the threshold of the explicit mode. When the length of the data segment reaches $2^{log2explicit}$, the mode is switched to the sparse or full mode. If the input is **–1**, the built-in default value of **log2explicit** is used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: HLL

Example:

```
gaussdb=# SELECT hll_empty(10, 4);
                    hll_empty
--------------------------------------------------------------
 \x484c4c0000000000130400000000000000000000000000000000000000
(1 row)

gaussdb=# SELECT hll_empty(10, -1);
                    hll_empty
--------------------------------------------------------------
 \x484c4c00000000002b040000000000000000000000000000000000000000
(1 row)
```

- hll_empty(int32 log2m, int32 log2explicit, int64 log2sparse)

  Description: Creates an empty HLL and sets the **log2m**, **log2explicit** and **log2sparse** parameters in sequence. The value of **log2sparse** ranges from 0 to 14. The value **0** indicates that the sparse mode is skipped. This parameter is used to set the threshold of the sparse mode. When the length of the data segment reaches $2^{log2sparse}$, the mode is switched to the full mode. If the input is **–1**, the built-in default value of **log2sparse** is used.

  Return type: HLL

  Example:

  ```
  gaussdb=# SELECT hll_empty(10, 4, 8);
                      hll_empty
  --------------------------------------------------------------
   \x484c4c0000000000120400000000000000000000000000000000000000
  (1 row)

  gaussdb=# SELECT hll_empty(10, 4, -1);
                      hll_empty
  --------------------------------------------------------------
   \x484c4c0000000000130400000000000000000000000000000000000000
  (1 row)
  ```

- hll_empty(int32 log2m, int32 log2explicit, int64 log2sparse, int32 duplicatecheck)

  Description: Creates an empty HLL and sets the **log2m**, **log2explicit**, **log2sparse**, and **duplicatecheck** parameters in sequence. The value of **duplicatecheck** is **0** or **1**, indicating whether the duplicate check mode is enabled. By default, this mode is disabled. If the input is **–1**, the built-in default value of **duplicatecheck** is used.

  Return type: HLL

  Example:

  ```
  gaussdb=# SELECT hll_empty(10, 4, 8, 0);
                      hll_empty
  --------------------------------------------------------------
   \x484c4c0000000000120400000000000000000000000000000000000000
  (1 row)

  gaussdb=# SELECT hll_empty(10, 4, 8, -1);
                      hll_empty
  --------------------------------------------------------------
   \x484c4c0000000000120400000000000000000000000000000000000000
  (1 row)
  ```

- hll_add(hll, hll_hashval)

  Description: Adds hll_hashval to an HLL.

  Return type: HLL

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT hll_add(hll_empty(), hll_hash_integer(1));
                                hll_add
-------------------------------------------------------------------------------
 \x484c4c08000002002b0900000000000000f03f3e2921ff133fbaed3e2921ff133fbaed00
(1 row)
```

- hll_add_rev(hll_hashval, hll)

  Description: Adds **hll_hashval** to an HLL. This function works the same as **hll_add**, except that the positions of parameters are switched.

  Return type: HLL

  Example:

```
gaussdb=# SELECT hll_add_rev(hll_hash_integer(1), hll_empty());
                               hll_add_rev
-------------------------------------------------------------------------------
 \x484c4c08000002002b0900000000000000f03f3e2921ff133fbaed3e2921ff133fbaed00
(1 row)
```

- hll_eq(hll, hll)

  Description: Compares two HLLs to check whether they are the same.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT hll_eq(hll_add(hll_empty(), hll_hash_integer(1)), hll_add(hll_empty(),
hll_hash_integer(2)));
 hll_eq
--------
 f
(1 row)
```

- hll_ne(hll, hll)

  Description: Compares two HLLs to check whether they are different.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT hll_ne(hll_add(hll_empty(), hll_hash_integer(1)), hll_add(hll_empty(),
hll_hash_integer(2)));
 hll_ne
--------
 t
(1 row)
```

- hll_cardinality(hll)

  Description: Calculates the number of distinct values of an HLL.

  Return type: int

  Example:

```
gaussdb=# SELECT hll_cardinality(hll_empty() || hll_hash_integer(1));
 hll_cardinality
-----------------
               1
(1 row)
```

- hll_union(hll, hll)

  Description: Performs the UNION operation on two HLL data structures to obtain one HLL.

  Return type: HLL

  Example:

```
gaussdb=# SELECT hll_union(hll_add(hll_empty(), hll_hash_integer(1)), hll_add(hll_empty(),
hll_hash_integer(2)));
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
                                    hll_union
-----------------------------------------------------------------------------------------------

\x484c4c10002000002b090000000000000000040000000000000000b3ccc49320cca1ae3e2921ff133fba
ed00
(1 row)
```

Aggregate Functions

- hll_add_agg(hll_hashval)

  Description: Groups hashed data into HLL.

  Return type: HLL

  Example:

```
-- Prepare data:
gaussdb=# CREATE TABLE t_id(id int);
gaussdb=# INSERT INTO t_id VALUES(generate_series(1,500));
gaussdb=# CREATE TABLE t_data(a int, c text);
gaussdb=# INSERT INTO t_data SELECT mod(id,2), id FROM t_id;

-- Create a table and specify an HLL column:
gaussdb=# CREATE TABLE t_a_c_hll(a int, c hll);

-- Use GROUP BY on column a to group data, and insert the data to the HLL:
gaussdb=# INSERT INTO t_a_c_hll SELECT a, hll_add_agg(hll_hash_text(c)) FROM t_data GROUP BY a;

-- Calculate the number of distinct values for each group in the HLL:
gaussdb=# SELECT a, #c AS cardinality FROM t_a_c_hll ORDER BY a;
 a |   cardinality
---+------------------
 0 | 247.862354346299
 1 | 250.908710610377
(2 rows)
```

- hll_add_agg(hll_hashval, int32 log2m)

  Description: Groups hashed data into HLL and specifies the **log2m** parameter. The value ranges from 10 to 16. If the input is **–1** or **NULL**, the built-in default value is used.

  Return type: HLL

  Example:

```
gaussdb=# SELECT hll_cardinality(hll_add_agg(hll_hash_text(c), 12)) FROM t_data;
 hll_cardinality
------------------
 497.965240179228
(1 row)
```

- hll_add_agg(hll_hashval, int32 log2m, int32 log2explicit)

  Description: Groups hashed data into HLL and specifies the **log2m** and **log2explicit** parameters in sequence. The value of **log2explicit** ranges from 0 to 12. The value **0** indicates that the explicit mode is skipped. This parameter is used to set the threshold of the explicit mode. When the length of the data segment reaches $2^{log2explicit}$, the mode is switched to the sparse or full mode. If the input is **–1** or **NULL**, the built-in default value of **log2explicit** is used.

  Return type: HLL

  Example:

```
gaussdb=# SELECT hll_cardinality(hll_add_agg(hll_hash_text(c), NULL, 1)) FROM t_data;
 hll_cardinality
------------------
 498.496062953313
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- hll_add_agg(hll_hashval, int32 log2m, int32 log2explicit, int64 log2sparse)

  Description: Groups hashed data into HLL and sets the **log2m**, **log2explicit**, and **log2sparse** parameters in sequence. The value of **log2sparse** ranges from 0 to 14. The value **0** indicates that the sparse mode is skipped. This parameter is used to set the threshold of the sparse mode. When the length of the data segment reaches $2^{log2sparse}$, the mode is switched to the full mode. If the input is **–1** or **NULL**, the built-in default value of **log2sparse** is used.

  Return type: HLL

  Example:

  ```
  gaussdb=# SELECT hll_cardinality(hll_add_agg(hll_hash_text(c), NULL, 6, 10)) FROM t_data;
   hll_cardinality
  ------------------
   498.496062953313
  (1 row)
  ```

- hll_add_agg(hll_hashval, int32 log2m, int32 log2explicit, int64 log2sparse, int32 duplicatecheck)

  Description: Groups hashed data into HLL and specifies the **log2m**, **log2explicit**, **log2sparse**, and **duplicatecheck** parameters. The value of **duplicatecheck** can be **0** or **1**, indicating whether to enable this mode. By default, this mode is disabled. If the input is **–1** or **NULL**, the built-in default value of **duplicatecheck** is used.

  Return type: HLL

  Example:

  ```
  gaussdb=# SELECT hll_cardinality(hll_add_agg(hll_hash_text(c), NULL, 6, 10, -1)) FROM t_data;
   hll_cardinality
  ------------------
   498.496062953313
  (1 row)
  ```

- hll_union_agg(hll)

  Description: Performs the UNION operation on multiple pieces of data of the HLL type to obtain one HLL.

  Return type: HLL

  Example:

  ```
  -- Perform the UNION operation on data of the HLL type in each group to obtain one HLL, and calculate the number of distinct values:
  gaussdb=# SELECT #hll_union_agg(c) AS cardinality FROM t_a_c_hll;
    cardinality
  ------------------
   498.496062953313
  (1 row)

  -- Drop tables.
  gaussdb=# DROP TABLE t_id;
  gaussdb=# DROP TABLE t_data;
  gaussdb=# DROP TABLE t_a_c_hll;
  ```

  NOTE

  To perform the UNION operation on data in multiple HLLs, ensure that the HLLs have the same precision. Otherwise, **UNION** cannot be performed. This constraint also applies to the **hll_union(hll, hll)** function.

Obsolete Functions

Some old HLL functions are deprecated due to version upgrade. You can replace them with similar functions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- hll_schema_version(hll)

  Description: Checks the schema version in the current HLL. In earlier versions, the schema version is fixed at **1**, which is used to verify the header of the HLL field. After refactoring, the HLL field is added to the header for verification. The schema version is no longer used.

- hll_regwidth(hll)

  Description: Queries the bucket size in the HLL data structure. In earlier versions, the value of **regwidth** ranges from 1 to 5, which has a large error and limits the upper limit of the cardinality estimation. After refactoring, the value of **regwidth** is fixed at **6** and the variable is not used.

- hll_expthresh(hll)

  Description: Obtains the value of **expthresh** in the current HLL. The hll_log2explicit(hll) function is used to replace similar functions.

- hll_sparseon(hll)

  Description: Specifies whether the sparse mode is enabled. Use hll_log2sparse(hll) to replace similar functions. The value **0** indicates that the sparse mode is disabled.

Built-In Functions

HLL has a series of built-in functions for internal data processing. Generally, users do not need to know how to use these functions. For details, see **Table 1-92**.

Table 1-92 Built-in functions

Function	Description
hll_in	Receives HLL data in string format.
hll_out	Sends HLL data in string format.
hll_recv	Receives HLL data in bytea format.
hll_send	Sends HLL data in bytea format.
hll_trans_in	Receives hll_trans_type data in string format.
hll_trans_out	Sends hll_trans_type data in string format.
hll_trans_recv	Receives hll_trans_type data in bytea format.
hll_trans_send	Sends hll_trans_type data in bytea format.
hll_typmod_in	Receives typmod data.
hll_typmod_out	Sends typmod data.
hll_hashval_in	Receives hll_hashval data.
hll_hashval_out	Sends hll_hashval data.
hll_add_trans0	Works similar to hll_add. No input parameter is specified during initialization. It is usually used in the first phase of DNs in distributed aggregation operations.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Description
hll_add_trans1	Works similar to hll_add. An input parameter is specified during initialization. It is usually used in the first phase of DNs in distributed aggregation operations.
hll_add_trans2	Works similar to hll_add. Two input parameters are specified during initialization. It is usually used in the first phase of DNs in distributed aggregation operations.
hll_add_trans3	Works similar to hll_add. Three input parameters are specified during initialization. It is usually used in the first phase of DNs in distributed aggregation operations.
hll_add_trans4	Works similar to hll_add. Four input parameters are specified during initialization. It is usually used in the first phase of DNs in distributed aggregation operations.
hll_union_trans	Works similar to hll_union and is used on the first phase of DNs in distributed aggregation operations.
hll_union_collect	Works similar to hll_union and is used on the second phase of CNs in distributed aggregation operations to summarize the results of each DN.
hll_pack	Is used on the third phase of CNs in distributed aggregation operations to convert a user-defined type hll_trans_type to the HLL type.
hll	Converts an HLL type to another HLL type. Input parameters can be specified.
hll_hashval	Converts the bigint type to the hll_hashval type.
hll_hashval_int4	Converts the int4 type to the hll_hashval type.

Operators

- =

  Description: Compares the values of the HLL or hll_hashval type to check whether they are the same.

  Return type: Boolean

  Example:

```
--hll
gaussdb=# SELECT (hll_empty() || hll_hash_integer(1)) = (hll_empty() || hll_hash_integer(1));
 ?column?
----------
 t
(1 row)

--hll_hashval
gaussdb=# SELECT hll_hash_integer(1) = hll_hash_integer(1);
 ?column?
----------
 t
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- <> or !=

  Description: Compares the values of the HLL or hll_hashval type to check whether they are different.

  Return type: Boolean

  Example:

```
--hll
gaussdb=# SELECT (hll_empty() || hll_hash_integer(1)) <> (hll_empty() || hll_hash_integer(2));
 ?column?
----------
 t
(1 row)

--hll_hashval
gaussdb=# SELECT hll_hash_integer(1) <> hll_hash_integer(2);
 ?column?
----------
 t
(1 row)
```

- ||

  Description: Represents the functions of hll_add, hll_union, and hll_add_rev.

  Return type: HLL

  Example:

```
--hll_add
gaussdb=# SELECT hll_empty() || hll_hash_integer(1);
                                ?column?
--------------------------------------------------------------------------------
 \x484c4c08000002002b0900000000000000f03f3e2921ff133fbaed3e2921ff133fbaed00
(1 row)

--hll_add_rev
gaussdb=# SELECT hll_hash_integer(1) || hll_empty();
                                ?column?
--------------------------------------------------------------------------------
 \x484c4c08000002002b0900000000000000f03f3e2921ff133fbaed3e2921ff133fbaed00
(1 row)

--hll_union
gaussdb=# SELECT (hll_empty() || hll_hash_integer(1)) || (hll_empty() || hll_hash_integer(2));
                                    ?column?
-----------------------------------------------------------------------------------------------

\x484c4c10002000002b0900000000000000004000000000000000000b3ccc49320cca1ae3e2921ff133fba
ed00
(1 row)
```

- #

  Description: Calculates the number of distinct values of an HLL. It works the same as the hll_cardinality function.

  Return type: int

  Example:

```
gaussdb=# SELECT #(hll_empty() || hll_hash_integer(1));
 ?column?
----------
        1
(1 row)
```

1.6.15 SEQUENCE Functions

The sequence functions provide a simple method to ensure security of multiple users for users to obtain sequence values from sequence objects.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- nextval(regclass)

  Description: Specifies an increasing sequence and returns a new value.

  NOTE

  - To avoid blocking of concurrent transactions that obtain numbers from the same sequence, a nextval operation is never rolled back; that is, once a value is fetched, it is considered used, even if the transaction that did the nextval later aborts. This means that aborted transactions may leave unused "holes" in the sequence of assigned values. Therefore, the GaussDB sequence object cannot be used to obtain "holeless" sequences.
  - If the nextval function is pushed to DNs, each DN will automatically connect to the GTM and requests the next value. For example, in the **insert into t1 select** *xxx* statement, a column in table **t1** needs to call the nextval function. If the maximum number of connections on the GTM is 8192, this type of pushed statements occupies too many GTM connections. Therefore, the number of concurrent connections for these statements is limited to 7000 divided by the number of cluster DNs. The other connections are reserved for other statements.
  - If an insert statement contains both a set returning function (SRF) and an auto-increment sequence column, the sequence value will skip one increment. The following uses the generate_series function as an example:

    ```
    CREATE TABLE t1 (a int, b serial);
    INSERT INTO t1(a) VALUES(generate_series(1,1)); -- Insert data with column b as 1.
    INSERT INTO t1(a) VALUES(generate_series(1,1));  -- Insert data with column b as 3, skipping 2.
    ```

  - For the sequence of the **rowid** system column, nextval() can be operated only in maintenance mode.

  Return type: numeric

  The nextval function can be called in either of the following ways (In example 2, the ORA syntax is supported. Currently, the sequence name cannot contain a dot.):

  ```
  gaussdb=# CREATE SEQUENCE seqDemo;

  -- Example 1:
  gaussdb=# SELECT nextval('seqDemo');
   nextval
  ---------
        1
  (1 row)

  -- Example 2:
  gaussdb=# SELECT seqDemo.nextval;
   nextval
  ---------
        2
  (1 row)
  gaussdb=# DROP SEQUENCE seqDemo;
  ```

- currval(regclass)

  Description: Returns the last value of **nextval** in the current session. If nextval has not been called for the specified sequence in the current session, an error is reported when currval is called. By default, **currval** is disabled. To enable it, set **enable_beta_features** to **true**. After **enable_beta_features** is set to **true**, **nextval** will not be pushed down.

  Return type: numeric

  The currval function can be called in either of the following ways (In example 2, the ORA syntax is supported. Currently, the sequence name cannot contain a dot.):

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CREATE SEQUENCE seq1;
gaussdb=# SELECT nextval('seq1');
gaussdb=# SET enable_beta_features = true;

-- Example 1:
gaussdb=# SELECT currval('seq1');
 currval
---------
       1
(1 row)

-- Example 2:
gaussdb=# SELECT seq1.currval seq1;
 seq1
------
    1
(1 row)
gaussdb=# DROP SEQUENCE seq1;
gaussdb=# SET enable_beta_features = false;
```

- lastval()

  Description: Returns the last value of **nextval** in the current session. This function is equivalent to currval, except that it does not use sequence names as parameters. It fetches the sequence used by nextval last time in the current session. If nextval has not been called in the current session, an error is reported when lastval is called.

  By default, lastval is disabled. To enable it, set **enable_beta_features** or **lastval_supported** to **true**. After lastval is enabled, nextval() will not be pushed down.

  Return type: numeric

  Example:

  ```
  gaussdb=# CREATE SEQUENCE seq1;
  gaussdb=# SELECT nextval('seq1');
  gaussdb=# SET enable_beta_features = true;
  gaussdb=# SELECT lastval();
   lastval
  ---------
         1
  (1 row)
  gaussdb=# DROP SEQUENCE seq1;
  gaussdb=# SET enable_beta_features = false;
  ```

- setval(regclass, bigint)

  Description: Sets the current value of a sequence.

  Return type: numeric

  Example:

  ```
  gaussdb=# CREATE SEQUENCE seqDemo;
  gaussdb=# SELECT nextval('seqDemo');
  gaussdb=# SELECT setval('seqDemo',3);
   setval
  --------
       3
  (1 row)
  gaussdb=# DROP SEQUENCE seqDemo;
  ```

- setval(regclass, numeric, Boolean)

  Sets the current value of a sequence and the is_called sign.

  Return type: numeric

  Example:

  ```
  gaussdb=# CREATE SEQUENCE seqDemo;
  gaussdb=# SELECT nextval('seqDemo');
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT setval('seqDemo',5,true);
 setval
--------
      5
(1 row)
gaussdb=# DROP SEQUENCE seqDemo;
```

NOTE

- After this function is used, the new value takes effect immediately on the GTM. If the cache value is greater than the **session_sequence_cache** value, the current session continues to obtain the cache until the cache value is used up. If the cache value is less than or equal to the **session_sequence_cache** value, the new value takes effect immediately in the current session. However, if other sessions have cached sequence values, the new value takes effect only after the cached values are used up. To avoid sequence value conflicts, exercise caution when using this function.
- Changes made by this function will not be undone if a transaction is rolled back, as the sequence is non-transactional.
- For sequences of the **rowid** system column, this function can be used only in maintenance mode.

- pg_sequence_last_value(sequence_oid oid, OUT cache_value int16, OUT last_value int16)

  Description: Obtains the parameters of a specified sequence, including the cache value and current value.

  Return type: int16, int16

- gs_get_sequence_parameters(sequence_oid, OUT start_value int16, OUT minimum_value int16, OUT maximum_value int16, OUT increment int16, OUT cycle_option Boolean, OUT order_option Boolean)

  Description: Obtains the parameter information about a specified sequence, including the start value, minimum value, maximum value, increment, cycling option, and order option.

  Return type: int16, int16, int16, int16, boolean, boolean

- gs_get_sequence_last_value(sequence_oid oid, OUT cache_value int16, OUT last_value int16)

  Description: Obtains the parameters of a specified sequence, including the cache value and current value.

  Return type: int16, int16

- get_global_last_value_from_gtm(sequence_regclass regclass, OUT is_called boolean, OUT last_value numeric)

  Description: Obtains the flag indicating whether a sequence has been called and the latest value of the requested sequence from GTM.

  Parameter: name or OID of a sequence to be queried.

  Return types: Boolean and numeric

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- For upgrade from a version that does not support get_global_last_value_from_gtm to a version that supports get_global_last_value_from_gtm, this function cannot be used during the upgrade observation period.
- The get_global_last_value_from_gtm function requires communication with the GTM each time it is called; therefore, it is not recommended to execute it frequently.
- The latest value that has been requested for a sequence does not necessarily indicate the maximum value that has been used. If the sequence's cache is greater than 1 and the sequence values requested by the node are not used up, this will leave unused "holes."

1.6.16 Array Functions and Operators

Array Operators

- =

  Description: Specifies whether two arrays are equal.

  Example:

```
gaussdb=# SELECT ARRAY[1.1,2.1,3.1]::int[] = ARRAY[1,2,3] AS RESULT ;
 result
--------
 t
(1 row)
```

- <>

  Description: Specifies whether two arrays are not equal.

  Example:

```
gaussdb=# SELECT ARRAY[1,2,3] <> ARRAY[1,2,4] AS RESULT;
 result
--------
 t
(1 row)
```

- <

  Description: Specifies whether an array is less than another.

  Example:

```
gaussdb=# SELECT ARRAY[1,2,3] < ARRAY[1,2,4] AS RESULT;
 result
--------
 t
(1 row)
```

- >

  Description: Specifies whether an array is greater than another.

  Example:

```
gaussdb=# SELECT ARRAY[1,4,3] > ARRAY[1,2,4] AS RESULT;
 result
--------
 t
(1 row)
```

- <=

  Description: Specifies whether an array is less than or equal to another.

  Example:

```
gaussdb=# SELECT ARRAY[1,2,3] <= ARRAY[1,2,3] AS RESULT;
 result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------
 t
(1 row)
```

- >=

  Description: Specifies whether an array is greater than or equal to another.

  Example:

```
gaussdb=# SELECT ARRAY[1,4,3] >= ARRAY[1,4,3] AS RESULT;
 result
--------
 t
(1 row)
```

- @>

  Description: Specifies whether an array contains another.

  Example:

```
gaussdb=# SELECT ARRAY[1,4,3] @> ARRAY[3,1] AS RESULT;
 result
--------
 t
(1 row)
```

- <@

  Description: Specifies whether an array is contained in another.

  Example:

```
gaussdb=# SELECT ARRAY[2,7] <@ ARRAY[1,7,4,2,6] AS RESULT;
 result
--------
 t
(1 row)
```

- &&

  Description: Specifies whether an array overlaps another (have common elements).

  Example:

```
gaussdb=# SELECT ARRAY[1,4,3] && ARRAY[2,1] AS RESULT;
 result
--------
 t
(1 row)
```

- ||

  Description: Array-to-array concatenation.

  Example:

```
gaussdb=# SELECT ARRAY[1,2,3] || ARRAY[4,5,6] AS RESULT;
    result
---------------
 {1,2,3,4,5,6}
(1 row)
gaussdb=# SELECT ARRAY[1,2,3] || ARRAY[[4,5,6],[7,8,9]] AS RESULT;
          result
---------------------------
 {{1,2,3},{4,5,6},{7,8,9}}
(1 row)
```

- ||

  Description: Element-to-array concatenation.

  Example:

```
gaussdb=# SELECT 3 || ARRAY[4,5,6] AS RESULT;
  result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-----------
 {3,4,5,6}
(1 row)
```

- ||

  Description: Array-to-element concatenation.

  Example:

```
gaussdb=# SELECT ARRAY[4,5,6] || 7 AS RESULT;
  result
-----------
 {4,5,6,7}
(1 row)
```

Array comparisons compare the array contents element-by-element, using the default B-tree comparison function for the element data type. In multidimensional arrays, the elements are accessed in row-major order. If the contents of two arrays are equal but the dimensionality is different, the first difference in the dimensionality information determines the sort order.

Array Functions

- array_append(anyarray, anyelement)

  Description: Appends an element to the end of an array, and only supports one-dimensional arrays.

  Return type: anyarray

  Example:

```
gaussdb=# SELECT array_append(ARRAY[1,2], 3) AS RESULT;
 result
---------
 {1,2,3}
(1 row)
```

- array_prepend(anyelement, anyarray)

  Description: Appends an element to the beginning of an array, and only supports one-dimensional arrays.

  Return type: anyarray

  Example:

```
gaussdb=# SELECT array_prepend(1, ARRAY[2,3]) AS RESULT;
 result
---------
 {1,2,3}
(1 row)
```

- array_cat(anyarray, anyarray)

  Description: Concatenates two arrays, and supports multi-dimensional arrays.

  Return type: anyarray

  Example:

```
gaussdb=# SELECT array_cat(ARRAY[1,2,3], ARRAY[4,5]) AS RESULT;
   result
-------------
 {1,2,3,4,5}
(1 row)

gaussdb=# SELECT array_cat(ARRAY[[1,2],[4,5]], ARRAY[6,7]) AS RESULT;
      result
--------------------
 {{1,2},{4,5},{6,7}}
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- array_union(anyarray, anyarray)

  Description: Concatenates two arrays. Only one-dimensional arrays are supported. If an input parameter is **NULL**, another input parameter is returned.

  Return type: anyarray

  Example:

```
gaussdb=# SELECT array_union(ARRAY[1,2,3], ARRAY[3,4,5]) AS RESULT;
   result
-------------
 {1,2,3,3,4,5}
(1 row)

gaussdb=# SELECT array_union(ARRAY[1,2,3], NULL) AS RESULT;
 result
---------
 {1,2,3}
(1 row)
```

- array_union_distinct(anyarray, anyarray)

  Description: Concatenates two arrays and deduplicates them. Only one-dimensional arrays are supported. If an input parameter is **NULL**, the other input parameter is returned. Make sure that the returned input parameter contains only one-dimensional arrays. For multi-dimensional arrays, an error is reported.

  Return type: anyarray

  Example:

```
gaussdb=# SELECT array_union_distinct(ARRAY[1,2,3], ARRAY[3,4,5]) AS RESULT;
   result
-------------
 {1,2,3,4,5}
(1 row)

gaussdb=# SELECT array_union_distinct(ARRAY[1,2,3], NULL) AS RESULT;
 result
---------
 {1,2,3}
(1 row)
```

- array_intersect(anyarray, anyarray)

  Description: Intersects two arrays. Only one-dimensional arrays are supported. If any input parameter is **NULL**, **NULL** is returned.

  Return type: anyarray

  Example:

```
gaussdb=# SELECT array_intersect(ARRAY[1,2,3], ARRAY[3,4,5]) AS RESULT;
   result
-------------
 {3}
(1 row)

gaussdb=# SELECT array_intersect(ARRAY[1,2,3], NULL) AS RESULT;
 result
--------

(1 row)
```

- array_intersect_distinct(anyarray, anyarray)

  Description: Intersects two arrays and deduplicates them. Only one-dimensional arrays are supported. If any input parameter is **NULL**, **NULL** is returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: anyarray

Example:

```
gaussdb=# SELECT array_intersect_distinct(ARRAY[1,2,2], ARRAY[2,2,4,5]) AS RESULT;
  result
-------------
 {2}
(1 row)

gaussdb=# SELECT array_intersect_distinct(ARRAY[1,2,3], NULL) AS RESULT;
 result
--------

(1 row)
```

- array_except(anyarray, anyarray)

  Description: Calculates the difference between two arrays. Only one-dimensional arrays are supported. If the first input parameter is **NULL**, **NULL** is returned. If the second input parameter is **NULL**, the first input parameter is returned.

  Return type: anyarray

  Example:

```
gaussdb=# SELECT array_except(ARRAY[1,2,3], ARRAY[3,4,5]) AS RESULT;
  result
-------------
 {1,2}
(1 row)

gaussdb=# SELECT array_except(ARRAY[1,2,3], NULL) AS RESULT;
 result
---------
 {1,2,3}
(1 row)

gaussdb=# SELECT array_except(NULL, ARRAY[3,4,5]) AS RESULT;
 result
--------

(1 row)
```

- array_except_distinct(anyarray, anyarray)

  Description: Calculates the difference between two arrays and deduplicates them. Only one-dimensional arrays are supported. If the first input parameter is **NULL**, **NULL** is returned. If the second input parameter is **NULL**, the first input parameter is returned. The first input parameter can contain only one-dimensional arrays. For multi-dimensional arrays, an error is reported.

  Return type: anyarray

  Example:

```
gaussdb=# SELECT array_except_distinct(ARRAY[1,2,2,3], ARRAY[3,4,5]) AS RESULT;
  result
-------------
 {1,2}
(1 row)

gaussdb=# SELECT array_except_distinct(ARRAY[1,2,3], NULL) AS RESULT;
 result
---------
 {1,2,3}
(1 row)

gaussdb=# SELECT array_except_distinct(NULL, ARRAY[3,4,5]) AS RESULT;
 result
--------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)
```

- array_ndims(anyarray)

  Description: Returns the number of dimensions of an array.

  Return type: int

  Example:

  ```
  gaussdb=# SELECT array_ndims(ARRAY[[1,2,3], [4,5,6]]) AS RESULT;
   result
  --------
       2
  (1 row)
  ```

- array_dims(anyarray)

  Description: Returns the lower and upper bounds of each dimension of an array.

  Return type: text

  Example:

  ```
  gaussdb=# SELECT array_dims(ARRAY[[1,2,3], [4,5,6]]) AS RESULT;
    result
  ------------
   [1:2][1:3]
  (1 row)
  ```

- array_length(anyarray, int)

  Description: Returns the length of the requested array dimension.

  Return type: int

  Example:

  ```
  gaussdb=# SELECT array_length(array[1,2,3], 1) AS RESULT;
   result
  --------
       3
  (1 row)
  ```

- array_lower(anyarray, int)

  Description: Returns lower bound of the requested array dimension.

  Return type: int

  Example:

  ```
  gaussdb=# SELECT array_lower('[0:2]={1,2,3}'::int[], 1) AS RESULT;
   result
  --------
       0
  (1 row)
  ```

- array_sort(anyarray)

  Description: Returns an array in ascending order. Only one-dimensional array of the anyarray type supports sorting. Multidimensional arrays return **NULL**. Currently, the record, XML, XMLtype, and JSON arrays cannot be sorted.

  Return type: anyarray

  Example:

  ```
  gaussdb=# SELECT array_sort(ARRAY[5,1,3,6,2,7]) AS RESULT;
   result
  -------------
  {1,2,3,5,6,7}
  (1 row)
  gaussdb=# SELECT array_sort(array[array[1,23], array[1,34]]);
   array_sort
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-------------
 {NULL,NULL}
(1 row)
```

- array_upper(anyarray, int)

  Description: Returns upper bound of the requested array dimension.

  Return type: int

  Example:

```
gaussdb=# SELECT array_upper(ARRAY[1,8,3,7], 1) AS RESULT;
 result
--------
      4
(1 row)
```

- array_to_string(anyarray, text [, text])

  Description: Uses the first **text** as the new delimiter and the second **text** to replace **NULL** values.

  Return type: text

  Example:

```
gaussdb=# SELECT array_to_string(ARRAY[1, 2, 3, NULL, 5], ',', '*') AS RESULT;
  result
-----------
 1,2,3,*,5
(1 row)
```

- array_delete(anyarray)

  Description: Clears elements in an array and returns an empty array of the same type.

  Return type: anyarray

  Example:

```
gaussdb=# SELECT array_delete(ARRAY[1,8,3,7]) AS RESULT;
 result
--------
 {}
(1 row)
```

- array_deleteidx(anyarray, int)

  Description: Deletes specified index elements from an array and returns an array consisting of the remaining elements.

  Return type: anyarray

  Example:

```
gaussdb=# SELECT array_deleteidx(ARRAY[1,2,3,4,5], 1) AS RESULT;
  result
-----------
 {2,3,4,5}
(1 row)
```

  NOTE

  array_deleteidx(anyarray, int): This function is disabled when the value of **a_format_version** is **10c** and the value of **a_format_dev_version** is **s1**.

- array_extendnull(anyarray, int)

  Description: Adds a specified number of null elements to the end of an array.

  Return type: anyarray

  Example:

```
gaussdb=# SELECT array_extendnull(ARRAY[1,8,3,7],1) AS RESULT;
   result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------------
{1,8,3,7,null}
(1 row)
```

- array_extendnull(anyarray, int, int)

  Description: Adds a specified number of elements with a specified index to the end of an array.

  Return type: anyarray

  Example:

```
gaussdb=# SELECT array_extendnull(ARRAY[1,8,3,7],2,2) AS RESULT;
   result
--------------
{1,8,3,7,8,8}
(1 row)
```

  NOTE

  array_extendnull(anyarray, int, int): This function takes effect when the value of **a_format_version** is **10c** and the value of **a_format_dev_version** is **s1**.

- array_trim(anyarray, int)

  Description: Removes a specified number of elements from the end of an array.

  Return type: anyarray

  Example:

```
gaussdb=# SELECT array_trim(ARRAY[1,8,3,7],1) AS RESULT;
 result
---------
 {1,8,3}
(1 row)
```

- array_exists(anyarray, int)

  Description: Checks whether the second parameter is a valid index of an array.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT array_exists(ARRAY[1,8,3,7],1) AS RESULT;
 result
--------
 t
(1 row)
```

- array_next(anyarray, int)

  Description: Returns the index of the element following a specified index in an array based on the second input parameter.

  Return type: int

  Example:

```
gaussdb=# SELECT array_next(ARRAY[1,8,3,7],1) AS RESULT;
 result
--------
      2
(1 row)
```

- array_prior(anyarray, int)

  Description: Returns the index of the element followed by a specified index in an array based on the second input parameter.

  Return type: int

  Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT array_prior(ARRAY[1,8,3,7],2) AS RESULT;
 result
--------
  1
(1 row)
```

- string_to_array(text, text, [text])

  Description: Uses the second **text** as the new delimiter and the third **text** as the substring to be replaced by **NULL** values. A substring can be replaced by **NULL** values only when it is the same as the third **text**.

  Return type: text[]

  Example:

```
gaussdb=# SELECT string_to_array('xx~^~yy~^~zz', '~^~', 'yy') AS RESULT;
    result
--------------
 {xx,NULL,zz}
(1 row)
gaussdb=# SELECT string_to_array('xx~^~yy~^~zz', '~^~', 'y') AS RESULT;
   result
------------
 {xx,yy,zz}
(1 row)
```

  In **string_to_array**, if the delimiter parameter is **NULL**, each character in the input string will become a separate element in the resulting array. If the delimiter is an empty string, then the entire input string is returned as a one-element array. Otherwise, the input string is split at each occurrence of the delimiter string.

  In **string_to_array**, if the null-string parameter is omitted or **NULL**, none of the substrings of the input will be replaced by **NULL**.

  In **array_to_string**, if the null-string parameter is omitted or **NULL**, any null elements in the array are simply skipped and not represented in the output string.

- unnest(anyarray)

  Description: Expands an array to a set of rows.

  Return type: setof anyelement

  Example:

```
gaussdb=# SELECT unnest(ARRAY[1,2]) AS RESULT;
 result
--------
      1
      2
(2 rows)
```

- _pg_keysequal

  Description: Checks whether two smallint arrays are the same.

  Parameter: smallint[], smallint[]

  Return type: Boolean

  > **NOTE**
  >
  > This function exists in the information_schema namespace.

- cardinality(anyarray)

  Description: Returns the total number of elements in each dimension of an array. If the array is empty, **0** is returned.

  Return type: integer

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT cardinality(array[[1, 2], [3, 4]]);
 cardinality
-------------
           4
(1 row)
```

- array_positions(anyarray, anyelement)

  Description: Returns an array of indexes of all second parameters that appear in the array given as the first parameter.

  Return type: int[]

  Example:

```
gaussdb=# SELECT array_positions(array[1, 2, 3, 1], 1) AS RESULT;
 result
--------
 {1,4}
(1 row)
```

  NOTE

  - The array must be one-dimensional.
  - The second parameter can be set to **NULL**.
  - If no second parameter is found in the array, an empty array is returned.

1.6.17 Range Functions and Operators

Range Operators

- =

  Description: Equals

  Example:

```
gaussdb=# SELECT int4range(1,5) = '[1,4]'::int4range AS RESULT;
 result
--------
 t
(1 row)
```

- <>

  Description: Does not equal

  Example:

```
gaussdb=# SELECT numrange(1.1,2.2) <> numrange(1.1,2.3) AS RESULT;
 result
--------
 t
(1 row)
```

- <

  Description: Is less than

  Example:

```
gaussdb=# SELECT int4range(1,10) < int4range(2,3) AS RESULT;
 result
--------
 t
(1 row)
```

- >

  Description: Is greater than

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT int4range(1,10) > int4range(1,5) AS RESULT;
 result
--------
 t
(1 row)
```

- <=

  Description: Is less than or equals

  Example:

  ```
  gaussdb=# SELECT numrange(1.1,2.2) <= numrange(1.1,2.2) AS RESULT;
   result
  --------
   t
  (1 row)
  ```

- >=

  Description: Greater than or equal

  Example:

  ```
  gaussdb=# SELECT numrange(1.1,2.2) >= numrange(1.1,2.0) AS RESULT;
   result
  --------
   t
  (1 row)
  ```

- @>

  Description: Contains range

  Example:

  ```
  gaussdb=# SELECT int4range(2,4) @> int4range(2,3) AS RESULT;
   result
  --------
   t
  (1 row)
  ```

- @>

  Description: Contains element

  Example:

  ```
  gaussdb=# SELECT '[2011-01-01,2011-03-01)'::tsrange @> '2011-01-10'::timestamp AS RESULT;
   result
  --------
   t
  (1 row)
  ```

- <@

  Description: Range is contained by

  Example:

  ```
  gaussdb=# SELECT int4range(2,4) <@ int4range(1,7) AS RESULT;
   result
  --------
   t
  (1 row)
  ```

- <@

  Description: Element is contained by

  Example:

  ```
  gaussdb=# SELECT 42 <@ int4range(1,7) AS RESULT;
   result
  --------
   f
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- &&

  Description: Overlap (have points in common)

  Example:

```
gaussdb=# SELECT int8range(3,7) && int8range(4,12) AS RESULT;
 result
--------
 t
(1 row)
```

- <<

  Description: Strictly left of

  Example:

```
gaussdb=# SELECT int8range(1,10) << int8range(100,110) AS RESULT;
 result
--------
 t
(1 row)
```

- >>

  Description: Strictly right of

  Example:

```
gaussdb=# SELECT int8range(50,60) >> int8range(20,30) AS RESULT;
 result
--------
 t
(1 row)
```

- &<

  Description: Does not extend to the right of

  Example:

```
gaussdb=# SELECT int8range(1,20) &< int8range(18,20) AS RESULT;
 result
--------
 t
(1 row)
```

- &>

  Description: Does not extend to the left of

  Example:

```
gaussdb=# SELECT int8range(7,20) &> int8range(5,10) AS RESULT;
 result
--------
 t
(1 row)
```

- -|-

  Description: Is adjacent to

  Example:

```
gaussdb=# SELECT numrange(1.1,2.2) -|- numrange(2.2,3.3) AS RESULT;
 result
--------
 t
(1 row)
```

- +

  Description: Union

  Example:

```
gaussdb=# SELECT numrange(5,15) + numrange(10,20) AS RESULT;
 result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------
 [5,20)
(1 row)
```

- *

  Description: Intersection

  Example:

```
gaussdb=# SELECT int8range(5,15) * int8range(10,20) AS RESULT;
 result
---------
 [10,15)
(1 row)
```

- -

  Description: Difference

  Example:

```
gaussdb=# SELECT int8range(5,15) - int8range(10,20) AS RESULT;
 result
--------
 [5,10)
(1 row)
```

The simple comparison operators **<**, **>**, **<=**, and **>=** compare the lower bounds first, and only if those are equal, compare the upper bounds.

The **<<**, **>>**, and **-|-** operators always return false when an empty range is involved; that is, an empty range is not considered to be either before or after any other range.

The union and difference operators will fail if the resulting range would need to contain two disjoint sub-ranges.

Range Functions

The lower and upper functions return null if the range is empty or the requested bound is infinite. The lower_inc, upper_inc, lower_inf, and upper_inf functions all return false for an empty range.

- numrange(numeric, numeric, [text])

  Description: Specifies a range.

  Return type: range's element type

  Example:

```
gaussdb=# SELECT numrange(1.1,2.2) AS RESULT;
 result
--------
[1.1,2.2)
(1 row)
gaussdb=# SELECT numrange(1.1,2.2, '()') AS RESULT;
 result
--------
(1.1,2.2)
(1 row)
```

- lower(anyrange)

  Description: Lower bound of range

  Return type: range's element type

  Example:

```
gaussdb=# SELECT lower(numrange(1.1,2.2)) AS RESULT;
 result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------
   1.1
(1 row)
```

- upper(anyrange)

  Description: Specifies the upper bound of a range.

  Return type: range's element type

  Example:

  ```
  gaussdb=# SELECT upper(numrange(1.1,2.2)) AS RESULT;
   result
  --------
     2.2
  (1 row)
  ```

- isempty(anyrange)

  Description: Specifies whether a range is empty.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT isempty(numrange(1.1,2.2)) AS RESULT;
   result
  --------
   f
  (1 row)
  ```

- lower_inc(anyrange)

  Description: Specifies whether the lower bound is inclusive.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT lower_inc(numrange(1.1,2.2)) AS RESULT;
   result
  --------
   t
  (1 row)
  ```

- upper_inc(anyrange)

  Description: Specifies whether the upper bound is inclusive.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT upper_inc(numrange(1.1,2.2)) AS RESULT;
   result
  --------
   f
  (1 row)
  ```

- lower_inf(anyrange)

  Description: Specifies whether the lower bound is infinite.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT lower_inf('(,)'::daterange) AS RESULT;
   result
  --------
   t
  (1 row)
  ```

- upper_inf(anyrange)

  Description: Specifies whether the upper bound is infinite.

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT upper_inf('(,)'::daterange) AS RESULT;
 result
--------
 t
(1 row)
```

- elem_contained_by_range(anyelement, anyrange)

  Description: Determines whether an element is within the range.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT elem_contained_by_range('2', numrange(1.1,2.2));
   elem_contained_by_range
  -------------------------
   t
  (1 row)
  ```

  NOTE

  In Oracle-compatible mode, after the **mapping_date_to_datea** parameter is enabled, the date type is mapped to the datea type. The datea type does not support being used as a range type, and this function does not support the datea type as an input parameter.

- in_range_int16_int16(int16, int16, int16, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  – First parameter: It specifies the number to be checked against the given range.

  – The second parameter: It specifies the start value of the current range.

  – Third parameter: It is a range value, which must be a positive number.

  – Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.

    ▪ TRUE: Compare the second parameter minus the third parameter with the first parameter.

    ▪ FALSE: Compare the second parameter plus the third parameter with the first parameter.

  – Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.

    ▪ TRUE: Check whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is TRUE. Otherwise, the return value of the function is FALSE.

    ▪ FALSE: Check whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is TRUE. Otherwise, the return value of the function is FALSE.

  Return type: Boolean

  Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT in_range_int16_int16(1::int16, 1::int16, 1::int16, true, true);
 in_range_int16_int16
----------------------
 f
(1 row)

gaussdb=# SELECT in_range_int16_int16(1::int16, 4::int16, 2::int16, true, true);
 in_range_int16_int16
----------------------
 t
(1 row)
```

- in_range_int8_int16(int8, int8, int16, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  – First parameter: It specifies the number to be checked against the given range.

  – The second parameter: It specifies the start value of the current range.

  – Third parameter: It is a range value, which must be a positive number.

  – Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.

    ▪ TRUE: Compare the second parameter minus the third parameter with the first parameter.

    ▪ FALSE: Compare the second parameter plus the third parameter with the first parameter.

  – Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.

    ▪ TRUE: Check whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is TRUE. Otherwise, the return value of the function is FALSE.

    ▪ FALSE: Check whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is TRUE. Otherwise, the return value of the function is FALSE.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int8_int16(1::int8, 1::int8, 1::int16, true, true);
 in_range_int8_int16
---------------------
 f
(1 row)

gaussdb=# SELECT in_range_int8_int16(1::int8, 4::int8, 2::int16, true, true);
 in_range_int8_int16
---------------------
 t
(1 row)
```

- in_range_int4_int16(int4, int4, int16, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters:

- First parameter: It specifies the number to be checked against the given range.
- The second parameter: It specifies the start value of the current range.
- Third parameter: It specifies a range value, which must be a positive number.
- Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
  - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
  - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
- Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
  - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
  - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

Return type: Boolean

Example:

```
gaussdb=# SELECT in_range_int4_int16(1::int4, 1::int4, 1::int16, true, true);
 in_range_int4_int16
---------------------
 f
(1 row)

gaussdb=# SELECT in_range_int4_int16(1::int4, 4::int4, 2::int16, true, true);
 in_range_int4_int16
---------------------
 t
(1 row)
```

- in_range_int2_int16(int2, int2, int16, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int2_int16(1::int2, 1::int2, 1::int16, true, true);
 in_range_int2_int16
---------------------
 f
(1 row)

gaussdb=# SELECT in_range_int2_int16(1::int2, 4::int2, 2::int16, true, true);
 in_range_int2_int16
---------------------
 t
```

- in_range_int1_int16(int1, int1, int16, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

Return type: Boolean

```
gaussdb=# SELECT in_range_int1_int16(1::int1, 1::int1, 1::int16, true, true);
 in_range_int1_int16
---------------------
 f
(1 row)

gaussdb=# SELECT in_range_int1_int16(1::int1, 4::int1, 2::int16, true, true);
 in_range_int1_int16
---------------------
 t
(1 row)
```

- in_range_float8_float8(float8, float8, float8, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  – First parameter: It specifies the number to be checked against the given range.

  – The second parameter: It specifies the start value of the current range.

  – Third parameter: It specifies a range value, which must be a positive number.

  – Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.

    ▪ **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.

    ▪ **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.

  – Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.

    ▪ **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    ▪ **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_float8_float8(1.0::float8, 1.9::float8, 1.0::float8, true, true);
 in_range_float8_float8
------------------------
 f
(1 row)
```

- in_range_float4_float8(float4, float4, float8, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters:

- First parameter: It specifies the number to be checked against the given range.
- The second parameter: It specifies the start value of the current range.
- Third parameter: It specifies a range value, which must be a positive number.
- Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
  - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
  - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
- Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
  - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
  - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

Return type: Boolean

Example:

```
gaussdb=# SELECT in_range_float4_float8(1.0::float4, 1.9::float4, 1.0::float8, true, true);
 in_range_float4_float8
------------------------
 f
(1 row)
```

- in_range_numeric_numeric(numeric, numeric, numeric, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_numeric_numeric(1.0::numeric(10,4), 1.9::numeric(10,4),
1.0::numeric(10,4), true, true);
 in_range_numeric_numeric
--------------------------
 f
(1 row)
```

- in_range_interval_interval(interval, interval, interval, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_interval_interval(interval '13' hour, interval '10' hour, interval '3' hour,
false, true);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 in_range_interval_interval
----------------------------
 t
(1 row)
```

- in_range_timestamp_interval(timestamp, timestamp, interval, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  – First parameter: It specifies the number to be checked against the given range.

  – The second parameter: It specifies the start value of the current range.

  – Third parameter: It specifies a range value, which must be a positive number.

  – Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.

    ▪ **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.

    ▪ **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.

  – Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.

    ▪ **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    ▪ **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

```
gaussdb=# SELECT in_range_timestamp_interval('12-10-2010 11:25:00', '10-11-2010 11:26:00', interval
'60' day, false, true);
 in_range_timestamp_interval
-----------------------------
 t
(1 row)
```

- in_range_timestamptz_interval(timestamptz, timestamptz, interval, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  – First parameter: It specifies the number to be checked against the given range.

  – The second parameter: It specifies the start value of the current range.

  – Third parameter: It specifies a range value, which must be a positive number.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_timestamptz_interval('12-10-2010 11:25:00+00', '10-11-2010
11:26:00+00', interval '60' day, false, true);
 in_range_timestamptz_interval
-------------------------------
 t
(1 row)
```

- in_range_time_interval(time, time, interval, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_time_interval('00:00:00', '12:00:00', interval '13' hour, true, true);
 in_range_time_interval
------------------------
 f
(1 row)
```

- in_range_timetz_interval(timetz, timetz, interval, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_timetz_interval('00:00:00+00', '12:00:00+00', interval '13' hour, true, true);
 in_range_timetz_interval
--------------------------
 f
(1 row)
```

- in_range_int8_numeric(int8, int8, numeric, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int8_numeric(1::int8, 2::int8, 0.9::numeric(10,4), true, true);
 in_range_int8_numeric
-----------------------
 t
(1 row)
```

- in_range_int4_numeric(int4, int4, numeric, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int4_numeric(1::int4, 2::int4, 0.9::numeric(10,4), true, true);
 in_range_int4_numeric
-----------------------
 t
(1 row)
```

- in_range_int2_numeric(int2, int2, numeric, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  – First parameter: It specifies the number to be checked against the given range.
  – The second parameter: It specifies the start value of the current range.
  – Third parameter: It specifies a range value, which must be a positive number.
  – Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  – Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int2_numeric(1::int2, 2::int2, 0.9::numeric(10,4), true, true);
 in_range_int2_numeric
-----------------------
 t
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- in_range_int1_numeric(int1, int1, numeric, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT in_range_int1_numeric(1::int1, 2::int1, 0.9::numeric(10,4), true, true);
   in_range_int1_numeric
  -----------------------
   t
  (1 row)
  ```

- in_range_numeric_float4(numeric, numeric, float4, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

        - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
    - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
        - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
        - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    Return type: Boolean

    Example:

    ```
    gaussdb=# SELECT in_range_numeric_float4(1.0::numeric(10,4), 1.9::numeric(10,4), 1.0::float4, true, true);
     in_range_numeric_float4
    -------------------------
     f
    (1 row)
    ```

- in_range_numeric_float8(numeric, numeric, float8, boolean, boolean)

    Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

    Parameters:

    - First parameter: It specifies the number to be checked against the given range.
    - The second parameter: It specifies the start value of the current range.
    - Third parameter: It specifies a range value, which must be a positive number.
    - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
        - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
        - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
    - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
        - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
        - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT in_range_numeric_float8(1.0::numeric(10,4), 1.9::numeric(10,4), 1.0::float8, true,
true);
 in_range_numeric_float8
-------------------------
 f
(1 row)
```

- in_range_int8_float8(int8, int8, float8, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT in_range_int8_float8(1::int8, 2::int8, 0.9::float8, true, true);
   in_range_int8_float8
  ----------------------
   t
  (1 row)
  ```

- in_range_int4_float8(int4, int4, float8, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int4_float8(1::int4, 2::int4, 0.9::float8, true, true);
 in_range_int4_float8
----------------------
 t
(1 row)
```

- in_range_int2_float8(int2, int2, float8, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

        - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    Return type: Boolean

    Example:

```
gaussdb=# SELECT in_range_int2_float8(1::int2, 2::int2, 0.9::float8, true, true);
 in_range_int2_float8
----------------------
 t
(1 row)
```

- in_range_int1_float8(int1, int1, float8, boolean, boolean)

    Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

    Parameters:

    - First parameter: It specifies the number to be checked against the given range.
    - The second parameter: It specifies the start value of the current range.
    - Third parameter: It specifies a range value, which must be a positive number.
    - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
        - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
        - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
    - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
        - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
        - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    Return type: Boolean

    Example:

```
gaussdb=# SELECT in_range_int1_float8(1::int1, 2::int1, 0.9::float8, true, true);
 in_range_int1_float8
----------------------
 t
(1 row)
```

- in_range_float8_int16(float8, float8, int16, boolean, boolean)

    Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

    Parameters:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_float8_int16(1.0::float8, 1.9::float8, 1::int16, true, true);
 in_range_float8_int16
-----------------------
 f
(1 row)
```

- in_range_float4_int16(float4, float4, int16, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_float4_int16(1.0::float4, 1.9::float4, 1::int16, true, true);
 in_range_float4_int16
-----------------------
 f
(1 row)
```

- in_range_float8_int8(float8, float8, int8, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_float8_int8(1.0::float8, 1.9::float8, 1::int8, true, true);
 in_range_float8_int8
----------------------
 f
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- in_range_float4_int8(float4, float4, int8, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT in_range_float4_int8(1.0::float4, 1.9::float4, 1::int8, true, true);
   in_range_float4_int8
  ----------------------
   f
  (1 row)
  ```

- in_range_float8_int4(float8, float8, int4, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

        - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
    - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
        - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
        - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    Return type: Boolean

    Example:

    ```
    gaussdb=# SELECT in_range_float8_int4(1.0::float8, 1.9::float8, 1::int4, true, true);
     in_range_float8_int4
    ----------------------
     f
    (1 row)
    ```

- in_range_float4_int4(float4, float4, int4, boolean, boolean)

    Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

    Parameters:

    - First parameter: It specifies the number to be checked against the given range.
    - The second parameter: It specifies the start value of the current range.
    - Third parameter: It specifies a range value, which must be a positive number.
    - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
        - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
        - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
    - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
        - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
        - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT in_range_float4_int4(1.0::float4, 1.9::float4, 1::int4, true, true);
 in_range_float4_int4
----------------------
 f
(1 row)
```

- in_range_float8_int2(float8, float8, int2, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT in_range_float8_int2(1.0::float8, 1.9::float8, 1::int2, true, true);
   in_range_float8_int2
  ----------------------
   f
  (1 row)
  ```

- in_range_float4_int2(float4, int2, int2, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_float4_int2(1.0::float4, 1.9::float4, 1::int2, true, true);
 in_range_float4_int2
----------------------
 f
(1 row)
```

- in_range_float8_int1(float8, float8, int1, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_float8_int1(1.0::float8, 1.9::float8, 1::int1, true, true);
 in_range_float8_int1
----------------------
 f
(1 row)
```

- in_range_float4_int1(float4, float4, int1, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  – First parameter: It specifies the number to be checked against the given range.
  – The second parameter: It specifies the start value of the current range.
  – Third parameter: It specifies a range value, which must be a positive number.
  – Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  – Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_float4_int1(1.0::float4, 1.9::float4, 1::int1, true, true);
 in_range_float4_int1
----------------------
 f
(1 row)
```

- in_range_float8_float4(float8, float8, float4, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_float8_float4(1.0::float8, 1.9::float8, 1.0::float4, true, true);
 in_range_float8_float4
------------------------
 f
(1 row)
```

- in_range_float4_float4(float4, float4, float4, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_float4_float4(1.0::float4, 1.9::float4, 1.0::float4, true, true);
 in_range_float4_float4
------------------------
 f
(1 row)
```

- in_range_float8_numeric(float8, float8, numeric, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_float8_numeric(1.0::float8, 1.9::float8, 1.0::numeric(10,4), true, true);
 in_range_float8_numeric
-------------------------
 f
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- in_range_float4_numeric(float4, float4, numeric, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT in_range_float4_numeric(1.0::float4, 1.9::float4, 1.0::numeric(10,4), true, true);
   in_range_float4_numeric
  -------------------------
   f
  (1 row)
  ```

- in_range_numeric_int16(numeric, numeric, int16, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

        - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
    - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
        - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
        - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    Return type: Boolean

    Example:

    ```
    gaussdb=# SELECT in_range_numeric_int16(1.0::numeric(10,4), 1.9::numeric(10,4), 1.0::int16, true, true);
     in_range_numeric_int16
    ------------------------
     f
    (1 row)
    ```

- in_range_numeric_int8(numeric, numeric, int8, boolean, boolean)

    Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

    Parameters:

    - First parameter: It specifies the number to be checked against the given range.
    - The second parameter: It specifies the start value of the current range.
    - Third parameter: It specifies a range value, which must be a positive number.
    - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
        - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
        - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
    - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
        - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
        - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT in_range_numeric_int8(1.0::numeric(10,4), 1.9::numeric(10,4), 1.0::int8, true, true);
 in_range_numeric_int8
-----------------------
 f
(1 row)
```

- in_range_numeric_int4(numeric, numeric, int4, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT in_range_numeric_int4(1.0::numeric(10,4), 1.9::numeric(10,4), 1.0::int4, true, true);
   in_range_numeric_int4
  -----------------------
   f
  (1 row)
  ```

- in_range_numeric_int1(numeric, numeric, int1, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_numeric_int1(1.0::numeric(10,4), 1.9::numeric(10,4), 1.0::int1, true, true);
 in_range_numeric_int1
-----------------------
 f
(1 row)
```

- in_range_int8_int8(int8, int8, int8, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

        - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    Return type: Boolean

    Example:

```
gaussdb=# SELECT in_range_int8_int8(1::int8, 1::int8, 1::int8, true, true);
 in_range_int8_int8
--------------------
 f
(1 row)
```

- in_range_int4_int8(int4, int4, int8, boolean, boolean)

    Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

    Parameters:

    - First parameter: It specifies the number to be checked against the given range.
    - The second parameter: It specifies the start value of the current range.
    - Third parameter: It specifies a range value, which must be a positive number.
    - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
        - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
        - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
    - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
        - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
        - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    Return type: Boolean

    Example:

```
gaussdb=# SELECT in_range_int4_int8(1::int4, 1::int4, 1::int8, true, true);
 in_range_int4_int8
--------------------
 f
(1 row)
```

- in_range_int2_int8(int2, int2, int8, boolean, boolean)

    Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

    Parameters:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int2_int8(1::int2, 1::int2, 1::int8, true, true);
 in_range_int2_int8
--------------------
 f
(1 row)
```

- in_range_int1_int8(int1, int1, int8, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int1_int8(1::int1, 1::int1, 1::int8, true, true);
 in_range_int1_int8
--------------------
 f
(1 row)
```

- in_range_int8_int4(int8, int8, int4, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  – First parameter: It specifies the number to be checked against the given range.
  – The second parameter: It specifies the start value of the current range.
  – Third parameter: It specifies a range value, which must be a positive number.
  – Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  – Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int8_int4(1::int8, 1::int8, 1::int4, true, true);
 in_range_int8_int4
--------------------
 f
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- in_range_int4_int4(int4, int4, int4, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT in_range_int4_int4(1::int4, 1::int4, 1::int4, true, true);
   in_range_int4_int4
  --------------------
   f
  (1 row)
  ```

- in_range_int2_int4(int2, int2, int4, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int2_int4(1::int2, 1::int2, 1::int4, true, true);
 in_range_int2_int4
--------------------
 f
(1 row)
```

- in_range_int1_int4(int1, int1, int4, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT in_range_int1_int4(1::int1, 1::int1, 1::int4, true, true);
 in_range_int1_int4
--------------------
 f
(1 row)
```

- in_range_int8_int2(int8, int8, int2, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  – First parameter: It specifies the number to be checked against the given range.

  – The second parameter: It specifies the start value of the current range.

  – Third parameter: It specifies a range value, which must be a positive number.

  – Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.

    ▪ **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.

    ▪ **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.

  – Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.

    ▪ **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    ▪ **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT in_range_int8_int2(1::int8, 1::int8, 1::int2, true, true);
   in_range_int8_int2
  --------------------
   f
  (1 row)
  ```

- in_range_int4_int2(int4, int4, int2, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  – First parameter: It specifies the number to be checked against the given range.

  – The second parameter: It specifies the start value of the current range.

  – Third parameter: It specifies a range value, which must be a positive number.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int4_int2(1::int4, 1::int4, 1::int2, true, true);
 in_range_int4_int2
--------------------
 f
(1 row)
```

- in_range_int2_int2(int2, int2, int2, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int2_int2(1::int2, 1::int2, 1::int2, true, true);
 in_range_int2_int2
--------------------
 f
(1 row)
```

- in_range_int1_int2(int1, int1, int2, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int1_int2(1::int1, 1::int1, 1::int2, true, true);
 in_range_int1_int2
--------------------
 f
(1 row)
```

- in_range_int8_int1(int8, int8, int1, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int8_int1(1::int8, 1::int8, 1::int1, true, true);
 in_range_int8_int1
--------------------
 f
(1 row)
```

- in_range_int4_int1(int4, int4, int1, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
  - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

Return type: Boolean

Example:

```
gaussdb=# SELECT in_range_int4_int1(1::int4, 1::int4, 1::int1, true, true);
 in_range_int4_int1
--------------------
 f
(1 row)
```

- in_range_int2_int1(int2, int2, int1, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  – First parameter: It specifies the number to be checked against the given range.
  – The second parameter: It specifies the start value of the current range.
  – Third parameter: It specifies a range value, which must be a positive number.
  – Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  – Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int2_int1(1::int2, 1::int2, 1::int1, true, true);
 in_range_int2_int1
--------------------
 f
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- in_range_int1_int1(int1, int1, int1, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT in_range_int1_int1(1::int1, 1::int1, 1::int1, true, true);
   in_range_int1_int1
  --------------------
   f
  (1 row)
  ```

- in_range_int8_float4(int8, int8, float4, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

        - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
    - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
        - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
        - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

    Return type: Boolean

    Example:

```
gaussdb=# SELECT in_range_int8_float4(1::int8, 2::int8, 0.9::float4, true, true);
 in_range_int8_float4
----------------------
 t
(1 row)
```

- in_range_int4_float4(int4, int4, float4, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT in_range_int4_float4(1::int4, 2::int4, 0.9::float4, true, true);
 in_range_int4_float4
----------------------
 t
(1 row)
```

- in_range_int2_float4(int2, int2, float4, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.
  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT in_range_int2_float4(1::int2, 2::int2, 0.9::float4, true, true);
   in_range_int2_float4
  ----------------------
   t
  (1 row)
  ```

- in_range_int1_float4(int1, int1, float4, boolean, boolean)

  Description: Checks whether the first parameter is within the range of the second parameter plus or minus the third parameter.

  Parameters:

  - First parameter: It specifies the number to be checked against the given range.
  - The second parameter: It specifies the start value of the current range.
  - Third parameter: It specifies a range value, which must be a positive number.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Fourth parameter: It specifies whether the current operation is a subtraction or addition operation.
    - **TRUE**: The function compares the second parameter minus the third parameter with the first parameter.
    - **FALSE**: The function compares the second parameter plus the third parameter with the first parameter.
  - Fifth parameter: It compares the operation result of the second and third parameters with the first parameter.
    - **TRUE**: The function checks whether the result is greater than the first parameter. If the result is greater than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.
    - **FALSE**: The function checks whether the result is less than or equal to the first parameter. If the result is less than or equal to the first parameter, the return value of the function is **TRUE**. Otherwise, the return value of the function is **FALSE**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT in_range_int1_float4(1::int1, 2::int1, 0.9::float4, true, true);
 in_range_int1_float4
----------------------
 t
(1 row)
```

1.6.18 Aggregate Functions

Aggregate Functions

- sum(expression)

  Description: Sum of expression across all input values

  Return type:

  Generally, same as the argument data type. In the following cases, type conversion occurs:

  - BIGINT for SMALLINT or INT arguments
  - NUMBER for BIGINT arguments
  - DOUBLE PRECISION for floating-point arguments

  Example:

```
gaussdb=# CREATE TABLE tab(a int);
CREATE TABLE
gaussdb=# INSERT INTO tab VALUES(1);
INSERT 0 1
gaussdb=# INSERT INTO tab VALUES(2);
INSERT 0 1
gaussdb=# SELECT sum(a) FROM tab;
 sum
-----
   3
(1 row)
gaussdb=# DROP TABLE tab;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- max(expression)

  Description: Specifies the maximum value of expression across all input values.

  Parameter type: any array, number, character string, date/time, or ROWID.

  Return type: same as the parameter type

  Example:

```
gaussdb=# CREATE TABLE max_t1(a int, b int);

gaussdb=# INSERT INTO max_t1 VALUES(1,2),(2,3),(3,4),(4,5);

gaussdb=# SELECT MAX(a) FROM max_t1;
 max
-----
   4
(1 row)

gaussdb=# DROP TABLE max_t1;
```

- min(expression)

  Description: Minimum value of expression across all input values

  Parameter type: any array, number, character string, date/time, or ROWID.

  Return type: same as the parameter type

  Example:

```
gaussdb=# CREATE TABLE min_t1(a int, b int);

gaussdb=# INSERT INTO min_t1 VALUES(1,2),(2,3),(3,4),(4,5);

gaussdb=# SELECT MIN(a) FROM min_t1;
 min
-----
   1
(1 row)

gaussdb=# DROP TABLE min_t1;
```

- avg(expression)

  Description: Average (arithmetic mean) of all input values

  Return type:

  – For any input parameter of the integer type or that can be implicitly converted to the integer type, the return type is NUMBER.
  – For any input parameter of the floating-point type or that can be implicitly converted to the floating-point type, the return type is DOUBLE PRECISION.
  – In other cases, the data type is the same as that of the input parameter.

  Example:

```
gaussdb=# CREATE TABLE avg_t1(a int, b int);

gaussdb=# INSERT INTO avg_t1 VALUES(1,2),(2,3),(3,4),(4,5);

gaussdb=# SELECT AVG(a) FROM avg_t1;
        avg
--------------------
 2.5000000000000000
(1 row)

gaussdb=# DROP TABLE avg_t1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- count(expression)

  Description: Returns the number of input rows for which the value of expression is not null.

  Return type: bigint

  Operations on XML data are supported.

  Example:

```
gaussdb=# CREATE TABLE count_t1(a int, b int);

gaussdb=# INSERT INTO count_t1 VALUES (NULL,1),(1,2),(2,3),(3,4),(4,5);

gaussdb=# SELECT COUNT(a) FROM count_t1;
 count
-------
     4
(1 row)

gaussdb=# DROP TABLE count_t1;
```

- count(*)

  Description: Returns the number of input rows.

  Return type: bigint

  Operations on XML data are supported.

  Example:

```
gaussdb=# CREATE TABLE count_t1(a int, b int);

gaussdb=# INSERT INTO count_t1 VALUES (NULL,1),(1,2),(2,3),(3,4),(4,5);

gaussdb=# SELECT COUNT(*) FROM count_t1;
 count
-------
     5
(1 row)

gaussdb=# DROP TABLE count_t1;
```

- KEEP(DENSE_RANK { FIRST | LAST } ORDER BY expr)

  Description: Sorts data rows in a query result based on specified rules and returns the first or last group of values. This can be used in aggregate or window functions.

  Syntax:

```
aggregate_function KEEP ( DENSE_RANK { FIRST | LAST }  ORDER BY expr [ DESC | ASC ] [ NULLS { FIRST | LAST } ]
     [, expr [ DESC | ASC ] [ NULLS { FIRST | LAST } ]]...)
```

  Return type: data type of the specified column in aggregate_function.

  NOTE

  - aggregate_function is an aggregate function. Currently, only MIN, MAX, SUM, AVG, COUNT, VARIANCE, and STDDEV are supported.
  - The KEEP keyword is used for clearer semantics and is limited to aggregate functions and window functions.
  - DENSE_RANK FIRST and DENSE_RANK LAST indicate that only rows after the minimum (FIRST) and maximum (LAST) dense ranking are aggregated, respectively.
  - Vectorized plans cannot be generated for execution.

  Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CREATE TABLE keep_t1(a int, b int, c int);

gaussdb=# INSERT INTO keep_t1 values(2, 111, 333), (2, 11, 33), (2, 11, 3), (1, 22, 55), (1, 2, 5);

gaussdb=#  SELECT MIN(b) KEEP(DENSE_RANK FIRST ORDER BY a) MIN_FIRST,
        MIN(b) KEEP(DENSE_RANK LAST ORDER BY a) MIN_LAST,
        MAX(b) KEEP(DENSE_RANK FIRST ORDER BY a) MAX_FIRST,
        MAX(b) KEEP(DENSE_RANK LAST ORDER BY a) MAX_LAST FROM keep_t1;
 min_first | min_last | max_first | max_last
-----------+----------+-----------+----------
         2 |       11 |        22 |      111
(1 row)

gaussdb=# DROP TABLE keep_t1;
```

- default(column_name)

  Description: Obtains the default value of a table column.

  Return type: text

  Example:

```
-- Create a MySQL-compatible database.
gaussdb=# CREATE DATABASE gaussdb_m WITH DBCOMPATIBILITY 'MYSQL';

gaussdb=# \c gaussdb_m

gaussdb_m=# CREATE TABLE t1(id int DEFAULT 100, name varchar(20) DEFAULT 'tt');

gaussdb_m=# INSERT INTO t1 VALUES(1,'test');

-- Execute the query.
gaussdb_m=# SELECT default(id), default(name) FROM t1;
 default | default
---------+---------
     100 | tt
(1 row)

-- Delete the database.
gaussdb_m=# \c postgres
gaussdb=# DROP DATABASE gaussdb_m;
```

  NOTE

  - The default function takes effect only when **sql_compatibility** is set to **'MYSQL'**.
  - If a table column does not have a default value, the default function returns a null value.
  - If a table column is a hidden column (such as **xmin** or **cmin**), the default function returns a null value.
  - If a table column is an auto-increment column, the default function returns **0**.
  - GaussDB supports default values of partitioned tables, temporary tables, and multi-table join query.
  - GaussDB supports the query of nodes whose column names contain character string values (indicating names) and A_Star nodes (indicating that asterisks (*) appear), for example, **default(tt.t4.id)** and **default(tt.t4.*)**.
  - When the default value of a column is created in GaussDB, the range of the column type is not verified. As a result, an error may be reported when the default function is used.
  - If the default value of a column is a function expression, the default function of GaussDB will return the calculated value of the default expression set for the column during table creation.

- array_agg(expression)

  Description: Input values, including nulls, concatenated into an array

  Return type: array of the parameter type

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operations on XML data are supported.

Example:

```
gaussdb=# CREATE TABLE array_agg_t1(a int, b int);

gaussdb=# INSERT INTO array_agg_t1 VALUES (NULL,1),(1,2),(2,3),(3,4),(4,5);

gaussdb=# SELECT ARRAY_AGG(a) FROM array_agg_t1;
  array_agg
----------------
 {NULL,1,2,3,4}
(1 row)

gaussdb=# DROP TABLE array_agg_t1;
```

- string_agg(expression, delimiter)

  Description: Input values concatenated into a string, separated by delimiter

  Return type: same as the parameter data type.

  Operations on XML data that is explicitly converted to the character type are supported.

  Example:

```
gaussdb=# CREATE TABLE string_agg_t1(a int, b int);

gaussdb=# INSERT INTO string_agg_t1 VALUES (NULL,1),(1,2),(2,3),(3,4),(4,5);

gaussdb=# SELECT STRING_AGG(a,';') FROM string_agg_t1;
 string_agg
------------
 1;2;3;4
(1 row)

gaussdb=# DROP TABLE string_agg_t1;
```

- listagg([DISTINCT | ALL] expression [, delimiter]) [WITHIN GROUP(ORDER BY order-list)]

  Description: Aggregation column data sorted according to the mode specified by WITHIN GROUP, and concatenated to a string using the specified delimiter

  - **DISTINCT**: Optional. It deduplicates the result after each row is concatenated. Only the ORA-compatible mode is supported.
  - **expression**: Required. It specifies an aggregation column name or a column-based, valid expression. It does not support the VARIADIC parameters.
  - **delimiter**: Optional. It specifies a delimiter, which can be a string constant or a deterministic expression based on a group of columns. The default value is empty.
  - **order-list**: Optional. It specifies the sorting mode in a group.

  Return type: text

  > **NOTE**
  >
  > listagg is a column-to-row aggregate function, compatible with Oracle Database 11g Release 2. You can specify the OVER clause as a window function. When listagg is used as a window function, the OVER clause does not support the window sorting or framework of ORDER BY, to avoid different meanings or behaviors in listagg and ORDER BY of the WITHIN GROUP clause.

  Example:

  The aggregation column is of the text character set type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CREATE TABLE listagg_t1(a int, b text);

gaussdb=# INSERT INTO listagg_t1 VALUES (NULL,'a1'),(1,'b2'),(1,'c3'),(2,'d4'),(2,'e5'),(3,'f6');

gaussdb=# SELECT a,LISTAGG(b,';') WITHIN GROUP(ORDER BY b) FROM listagg_t1 GROUP BY a
ORDER BY a;
 a | listagg
---+---------
 1 | b2;c3
 2 | d4;e5
 3 | f6
   | a1
(4 rows)

gaussdb=# SELECT a,LISTAGG(b,';') FROM listagg_t1 GROUP BY a ORDER BY a;
 a | listagg
---+---------
 1 | b2;c3
 2 | d4;e5
 3 | f6
   | a1
(4 rows)

gaussdb=# DROP TABLE listagg_t1;
```

The aggregation column is of the integer type.

```
gaussdb=# CREATE TABLE listagg_t1(a int, b int);

gaussdb=# INSERT INTO listagg_t1 VALUES (NULL,1),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT a,LISTAGG(b,';') WITHIN GROUP(ORDER BY b) FROM listagg_t1 GROUP BY a
ORDER BY a;
 a | listagg
---+---------
 1 | 2;3
 2 | 4;5
 3 | 6
   | 1
(4 rows)

gaussdb=# DROP TABLE listagg_t1;
```

The aggregation column is of the floating-point type.

```
gaussdb=# CREATE TABLE listagg_t1(a int, b float);

gaussdb=# INSERT INTO listagg_t1 VALUES (NULL,1.111),(1,2.222),(1,3.333),(2,4.444),(2,5.555),
(3,6.666);

gaussdb=# SELECT a,LISTAGG(b,';') WITHIN GROUP(ORDER BY b) FROM listagg_t1 GROUP BY a
ORDER BY a;
 a |      listagg
---+-------------------
 1 | 2.222000;3.333000
 2 | 4.444000;5.555000
 3 | 6.666000
   | 1.111000
(4 rows)

gaussdb=# DROP TABLE listagg_t1;
```

The aggregation column is of the time type.

```
gaussdb=# CREATE TABLE listagg_t1(a int, b timestamp);

gaussdb=# INSERT INTO listagg_t1 VALUES (NULL,'2000-01-01'),(1,'2000-02-02'),(1,'2000-03-03'),
(2,'2000-04-04'),(2,'2000-05-05'),(3,'2000-06-06');

gaussdb=# SELECT a,LISTAGG(b,';') WITHIN GROUP(ORDER BY b) FROM listagg_t1 GROUP BY a
ORDER BY a;
 a |                 listagg
---+-----------------------------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 1 | 2000-02-02 00:00:00;2000-03-03 00:00:00
 2 | 2000-04-04 00:00:00;2000-05-05 00:00:00
 3 | 2000-06-06 00:00:00
   | 2000-01-01 00:00:00
(4 rows)

gaussdb=# DROP TABLE listagg_t1;
```

The aggregation column is of the time interval type.

```
gaussdb=# CREATE TABLE listagg_t1(a int, b interval);

gaussdb=# INSERT INTO listagg_t1 VALUES (NULL,'1 days'),(1,'2 days'),(1,'3 days'),(2,'4 days'),(2,'5
days'),(3,'6 days');

gaussdb=# SELECT a,LISTAGG(b,';') WITHIN GROUP(ORDER BY b) FROM listagg_t1 GROUP BY a
ORDER BY a;
 a |    listagg
---+---------------
 1 | 2 days;3 days
 2 | 4 days;5 days
 3 | 6 days
   | 1 day
(4 rows)

gaussdb=# DROP TABLE listagg_t1;
```

The aggregation column is of the ROWID type.

```
gaussdb=# CREATE TABLE test_table_1(id int, name varchar(29), rwd rowid);
CREATE TABLE
gaussdb=# INSERT INTO test_table_1 VALUES (1, 'AAAAAAAAAP//AAAAAAAAAAAAAAAAAA',
'AAAAAAAAAP//AAAAAAAAAAAAAAAAAA');
INSERT 0 1
gaussdb=# INSERT INTO test_table_1 VALUES (2, 'AAAAAAAAAP//AAAAAAAAAAAAAAAABA',
'AAAAAAAAAP//AAAAAAAAAAAAAAAABA');
INSERT 0 1
gaussdb=# INSERT INTO test_table_1 VALUES (3, 'AAAAAAAAAP//AAAAAAAAAAAAAAACAA',
'AAAAAAAAAP//AAAAAAAAAAAAAAACAA');
INSERT 0 1
gaussdb=# -- 5. listagg function
gaussdb=# SELECT LISTAGG(rwd, ';') WITHIN GROUP(ORDER BY rwd) FROM test_table_1;
                                        listagg
-------------------------------------------------------------------------------------------
 AAAAAAAAAP//AAAAAAAAAAAAAAAAAAAAAAAAAAAP//AAAAAAAAAAAAAAABAAAAAAAAAAP//
AAAAAAAAAAAAAACAA
(1 row)
gaussdb=# DROP TABLE test_table_1;
DROP TABLE
```

By default, the delimiter is empty.

```
gaussdb=# CREATE TABLE listagg_t1(a int, b interval);

gaussdb=# INSERT INTO listagg_t1 VALUES (NULL,'1 days'),(1,'2 days'),(1,'3 days'),(2,'4 days'),(2,'5
days'),(3,'6 days');

gaussdb=# SELECT a,LISTAGG(b) WITHIN GROUP(ORDER BY b) FROM listagg_t1 GROUP BY a ORDER
BY a;
 a |   listagg
---+--------------
 1 | 2 days3 days
 2 | 4 days5 days
 3 | 6 days
   | 1 day
(4 rows)

gaussdb=# DROP TABLE listagg_t1;
```

Set **distinct** to deduplicate data.

```
gaussdb=# CREATE TABLE listagg_t1(a int, b interval);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# INSERT INTO listagg_t1 VALUES(1,'4 days'),(1,'3 days'),(2,'1 days'),(1,'5 days'),(3,'2 days'),
(1,'4 days'),(NULL,'1 days'),(2,'1 days');

gaussdb=# SELECT a,listagg(distinct b,',') FROM listagg_t1 GROUP BY a ORDER BY a;
 a |       listagg
---+----------------------
 1 | 3 days,4 days,5 days
 2 | 1 day
 3 | 2 days
   | 1 day
(4 rows)

gaussdb=# DROP TABLE listagg_t1;
```

Use WITHIN GROUP(ORDER BY order-list) to sort.

```
gaussdb=# CREATE TABLE listagg_t1(a int, b interval, c int);

gaussdb=# INSERT INTO listagg_t1 VALUES(1,'4 days',4),(1,'3 days',3),(2,'1 days',1),(1,'5 days',1),(3,'2
days',1),(1,'4 days',2),(NULL,'1 days',1),(2,'1 days',2);

gaussdb=# SELECT a,listagg(distinct b,',') WITHIN GROUP(ORDER BY b) FROM listagg_t1 GROUP BY a
ORDER BY a;
 a |       listagg
---+----------------------
 1 | 3 days,4 days,5 days
 2 | 1 day
 3 | 2 days
   | 1 day
(4 rows)

gaussdb=# SELECT a,listagg(distinct b,',') WITHIN GROUP(ORDER BY c) FROM listagg_t1 GROUP BY a
ORDER BY a;
 a |       listagg
---+----------------------
 1 | 5 days,4 days,3 days
 2 | 1 day
 3 | 2 days
   | 1 day
(4 rows)

gaussdb=# DROP TABLE listagg_t1;
```

If WITHIN GROUP(ORDER BY order-list) is not used for sorting, the concatenation result may not be in order.

```
gaussdb=# CREATE TABLE listagg_t1(a int, b interval, c int);

gaussdb=# INSERT INTO listagg_t1 VALUES(1,'4 days',4),(1,'3 days',3),(2,'1 days',1),(1,'5 days',1),(3,'2
days',1),(1,'4 days',2),(NULL,'1 days',1),(2,'1 days',2);

gaussdb=# SELECT a,listagg(distinct b,',') FROM listagg_t1 GROUP BY a ORDER BY a;
 a |       listagg
---+----------------------
 1 | 3 days,4 days,5 days
 2 | 1 day
 3 | 2 days
   | 1 day
(4 rows)

gaussdb=# DROP TABLE listagg_t1;
```

When listagg is used as a window function, the **listagg** column is an ordered aggregation of the corresponding groups.

```
gaussdb=# CREATE TABLE listagg_t1(a int, b interval);

gaussdb=# INSERT INTO listagg_t1 VALUES (NULL,'1 days'),(1,'2 days'),(1,'3 days'),(2,'4 days'),(2,'5
days'),(3,'6 days');

gaussdb=# SELECT a,LISTAGG(b) WITHIN GROUP(ORDER BY b) OVER(PARTITION BY a) FROM
listagg_t1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 a |   listagg
---+--------------
 1 | 2 days3 days
 1 | 2 days3 days
 2 | 4 days5 days
 2 | 4 days5 days
 3 | 6 days
   | 1 day
(6 rows)

gaussdb=# DROP TABLE listagg_t1;
```

When listagg is used as a window function, **DISTINCT** is used for deduplication. Only the ORA-compatible mode is supported.

```
gaussdb=# CREATE TABLE listagg_t1(a int, b interval, c int);

gaussdb=# INSERT INTO listagg_t1 VALUES(1,'3 days',8),(1,'4 days',7),(2,'1 days',6),(1,'5 days',5),(3,'2
days',4),(1,'4 days',3),(NULL,'1 days',2),(2,'1 days',1);

gaussdb=# SELECT a,listagg(distinct b,',') over(partition by a) FROM listagg_t1 ORDER BY a;
 a |       listagg
---+----------------------
 1 | 3 days,4 days,5 days
 1 | 3 days,4 days,5 days
 1 | 3 days,4 days,5 days
 1 | 3 days,4 days,5 days
 2 | 1 day
 2 | 1 day
 3 | 2 days
   | 1 day
(8 rows)

gaussdb=# SELECT a,listagg(distinct b,',') WITHIN GROUP(ORDER BY b) over(partition by a) FROM
listagg_t1 ORDER BY a;
 a |       listagg
---+----------------------
 1 | 3 days,4 days,5 days
 1 | 3 days,4 days,5 days
 1 | 3 days,4 days,5 days
 1 | 3 days,4 days,5 days
 2 | 1 day
 2 | 1 day
 3 | 2 days
   | 1 day
(8 rows)

gaussdb=# SELECT a,listagg(distinct b,',') WITHIN GROUP(ORDER BY c) over(partition by a) FROM
listagg_t1 ORDER BY a;
 a |       listagg
---+----------------------
 1 | 4 days,5 days,3 days
 1 | 4 days,5 days,3 days
 1 | 4 days,5 days,3 days
 1 | 4 days,5 days,3 days
 2 | 1 day
 2 | 1 day
 3 | 2 days
   | 1 day
(8 rows)

gaussdb=# DROP TABLE listagg_t1;
```

- group_concat([DISTINCT | ALL] expression [,expression ...] [ORDER BY { expression [[ASC | DESC | USING operator] | nlssort_expression_clause] [NULLS { FIRST | LAST }] } [,...]] [SEPARATOR str_val])

  Description: The number of parameters is not fixed. Multiple columns can be concatenated. Aggregation columns are sorted based on the value of **ORDER**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

BY and concatenated into a character string using the specified separator. This function cannot be used as a window function.

- **DISTINCT**: Optional. It deduplicates the results after each row is concatenated.
- **expression**: Required. It specifies the aggregation column name or a valid column-based expression.
- **ORDER BY**: Optional. It is followed by a variable number of expressions and sorting rule. The group_concat function does not support the (ORDER BY + number) format.
- **SEPARATOR**: Optional. It is followed by a character or string. This separator is used to concatenate the expression results of two adjacent lines in a group. If it is not specified, the comma (,) is used by default.
- When both DISTINCT and ORDER BY are specified, all ORDER BY expressions must be in DISTINCT expressions. Otherwise, an error is reported.
- Parameter **group_concat_max_len** is used to limit the maximum length of **GROUP_CONCAT**. If the length exceeds the maximum, the system truncates the length. Currently, the maximum length that can be returned is 1073741823.

Return type: text

Example:

Set **separator** to **';'**.

```
gaussdb=# CREATE TABLE group_concat_t1(a int, b int);

gaussdb=# INSERT INTO group_concat_t1 VALUES (NULL,1),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT a,group_concat(b separator ';') FROM group_concat_t1 GROUP BY a ORDER BY a;
 a | group_concat
---+--------------
 1 | 2;3
 2 | 4;5
 3 | 6
   | 1
(4 rows)

gaussdb=# DROP TABLE group_concat_t1;
```

By default, the separator is a comma (,).

```
gaussdb=# CREATE TABLE group_concat_t1(a int, b int);

gaussdb=# INSERT INTO group_concat_t1 VALUES (NULL,1),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT a,group_concat(a,b) FROM group_concat_t1 GROUP BY a ORDER BY a;
 a | group_concat
---+--------------
 1 | 12,13
 2 | 24,25
 3 | 36
   |
(4 rows)

gaussdb=# DROP TABLE group_concat_t1;
```

The aggregation column is of the text character set type.

```
gaussdb=# CREATE TABLE group_concat_t1(a int, b text);

gaussdb=# INSERT INTO group_concat_t1 VALUES (NULL,'a1'),(1,'b2'),(1,'c3'),(2,'d4'),(2,'e5'),(3,'f6');

gaussdb=# SELECT a,group_concat(a,b) FROM group_concat_t1 GROUP BY a ORDER BY a;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 a | group_concat
---+--------------
 1 | 1b2,1c3
 2 | 2d4,2e5
 3 | 3f6
   |
(4 rows)

gaussdb=# DROP TABLE group_concat_t1;
```

The aggregation column is of the integer type.

```
gaussdb=# CREATE TABLE group_concat_t1(a int, b int);

gaussdb=# INSERT INTO group_concat_t1 VALUES (NULL,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,group_concat(b) FROM group_concat_t1 GROUP BY a ORDER BY a;
 a | group_concat
---+--------------
 1 | 2,3
 2 | 4,5
 3 | 6
   | 1
(4 rows)

gaussdb=# DROP TABLE group_concat_t1;
```

The aggregation column is of the floating-point type.

```
gaussdb=# CREATE TABLE group_concat_t1(a int, b float);

gaussdb=# INSERT INTO group_concat_t1 VALUES (NULL,1.11),(1,2.22),(1,3.33),(2,4.44),(2,5.55),
(3,6.66);

gaussdb=# SELECT a,group_concat(b) FROM group_concat_t1 GROUP BY a ORDER BY a;
 a | group_concat
---+--------------
 1 | 3,2,2,3
 2 | 6,4,5,4
 3 | 7,6
   | 1,1
(4 rows)

gaussdb=# DROP TABLE group_concat_t1;
```

The aggregation column is of the time type.

```
gaussdb=# CREATE TABLE group_concat_t1(a int, b timestamp);

gaussdb=# INSERT INTO group_concat_t1 VALUES (NULL,'2000-01-01'),(1,'2000-02-02'),
(1,'2000-03-03'),(2,'2000-04-04'),(2,'2000-05-05'),(3,'2000-06-06');

gaussdb=# SELECT a,group_concat(b) FROM group_concat_t1 GROUP BY a ORDER BY a;
 a |              group_concat
---+-----------------------------------------
 1 | 2000-02-02 00:00:00,2000-03-03 00:00:00
 2 | 2000-04-04 00:00:00,2000-05-05 00:00:00
 3 | 2000-06-06 00:00:00
   | 2000-01-01 00:00:00
(4 rows)

gaussdb=# DROP TABLE group_concat_t1;
```

The aggregation column is of the binary type.

```
gaussdb=# CREATE TABLE group_concat_t1(a int, b bytea);

gaussdb=# INSERT INTO group_concat_t1 VALUES (NULL,'1'),(1,'2'),(1,'3'),(2,'4'),(2,'5'),(3,'6');

gaussdb=# SELECT a,group_concat(b) FROM group_concat_t1 GROUP BY a ORDER BY a;
 a | group_concat
---+--------------
 1 | \x32,\x33
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 2 | \x34,\x35
 3 | \x36
   | \x31
(4 rows)

gaussdb=# DROP TABLE group_concat_t1;
```

The aggregation column is of the time interval type.

```
gaussdb=# CREATE TABLE group_concat_t1(a int, b interval);

gaussdb=# INSERT INTO group_concat_t1 VALUES (NULL,'1 days'),(1,'2 days'),(1,'3 days'),(2,'4 days'),
(2,'5 days'),(3,'6 days');

gaussdb=# SELECT a,group_concat(b) FROM group_concat_t1 GROUP BY a ORDER BY a;
 a | group_concat
---+---------------
 1 | 2 days,3 days
 2 | 4 days,5 days
 3 | 6 days
   | 1 day
(4 rows)

gaussdb=# DROP TABLE group_concat_t1;
```

Set **distinct** to deduplicate data.

```
gaussdb=# CREATE TABLE group_concat_t1(a int, b interval);

gaussdb=# INSERT INTO group_concat_t1 VALUES (NULL,'1 days'),(1,'2 days'),(1,'2 days'),(1,'3 days'),
(1,'3 days'),(2,'4 days'),(2,'5 days'),(3,'6 days');

gaussdb=# SELECT a,group_concat(distinct b) FROM group_concat_t1 GROUP BY a ORDER BY a;
 a | group_concat
---+---------------
 1 | 2 days,3 days
 2 | 4 days,5 days
 3 | 6 days
   | 1 day
(4 rows)

gaussdb=# DROP TABLE group_concat_t1;
```

Set **ORDER BY** to sort data.

```
gaussdb=# CREATE TABLE group_concat_t1(a int, b interval);

gaussdb=# INSERT INTO group_concat_t1 VALUES (NULL,'1 days'),(1,'2 days'),(1,'3 days'),(2,'4 days'),
(2,'5 days'),(3,'6 days');

gaussdb=# SELECT a,group_concat(b ORDER BY b desc) FROM group_concat_t1 GROUP BY a ORDER
BY a;
 a | group_concat
---+---------------
 1 | 3 days,2 days
 2 | 5 days,4 days
 3 | 6 days
   | 1 day
(4 rows)

gaussdb=# DROP TABLE group_concat_t1;
```

- wm_concat(expression)

  Description: Concatenates the expression data of a specified column or columns into a character string by commas (,) for output.

  Return type: text

  Example:

```
gaussdb=# CREATE TABLE wm_concat_t1(a int, b interval);
CREATE TABLE
gaussdb=# INSERT INTO wm_concat_t1 VALUES (NULL,'1 days'),(1,'2 days'),(1,'3 days'),(2,'4 days'),
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(2,'5 days'),(3,'6 days');
INSERT 0 6
gaussdb=# SELECT wm_concat(a) FROM wm_concat_t1;
 wm_concat
-----------
 1,1,2,2,3
(1 row)
gaussdb=# SELECT wm_concat(b) FROM wm_concat_t1;
              wm_concat
-------------------------------------------
 1 day,2 days,3 days,4 days,5 days,6 days
(1 row)
gaussdb=# DROP TABLE wm_concat_t1;
```

NOTE

The return value of this function is text, and the combined result is unordered.

- covar_pop(Y, X)

  Description: Overall covariance

  Return type: double precision

  Example:

```
gaussdb=# CREATE TABLE covar_pop_t1(a int, b int);

gaussdb=# INSERT INTO covar_pop_t1 VALUES (NULL,11),(11,21),(11,31),(21,41),(21,51),(31,61);

gaussdb=# SELECT COVAR_POP(a,b) FROM covar_pop_t1;
 covar_pop
-----------
       100
(1 row)

gaussdb=# DROP TABLE covar_pop_t1;
```

- covar_samp(Y, X)

  Description: Sample covariance

  Return type: double precision

  Example:

```
gaussdb=# CREATE TABLE covar_samp_t1(a int, b int);

gaussdb=# INSERT INTO covar_samp_t1 VALUES (NULL,11),(11,21),(11,31),(21,41),(21,51),(31,61);

gaussdb=# SELECT COVAR_SAMP(a,b) FROM covar_samp_t1;
 covar_samp
------------
        125
(1 row)

gaussdb=# DROP TABLE covar_samp_t1;
```

- stddev_pop(expression)

  Description: Overall standard difference

  Return type: DOUBLE PRECISION for floating-point arguments, otherwise NUMERIC

  Example:

```
gaussdb=# CREATE TABLE stddev_pop_t1(a int, b int);

gaussdb=# INSERT INTO stddev_pop_t1 VALUES (NULL,11),(11,21),(11,31),(21,41),(21,51),(31,61);

gaussdb=# SELECT STDDEV_POP(a) FROM stddev_pop_t1;
    stddev_pop
--------------------
```

```
 7.4833147735478828
(1 row)

gaussdb=# DROP TABLE stddev_pop_t1;
```

- stddev_samp(expression)

  Description: Sample standard deviation of the input values

  Return type: DOUBLE PRECISION for floating-point arguments, otherwise NUMERIC

  Example:

```
gaussdb=# CREATE TABLE stddev_samp_t1(a int, b int);

gaussdb=# INSERT INTO stddev_samp_t1 VALUES (NULL,11),(11,21),(11,31),(21,41),(21,51),(31,61);

gaussdb=# SELECT STDDEV_SAMP(a) FROM stddev_samp_t1;
   stddev_samp
--------------------
 8.3666002653407555
(1 row)

gaussdb=# DROP TABLE stddev_samp_t1;
```

- var_pop(expression)

  Description: Specifies the population variance of the input values (square of the population standard deviation).

  Return type: DOUBLE PRECISION for floating-point arguments, otherwise NUMERIC

  Example:

```
gaussdb=# CREATE TABLE var_pop_t1(a int, b int);

gaussdb=# INSERT INTO var_pop_t1 VALUES (NULL,11),(11,21),(11,31),(21,41),(21,51),(31,61);

gaussdb=# SELECT VAR_POP(a) FROM var_pop_t1;
      var_pop
--------------------
 56.0000000000000000
(1 row)

gaussdb=# DROP TABLE var_pop_t1;
```

- var_samp(expression)

  Description: Specifies the sample variance of the input values (square of the sample standard deviation).

  Return type: DOUBLE PRECISION for floating-point arguments, otherwise NUMERIC

  Example:

```
gaussdb=# CREATE TABLE var_samp_t1(a int, b int);

gaussdb=# INSERT INTO var_samp_t1 VALUES (NULL,11),(11,21),(11,31),(21,41),(21,51),(31,61);

gaussdb=# SELECT VAR_SAMP(a) FROM var_samp_t1;
     var_samp
--------------------
 70.0000000000000000
(1 row)

gaussdb=# DROP TABLE var_samp_t1;
```

- bit_and(expression)

  Description: The bitwise AND of all non-null input values, or null if none

  Return type: same as the parameter data type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# CREATE TABLE bit_and_t1(a int, b int);

gaussdb=# INSERT INTO bit_and_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT BIT_AND(a) FROM bit_and_t1;
 bit_and
---------
       0
(1 row)

gaussdb=# DROP TABLE bit_and_t1;
```

- bit_or(expression)

  Description: The bitwise OR of all non-null input values, or null if none

  Return type: same as the parameter data type.

  Example:

```
gaussdb=# CREATE TABLE bit_or_t1(a int, b int);

gaussdb=# INSERT INTO bit_or_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT BIT_OR(a) FROM bit_or_t1;
 bit_or
--------
      3
(1 row)

gaussdb=# DROP TABLE bit_or_t1;
```

- bool_and(expression)

  Description: Its value is **true** if all input values are **true**, otherwise **false**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT bool_and(100 <2500);
 bool_and
----------
 t
(1 row)
```

- bool_or(expression)

  Description: Its value is **true** if at least one input value is **true**, otherwise **false**.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT bool_or(100 <2500);
 bool_or
----------
 t
(1 row)
```

- corr(Y, X)

  Description: Specifies the correlation coefficient.

  Return type: double precision

  Example:

```
gaussdb=# CREATE TABLE corr_t1(a int, b int);

gaussdb=# INSERT INTO corr_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT CORR(a,b) FROM corr_t1;
      corr
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
------------------
 .944911182523068
(1 row)

gaussdb=# DROP TABLE corr_t1;
```

- every(expression)

  Description: Equivalent to **bool_and**

  Return type: Boolean

  Example:

```
gaussdb=# SELECT every(100 <2500);
 every
-------
 t
(1 row)
```

- rank(expression)

  Description: The tuples in different groups are sorted non-consecutively by **expression**.

  Return type: bigint

  Example:

```
gaussdb=# CREATE TABLE rank_t1(a int, b int);

gaussdb=# INSERT INTO rank_t1 VALUES(NULL,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,RANK() OVER(PARTITION BY a ORDER BY b) FROM rank_t1;
 a | b | rank
---+---+------
 1 | 2 |    1
 1 | 3 |    2
 2 | 4 |    1
 2 | 5 |    2
 3 | 6 |    1
   | 1 |    1
(6 rows)

gaussdb=# DROP TABLE rank_t1;
```

- regr_avgx(Y, X)

  Description: Specifies the average of the independent variable (**sum(X)/Y**).

  Return type: double precision

  Example:

```
gaussdb=# CREATE TABLE regr_t1(a int, b int);

gaussdb=# INSERT INTO regr_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT REGR_AVGX(a,b) FROM regr_t1;
 regr_avgx
-----------
         4
(1 row)

gaussdb=# DROP TABLE regr_t1;
```

- regr_avgy(Y, X)

  Description: Specifies the average of the dependent variable (**sum(Y)/X**).

  Return type: double precision

  Example:

```
gaussdb=# CREATE TABLE regr_avgy_t1(a int, b int);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# INSERT INTO regr_avgy_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT REGR_AVGY(a,b) FROM regr_avgy_t1;
 regr_avgy
-----------
       1.8
(1 row)

gaussdb=# DROP TABLE regr_avgy_t1;
```

- regr_count(Y, X)

  Description: Specifies the number of input rows in which both expressions are non-null.

  Return type: bigint

  Example:

```
gaussdb=# CREATE TABLE regr_count_t1(a int, b int);

gaussdb=# INSERT INTO regr_count_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT REGR_COUNT(a,b) FROM regr_count_t1;
 regr_count
------------
          5
(1 row)

gaussdb=# DROP TABLE regr_count_t1;
```

- regr_intercept(Y, X)

  Description: y-intercept of the least-squares-fit linear equation determined by the (X, Y) pairs

  Return type: double precision

  Example:

```
gaussdb=# CREATE TABLE regr_intercept_t1(a int, b int);

gaussdb=# INSERT INTO regr_intercept_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT REGR_INTERCEPT(b,a) FROM regr_intercept_t1;
  regr_intercept
------------------
 .785714285714286
(1 row)

gaussdb=# DROP TABLE regr_intercept_t1;
```

- regr_r2(Y, X)

  Description: Specifies the square of the correlation coefficient.

  Return type: double precision

  Example:

```
gaussdb=# CREATE TABLE regr_r2_t1(a int, b int);

gaussdb=# INSERT INTO regr_r2_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT REGR_R2(b,a) FROM regr_r2_t1;
     regr_r2
------------------
 .892857142857143
(1 row)

gaussdb=# DROP TABLE regr_r2_t1;
```

- regr_slope(Y, X)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Slope of the least-squares-fit linear equation determined by the (X, Y) pairs

Return type: double precision

Example:

```
gaussdb=# CREATE TABLE regr_slope_t1(a int, b int);

gaussdb=# INSERT INTO regr_slope_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT REGR_SLOPE(b,a) FROM regr_slope_t1;
    regr_slope
------------------
 1.78571428571429
(1 row)

gaussdb=# DROP TABLE regr_slope_t1;
```

- regr_sxx(Y, X)

  Description: **sum(Y^2) – sum(X)^2/N** (sum of squares of the independent variables)

  Return type: double precision

  Example:

```
gaussdb=# CREATE TABLE regr_sxx_t1(a int, b int);

gaussdb=# INSERT INTO regr_sxx_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT REGR_SXX(b,a) FROM regr_sxx_t1;
 regr_sxx
----------
      2.8
(1 row)

gaussdb=# DROP TABLE regr_sxx_t1;
```

- regr_sxy(Y, X)

  Description: **sum(X*Y) – sum(X) * sum(Y)/N** (sum of products of independent times dependent variable)

  Return type: double precision

  Example:

```
gaussdb=# CREATE TABLE regr_sxy_t1(a int, b int);

gaussdb=# INSERT INTO regr_sxy_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT REGR_SXY(b,a) FROM regr_sxy_t1;
 regr_sxy
----------
        5
(1 row)

gaussdb=# DROP TABLE regr_sxy_t1;
```

- regr_syy(Y, X)

  Description: **sum(Y^2) – sum(X)^2/N** ("sum of squares" of the dependent variable)

  Return type: double precision

  Example:

```
gaussdb=# CREATE TABLE regr_syy_t1(a int, b int);

gaussdb=# INSERT INTO regr_syy_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT REGR_SYY(b,a) FROM regr_syy_t1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 regr_syy
----------
       10
(1 row)

gaussdb=# DROP TABLE regr_syy_t1;
```

- stddev(expression)

  Description: Alias of **stddev_samp**

  Return type: DOUBLE PRECISION for floating-point arguments, otherwise NUMERIC

  Example:

```
gaussdb=# CREATE TABLE stddev_t1(a int, b int);

gaussdb=# INSERT INTO stddev_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT STDDEV(a) FROM stddev_t1;
        stddev
----------------------
 .83666002653407554798
(1 row)

gaussdb=# DROP TABLE stddev_t1;
```

- variance(expression,expression)

  Description: Alias of **var_samp**

  Return type: DOUBLE PRECISION for floating-point arguments, otherwise NUMERIC

  Example:

```
gaussdb=# CREATE TABLE variance_t1(a int, b int);

gaussdb=# INSERT INTO variance_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT VARIANCE(a) FROM variance_t1;
       variance
----------------------
 .70000000000000000000
(1 row)

gaussdb=# DROP TABLE variance_t1;
```

- spread()

  Description: Calculates the difference between the maximum value and minimum value in a certain period.

  Parameters: real type. For details, see **Table 1-93**.

  Return type: real

  Table 1-93 Parameters of the spread function

Parameter	Type	Description
field	Numeric	Column to be calculated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If there are fewer than two tuples in each group, the returned result is **0**. Do not use this function with the over statement.
- The spread function cannot be pushed down and requires data to be pulled from the CN for calculation. If the data volume is large, data will be flushed to disk, affecting performance.

Example:

```
gaussdb=# SELECT
gaussdb-#       SPREAD(value)
gaussdb-# FROM
gaussdb-#       (VALUES ('2019-07-12 00:00:00'::timestamptz, 1),('2019-07-12 00:01:00'::timestamptz,
2),('2019-07-12 00:02:00'::timestamptz, 3)) v(time,value);
 spread
--------
      2
(1 row)
```

- pivot_func(anyelement)

  Description: Returns the only non-null value in a column. If there are two or more non-null values, an error is reported. **value** is an input parameter and can be of any type.

  Return type: same as the input parameter type

  NOTE

  This aggregate function is mainly used inside the pivot syntax.

```
gaussdb=# CREATE TABLE pivot_func_t1(a int, b int);

gaussdb=# INSERT INTO pivot_func_t1 VALUES (NULL,11),(1,2);

gaussdb=# SELECT PIVOT_FUNC(a) FROM pivot_func_t1;
 pivot_func
------------
          1
(1 row)

gaussdb=# DROP TABLE pivot_func_t1;
```

- checksum(expression)

  Description: Returns the CHECKSUM value of all input values. This function can be used to check whether the data in the tables is the same before and after the backup, restoration, or migration of GaussDB (databases other than GaussDB are not supported). Before and after database backup, database restoration, or data migration, you need to manually run SQL commands to obtain the execution results. Compare the obtained execution results to check whether the data in the tables before and after the backup or migration is the same.

  NOTE

  - For large tables, the execution of CHECKSUM function may take a long time.
  - If the CHECKSUM values of two tables are different, it indicates that the contents of the two tables are different. Using the hash function in the CHECKSUM function may incur conflicts. There is low possibility that two tables with different contents may have the same CHECKSUM value. The same problem may occur when CHECKSUM is used for columns.
  - If the time type is timestamp, timestamptz, or smalldatetime, ensure that the time zone settings are the same when calculating the CHECKSUM value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - If the CHECKSUM value of a column is calculated and the column type can be changed to TEXT by default, set *expression* to the column name.
  - If the CHECKSUM value of a column is calculated and the column type cannot be converted to TEXT by default, set *expression* to *Column name*::**TEXT**.
  - If the CHECKSUM value of all columns is calculated, set *expression* to *Table name*::**TEXT**.

  The following types of data can be converted into the TEXT type by default: char, name, int8, int2, int1, int4, raw, pg_node_tree, float4, float8, bpchar, varchar, nvarchar2, date, datea (valid only for ORA compatibility), timestamp, timestamptz, numeric, and smalldatetime. Other types (for example, XML) need to be forcibly converted to TEXT.

  Return type: numeric

  Example:

  The following shows the CHECKSUM value of a column that can be converted to the TEXT type by default:

```
gaussdb=# CREATE TABLE checksum_t1(a int, b int);

gaussdb=# INSERT INTO checksum_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT CHECKSUM(a) FROM checksum_t1;
  checksum
-------------
 18126842830
(1 row)

gaussdb=# DROP TABLE checksum_t1;
```

  The following shows the CHECKSUM value of a column that cannot be converted to the TEXT type by default. Note that the CHECKSUM parameter is set to *Column name*::**TEXT**.

```
gaussdb=# CREATE TABLE checksum_t1(a int, b int);

gaussdb=# INSERT INTO checksum_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT CHECKSUM(a::TEXT) FROM checksum_t1;
  checksum
-------------
 18126842830
(1 row)

gaussdb=# DROP TABLE checksum_t1;
```

  The following shows the CHECKSUM value of all columns in a table. Note that the CHECKSUM parameter is set to *Table name*::**TEXT**. The table name is not modified by its schema.

```
gaussdb=# CREATE TABLE checksum_t1(a int, b int);

gaussdb=# INSERT INTO checksum_t1 VALUES (NULL,11),(1,2),(1,3),(2,4),(2,5),(3,6);

gaussdb=# SELECT CHECKSUM(checksum_t1::TEXT) FROM checksum_t1;
  checksum
-------------
 11160522226
(1 row)

gaussdb=# DROP TABLE checksum_t1;
```

- sample(value_array anyarray, N int)

  Description: Returns *N* random values for a given column.

Parameter: See **Table 1-94**.

Return type: anyelement

Table 1-94 Parameters of the sample function

Parameter	Type	Description
value_array	anyarray	Array of the queried column.
N	int	Number of random samples.

NOTE

- The value of *N* must be a positive number and the length cannot be greater than that of **value_array**.
- This function cannot be used together with the OVER clause.
- Arrays with data volumes exceeding the value range of the int8 type cannot be used.

Example:

```
gaussdb=# SELECT
gaussdb-#     sample(array_agg(value), 1)
gaussdb-# FROM
gaussdb-#     (VALUES ('2019-07-12 00:00:00'::timestamptz, 1),('2019-07-12 00:01:00'::timestamptz,
2),('2019-07-12 00:02:00'::timestamptz, 3)) v(time,value);
 sample
--------
      3
(1 row)
```

- mode() within group (order by value anyelement)

  Description: Returns the value with the highest occurrence frequency for a given column. If multiple values have the same frequency, this function returns the smallest value among these values.

  Parameter: See **Table 1-95**.

  Return type: anyelement

  Table 1-95 Parameters of the mode function

Parameter	Type	Description
value	anyelement	Queried column.

  NOTE

  - This function must be used together with the WITHIN GROUP statement. If the WITHIN GROUP statement does not exist, an error is reported. Parameters of this function are placed after ORDER BY of the group.
  - This function cannot be used together with the OVER clause.

  Example:

```
gaussdb=# SELECT
gaussdb-#     mode() within group (order by value)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb-# FROM
gaussdb-#     (VALUES ('2019-07-12 00:00:00'::timestamptz, 1),('2019-07-12 00:01:00'::timestamptz,
2),('2019-07-12 00:02:00'::timestamptz, 3)) v(time,value);
 mode
------
    1
(1 row)
```

- integral(time_array timestamptz[], value_array double precision[], unit interval)

  Description: Returns the trapezoidal area integral for a given column at a given time.

  Parameter: See **Table 1-96**.

  Return type: float

  Table 1-96 Parameters of the integral function

Parameter	Type	Description
time_array	timestamptz[]	Queried time array.
value_array	double precision[]	Array of the queried column.
unit	interval	Time unit.

  NOTE

  - The length of the time array must be the same as that of the value array. Otherwise, an error is reported.
  - This function cannot be used together with the OVER clause.
  - Arrays with data volumes exceeding the value range of the int type cannot be used.

  Example:

```
gaussdb=# SELECT
gaussdb-#     integral(array_agg(time), array_agg(value), interval '1 second')
gaussdb-# FROM
gaussdb-#     (VALUES ('2019-07-12 00:00:00'::timestamptz, 1),('2019-07-12 00:01:00'::timestamptz,
2),('2019-07-12 00:02:00'::timestamptz, 3)) v(time,value);
    integral
------------------
 239.999999999999
(1 row)
```

Aggregate Function Nesting

Description: Performs another aggregate function operation on the grouping calculation result of the aggregate function.

Generally, it can be described as follows:

```
SELECT AGG1(AGG2(column_name1)) FROM table_name GROUP BY column_name2;
```

It is equivalent to:

```
SELECT AGG1(value) FROM (SELECT AGG2(column_name1) value FROM table_name GROUP BY
column_name2);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

In the preceding information:

- **AGG1()**: outer aggregate function.
- **AGG2()**: inner aggregate function.
- **table_name**: table name.
- **column_name1** and **column_name2**: column names.
- **value**: alias of the result of the inner aggregate function.

The overall meaning can be described as follows: The grouping calculation result of the inner aggregate function AGG2() is used as the input of the outer aggregate function AGG1() for recalculation.

NOTE

1. The nested aggregate function is located between SELECT and FROM. Otherwise, it is meaningless.
2. The SELECT statement that uses a nested aggregate function contains a GROUP BY clause.
3. Only the nested aggregate functions or constant expressions can be selected together with another nested aggregate function. The constant expression does not support the CASE WHEN expression or functions logically equivalent to CASE WHEN (such as the DECODE function, NVL2 function in ORA-compatible mode, and IF function in MySQL-compatible mode).
4. The aggregate function supports only one nesting operation.
5. Currently, the following aggregate functions can be nested: avg, max, min, sum, var_pop, var_samp, variance, stddev_pop, stddev_samp, stddev, median, regr_sxx, regr_syy, regr_sxy, regr_avgx, regr_avgy, regr_r2, regr_slope, regr_intercept, covar_pop, covar_samp, corr, and listagg.
6. The return type of the inner aggregate function must comply with the parameter type of the outer aggregate function.

Example:

```
gaussdb=# CREATE TABLE test1 (id INT,val INT);
CREATE TABLE
gaussdb=# INSERT INTO test1 VALUES (1, 1);
INSERT 0 1
gaussdb=# INSERT INTO test1 VALUES (1, null);
INSERT 0 1
gaussdb=# INSERT INTO test1 VALUES (2, 10);
INSERT 0 1
gaussdb=# INSERT INTO test1 VALUES (2, 55);
INSERT 0 1

gaussdb=# SELECT SUM(MIN(val)) FROM test1 GROUP BY id;
 sum
-----
  11
(1 row)

gaussdb=# DROP TABLE test1;
DROP TABLE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.6.19 Window Functions

Window Functions

Window functions and the **OVER** clause are used together. The OVER clause is used for grouping data and sorting the elements in a group. Window functions are used for generating sequence numbers for the values in the group.

NOTE

order by in a window function must be followed by a column name. If it is followed by a number, the number is processed as a constant value and the target column is not ranked.

If a parent query contains filter criteria of a window function in a subquery, the filter criteria in the parent query can be pushed down to the subquery.

CAUTION

1. Only the <, <=, and = filter criteria of window functions in parent queries can be pushed down to subqueries.
2. The upper boundary of a restriction condition range can be a constant, constant expression, parameter, non-VOLATILE function, or non-correlated sublink.
3. Only the **ROW_NUMBER()**, **RANK()**, and **DENSE_RANK()** window functions are supported.

Example:

```
-- Create a table.
gaussdb=# CREATE TABLE t2 (c1 INT, c2 INT);
-- Import data.
gaussdb=# INSERT INTO t2 SELECT generate_series, generate_series FROM generate_series(1, 1000000);
-- Execute the query. The query is normal, no error is reported, and the execution result is correct.
gaussdb=# EXPLAIN ANALYZE SELECT nc1 FROM (
SELECT row_number() over() rid,
t2.c1 nc1
FROM t2
) WHERE rid BETWEEN 1 AND (1 + 10 - 1);
                                                  QUERY PLAN
-------------------------------------------------------------------------------------------------------------------------
Streaming (type: GATHER)  (cost=0.12..10.64 rows=5 width=4) (actual time=10.201..12.766 rows=10
loops=1)
  Node/s: All datanodes
  ->  Subquery Scan on __unnamed_subquery__  (cost=0.00..10.45 rows=5 width=4) (actual
time=[5.310,6.092]..[5.310,6.092], rows=10)
        Filter: (__unnamed_subquery__.rid >= 1)
        ->  WindowAgg  (cost=0.00..10.25 rows=16 width=4) (actual time=[5.262,5.873]..[5.262,5.873],
rows=10)
              row_number_filter: (row_number() OVER () <= 10)
              ->  Streaming(type: BROADCAST)  (cost=0.00..10.29 rows=32 width=4) (actual
time=[5.162,5.682]..[5.162,5.682], rows=11)
                    Spawn on: All datanodes
                    ->  WindowAgg  (cost=0.00..10.25 rows=16 width=4) (actual time=[1.759,1.815]..
[1.786,1.844], rows=20)
                          row_number_filter: (row_number() OVER () <= 10)
                          ->  Seq Scan on t2  (cost=0.00..13.13 rows=20 width=4) (actual time=[1.711,1.716]..
[1.743,1.747], rows=22)
 Total runtime: 13.428 ms
(12 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Clear the environment to prevent data leakage.
gaussdb=# DROP TABLE t2;
```

- RANK()

  Description: The RANK function is used for generating non-consecutive sequence numbers for the values in each group. The same values have the same sequence number.

  Return type: bigint

  Example:

```
gaussdb=# CREATE TABLE rank_t1(a int, b int);

gaussdb=# INSERT INTO rank_t1 VALUES(1,1),(1,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,RANK() OVER(PARTITION BY a ORDER BY b) FROM rank_t1;
 a | b | rank
---+---+------
 1 | 1 |    1
 1 | 1 |    1
 1 | 2 |    3
 1 | 3 |    4
 2 | 4 |    1
 2 | 5 |    2
 3 | 6 |    1
(7 rows)

gaussdb=# DROP TABLE rank_t1;
```

- ROW_NUMBER()

  Description: The ROW_NUMBER function is used for generating consecutive sequence numbers for the values in each group. The same values have different sequence numbers.

  Return type: bigint

  Example:

```
gaussdb=# CREATE TABLE row_number_t1(a int, b int);

gaussdb=# INSERT INTO row_number_t1 VALUES(1,1),(1,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,ROW_NUMBER() OVER(PARTITION BY a ORDER BY b) FROM row_number_t1;
 a | b | row_number
---+---+------------
 1 | 1 |          1
 1 | 1 |          2
 1 | 2 |          3
 1 | 3 |          4
 2 | 4 |          1
 2 | 5 |          2
 3 | 6 |          1
(7 rows)

gaussdb=# DROP TABLE row_number_t1;
```

- DENSE_RANK()

  Description: The DENSE_RANK function is used for generating consecutive sequence numbers for the values in each group. The same values have the same sequence number.

  Return type: bigint

  Example:

```
gaussdb=# CREATE TABLE dense_rank_t1(a int, b int);

gaussdb=# INSERT INTO dense_rank_t1 VALUES(1,1),(1,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT a,b,DENSE_RANK() OVER(PARTITION BY a ORDER BY b) FROM dense_rank_t1;
 a | b | dense_rank
---+---+------------
 1 | 1 |          1
 1 | 1 |          1
 1 | 2 |          2
 1 | 3 |          3
 2 | 4 |          1
 2 | 5 |          2
 3 | 6 |          1
(7 rows)

gaussdb=# DROP TABLE dense_rank_t1;
```

- PERCENT_RANK()

  Description: The PERCENT_RANK function is used for generating corresponding sequence numbers for the values in each group. That is, the function calculates the value according to the formula: Sequence number = (**rank** – 1) / (**totalrows** – 1). **rank** is the corresponding sequence number generated based on the **RANK** function for the value and **totalrows** is the total number of elements in a group.

  Return type: double precision

  Example:

```
gaussdb=# CREATE TABLE percent_rank_t1(a int, b int);

gaussdb=# INSERT INTO percent_rank_t1 VALUES(1,1),(1,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,PERCENT_RANK() OVER(PARTITION BY a ORDER BY b) FROM percent_rank_t1;
 a | b |    percent_rank
---+---+-----------------
 1 | 1 |                0
 1 | 1 |                0
 1 | 2 | .666666666666667
 1 | 3 |                1
 2 | 4 |                0
 2 | 5 |                1
 3 | 6 |                0
(7 rows)

gaussdb=# DROP TABLE percent_rank_t1;
```

- CUME_DIST()

  Description: The CUME_DIST function is used for generating accumulative distribution sequence numbers for the values in each group. That is, the function calculates the value according to the following formula: Sequence number = Number of rows preceding or peer with current row/Total rows.

  Return type: double precision

  Example:

```
gaussdb=# CREATE TABLE cume_dist_t1(a int, b int);

gaussdb=# INSERT INTO cume_dist_t1 VALUES(1,1),(1,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,CUME_DIST() OVER(PARTITION BY a ORDER BY b) FROM cume_dist_t1;
 a | b | cume_dist
---+---+-----------
 1 | 1 |        .5
 1 | 1 |        .5
 1 | 2 |       .75
 1 | 3 |         1
 2 | 4 |        .5
 2 | 5 |         1
 3 | 6 |         1
(7 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# DROP TABLE cume_dist_t1;
```

- NTILE(num_buckets integer)

  Description: The NTILE function is used for equally allocating sequential data sets to the buckets whose quantity is specified by **num_buckets** according to **num_buckets integer** and allocating the bucket number to each row. Divide the partition as evenly as possible.

  Return type: integer

  Example:

```
gaussdb=# CREATE TABLE ntile_t1(a int, b int);

gaussdb=# INSERT INTO ntile_t1 VALUES(1,1),(1,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,NTILE(2) OVER(PARTITION BY a ORDER BY b) FROM ntile_t1;
 a | b | ntile
---+---+-------
 1 | 1 |     1
 1 | 1 |     1
 1 | 2 |     2
 1 | 3 |     2
 2 | 4 |     1
 2 | 5 |     2
 3 | 6 |     1
(7 rows)

gaussdb=# DROP TABLE ntile_t1;
```

- LAG

  Description: The LAG function is used for generating lag values for the corresponding values in each group. That is, the value of the row obtained by moving forward the row corresponding to the current value by **offset** (integer) is the sequence number. If the row does not exist after the moving, the result value is the default value. If not specified, **offset** defaults to **1** and **default** to **NULL**. The value of **default** must be of the same type as **value**.

  Syntax:

```
LAG(value any [, offset integer [, default any ]])
LAG(value any ignore|respect nulls [, offset integer [, default any ]])
LAG(value any [, offset integer [, default any ]]) ignore|respect nulls
```

Figure 1-1 LAG syntax diagram

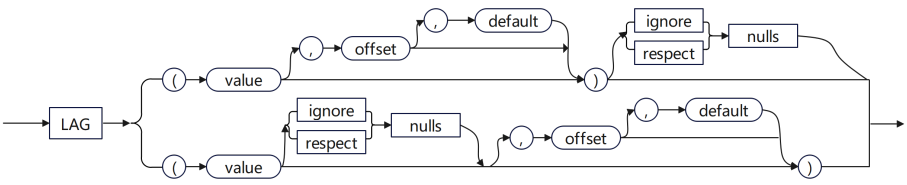


Figure 1-1 shows the syntax diagram for the LAG function. **ignore|respect nulls** determines whether NULL is included in the forward offset value. If not specified, the default value **respect nulls** is used. If **ignore nulls** is specified and the **value** is set to **NULL**, NULL is not included in the forward offset value. If **ignore nulls** is specified, the function performance deteriorates.

Return type: same as the parameter type

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example 1: Disable **ignore nulls** and set **offset** to **3** and **default** to **null**.

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE ta1 (hire_date date, last_name varchar(20), department_id int);
CREATE TABLE
gaussdb=# INSERT INTO ta1 values('07-DEC-02', 'Raphaely', 30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('24-JUL-05', 'Tobias',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('24-DEC-05', 'Baida',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('18-MAY-03', 'Khoo', 30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('15-NOV-06', 'Himuro',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('10-AUG-07', 'Colmenares',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('10-MAY-07', 'yq',  11);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('10-MAY-08', 'zi',  11);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('', 'yq1',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values(null, 'yq2',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('10-DEC-07', 'yq3',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values(null, null,  11);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values(null, null,  11);
INSERT 0 1

-- Call LAG and set offset to 3 and default to null.
gaussdb=# SELECT hire_date, last_name, department_id, lag(hire_date, 3, null)  OVER (PARTITION BY
department_id ORDER BY last_name) AS "NextHired" FROM ta1 ORDER BY department_id;
     hire_date      | last_name  | department_id |      NextHired
--------------------+------------+---------------+---------------------
 2007-05-10 00:00:00 | yq         |            11 |
 2008-05-10 00:00:00 | zi         |            11 |
                     |            |            11 |
                     |            |            11 | 2007-05-10 00:00:00
 2005-12-24 00:00:00 | Baida      |            30 |
 2007-08-10 00:00:00 | Colmenares |            30 |
 2006-11-15 00:00:00 | Himuro     |            30 |
 2003-05-18 00:00:00 | Khoo       |            30 | 2005-12-24 00:00:00
 2002-12-07 00:00:00 | Raphaely   |            30 | 2007-08-10 00:00:00
 2005-07-24 00:00:00 | Tobias     |            30 | 2006-11-15 00:00:00
                     | yq1        |            30 | 2003-05-18 00:00:00
                     | yq2        |            30 | 2002-12-07 00:00:00
 2007-12-10 00:00:00 | yq3        |            30 | 2005-07-24 00:00:00
(13 rows)
```

Example 2: Enable the **ignore nulls** and set **offset** to **3** and **default** to **'01-JAN-00'**.

```
gaussdb=# SELECT hire_date, last_name, department_id, lag(hire_date, 3, '01-JAN-00') ignore nulls
OVER (PARTITION BY department_id ORDER BY last_name) AS "NextHired"
 FROM ta1 ORDER BY department_id;
     hire_date      | last_name  | department_id |      NextHired
--------------------+------------+---------------+---------------------
 2007-05-10 00:00:00 | yq         |            11 | 2000-01-01 00:00:00
 2008-05-10 00:00:00 | zi         |            11 | 2000-01-01 00:00:00
                     |            |            11 | 2000-01-01 00:00:00
                     |            |            11 | 2000-01-01 00:00:00
 2005-12-24 00:00:00 | Baida      |            30 | 2000-01-01 00:00:00
 2007-08-10 00:00:00 | Colmenares |            30 | 2000-01-01 00:00:00
 2006-11-15 00:00:00 | Himuro     |            30 | 2000-01-01 00:00:00
 2003-05-18 00:00:00 | Khoo       |            30 | 2005-12-24 00:00:00
 2002-12-07 00:00:00 | Raphaely   |            30 | 2007-08-10 00:00:00
 2005-07-24 00:00:00 | Tobias     |            30 | 2006-11-15 00:00:00
                     | yq1        |            30 | 2003-05-18 00:00:00
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
                    | yq2        |           30 | 2003-05-18 00:00:00
 2007-12-10 00:00:00 | yq3        |           30 | 2003-05-18 00:00:00
(13 rows)

-- Drop the table.
gaussdb=# DROP TABLE ta1;
DROP TABLE
```

- LEAD

  Description: The LEAD function is used for generating leading values for the corresponding values in each group. That is, the value of the row obtained by moving backward the row corresponding to the current value by **offset** (integer) is the sequence number. If the row after the moving exceeds the total number of rows for the current group, the result value is the default value. If omitted, **offset** defaults to **1** and **default** to **NULL**. The value of **default** must be of the same type as **value**.

  Syntax:

```
LEAD(value any [, offset integer [, default any ]])
LEAD(value any ignore|respect nulls [, offset integer [, default any ]])
LEAD(value any [, offset integer [, default any ]]) ignore|respect nulls
```

Figure 1-2 LEAD syntax diagram

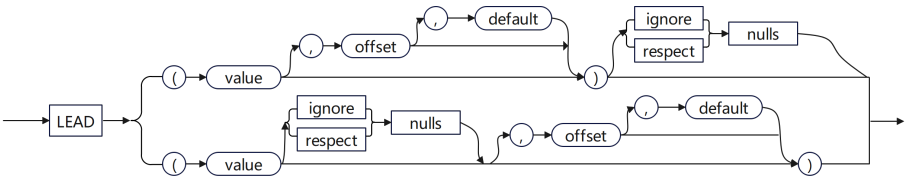


Figure 1-2 shows the syntax diagram for the LEAD function. **ignore|respect nulls** determines whether NULL is included in the backward offset value. If not specified, the default value **respect nulls** is used. If **ignore nulls** is specified and the **value** is set to **NULL**, NULL is not included in the backward offset value. If **ignore nulls** is specified, the function performance deteriorates.

Return type: same as the parameter type

Example 1: Disable **ignore nulls**, set **offset** to **2**, and do not specify **default**.

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE ta1 (hire_date date, last_name varchar(20), department_id int);
CREATE TABLE
gaussdb=# INSERT INTO ta1 values('07-DEC-02', 'Raphaely', 30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('24-JUL-05', 'Tobias',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('24-DEC-05', 'Baida',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('18-MAY-03', 'Khoo', 30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('15-NOV-06', 'Himuro',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('10-AUG-07', 'Colmenares',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('10-MAY-07', 'yq',  11);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('10-MAY-08', 'zi',  11);
INSERT 0 1
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# INSERT INTO ta1 values('', 'yq1',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values(null, 'yq2',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values('10-DEC-07', 'yq3',  30);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values(null, null,  11);
INSERT 0 1
gaussdb=# INSERT INTO ta1 values(null, null,  11);
INSERT 0 1

-- Call LEAD and set offset to 2.
gaussdb=# SELECT hire_date, last_name, department_id, lead(hire_date, 2) OVER (PARTITION BY
department_id ORDER BY last_name) AS "NextHired" FROM ta1 ORDER BY department_id;
     hire_date      | last_name  | department_id |      NextHired
--------------------+------------+---------------+---------------------
 2007-05-10 00:00:00 | yq         |            11 |
 2008-05-10 00:00:00 | zi         |            11 |
                     |            |            11 |
                     |            |            11 |
 2005-12-24 00:00:00 | Baida      |            30 | 2006-11-15 00:00:00
 2007-08-10 00:00:00 | Colmenares |            30 | 2003-05-18 00:00:00
 2006-11-15 00:00:00 | Himuro     |            30 | 2002-12-07 00:00:00
 2003-05-18 00:00:00 | Khoo       |            30 | 2005-07-24 00:00:00
 2002-12-07 00:00:00 | Raphaely   |            30 |
 2005-07-24 00:00:00 | Tobias     |            30 |
                     | yq1        |            30 | 2007-12-10 00:00:00
                     | yq2        |            30 |
 2007-12-10 00:00:00 | yq3        |            30 |
(13 rows)
```

Example 2: Enable the **ignore nulls** and set **offset** to **2** and **default** to **'01-JAN-00'**.

```
gaussdb=# SELECT hire_date, last_name, department_id, lead(hire_date, 2, '01-JAN-00') ignore nulls
OVER (PARTITION BY department_id ORDER BY last_name) AS "NextHired" FROM ta1 ORDER BY
department_id;
     hire_date      | last_name  | department_id |      NextHired
--------------------+------------+---------------+---------------------
 2007-05-10 00:00:00 | yq         |            11 | 2000-01-01 00:00:00
 2008-05-10 00:00:00 | zi         |            11 | 2000-01-01 00:00:00
                     |            |            11 | 2000-01-01 00:00:00
                     |            |            11 | 2000-01-01 00:00:00
 2005-12-24 00:00:00 | Baida      |            30 | 2006-11-15 00:00:00
 2007-08-10 00:00:00 | Colmenares |            30 | 2003-05-18 00:00:00
 2006-11-15 00:00:00 | Himuro     |            30 | 2002-12-07 00:00:00
 2003-05-18 00:00:00 | Khoo       |            30 | 2005-07-24 00:00:00
 2002-12-07 00:00:00 | Raphaely   |            30 | 2007-12-10 00:00:00
 2005-07-24 00:00:00 | Tobias     |            30 | 2000-01-01 00:00:00
                     | yq1        |            30 | 2000-01-01 00:00:00
                     | yq2        |            30 | 2000-01-01 00:00:00
 2007-12-10 00:00:00 | yq3        |            30 | 2000-01-01 00:00:00
(13 rows)

-- Drop the table.
gaussdb=# DROP TABLE ta1;
DROP TABLE
```

- FIRST_VALUE()

  Description: Returns the first value of each group.

  Return type: same as the parameter type

  Syntax:

  ```
  LAST_VALUE(value any)
  LAST_VALUE(value any IGNORE|RESPECT NULLS)
  LAST_VALUE(value any) IGNORE|RESPECT NULLS
  ```

  IGNORE|RESPECT NULLS: specifies whether to ignore NULL values. **RESPECT NULLS** is the default value. If **IGNORE NULLS** is enabled and the first value in

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

the group is NULL, the system continues to query the group until a non-NULL value is found. If the last value of the current group is still **NULL**, **NULL** is returned.

Example 1: If **IGNORE NULLS** is disabled, **NULL** values are not ignored by default.

```
gaussdb=# CREATE TABLE first_value_t1(a int, b int);

gaussdb=# INSERT INTO first_value_t1 VALUES(1, NULL),(1,1),(1,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,FIRST_VALUE(b) OVER(PARTITION BY a ORDER BY b) FROM first_value_t1;
 a | b | first_value
---+---+-------------
 1 | 1 |           1
 1 | 1 |           1
 1 | 2 |           1
 1 | 3 |           1
 1 |   |           1
 2 | 4 |           4
 2 | 5 |           4
 3 | 6 |           6
(8 rows)
gaussdb=# DROP TABLE first_value_t1;
```

Example 2: **IGNORE NULLS** is enabled.

```
gaussdb=# CREATE TABLE first_value_t1(a int, b int);

gaussdb=# INSERT INTO first_value_t1 VALUES(1, NULL),(1,1),(1,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,FIRST_VALUE(b) IGNORE NULLS OVER(PARTITION BY a ORDER BY b DESC)
FROM first_value_t1;
 a | b | first_value
---+---+-------------
 1 |   |
 1 | 3 |           3
 1 | 2 |           3
 1 | 1 |           3
 1 | 1 |           3
 2 | 5 |           5
 2 | 4 |           5
 3 | 6 |           6
(8 rows)

-- IGNORE NULLS is disabled.
gaussdb=# SELECT a,b,FIRST_VALUE(b) OVER (PARTITION BY a ORDER BY b DESC) FROM
first_value_t1;
 a | b | first_value
---+---+-------------
 1 |   |
 1 | 3 |
 1 | 2 |
 1 | 1 |
 1 | 1 |
 2 | 5 |           5
 2 | 4 |           5
 3 | 6 |           6
(8 rows)

gaussdb=# DROP TABLE first_value_t1;
```

- LAST_VALUE()

  Description: Returns the last value of each group.

  Return type: same as the parameter type

  Syntax:

  ```
  LAST_VALUE(value any)
  LAST_VALUE(value any IGNORE|RESPECT NULLS)
  LAST_VALUE(value any) IGNORE|RESPECT NULLS
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

IGNORE|RESPECT NULLS specifies whether to ignore **NULL** values. **RESPECT NULLS** is the default value. If **IGNORE NULLS** is enabled and the last value in the group is NULL, the system continues to query the group until a non-NULL value is found. If the first value of the current group is still **NULL**, **NULL** is returned.

Example 1: If **IGNORE NULLS** is disabled, **NULL** values are not ignored by default.

```
gaussdb=# CREATE TABLE last_value_t1(a int, b int);

gaussdb=# INSERT INTO last_value_t1 VALUES(1, NULL),(1,1),(1,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,LAST_VALUE(b) OVER(PARTITION BY a ORDER BY b) FROM last_value_t1;
 a | b | last_value
---+---+------------
 1 | 1 |          1
 1 | 1 |          1
 1 | 2 |          2
 1 | 3 |          3
 1 |   |
 2 | 4 |          4
 2 | 5 |          5
 3 | 6 |          6
(8 rows)
gaussdb=# DROP TABLE last_value_t1;
```

Example 2: **IGNORE NULLS** is enabled.

```
gaussdb=# CREATE TABLE last_value_t1(a int, b int);

gaussdb=# INSERT INTO last_value_t1 VALUES(1, NULL),(1,1),(1,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,LAST_VALUE(b) IGNORE NULLS OVER(PARTITION BY a ORDER BY b) FROM
last_value_t1;
 a | b | last_value
---+---+------------
 1 | 1 |          1
 1 | 1 |          1
 1 | 2 |          2
 1 | 3 |          3
 1 |   |          3
 2 | 4 |          4
 2 | 5 |          5
 3 | 6 |          6
(8 rows)

gaussdb=# DROP TABLE last_value_t1;
```

- NTH_VALUE(value any, nth integer)

  Description: Returns the *n*th row for a group. If the row does not exist, **NULL** is returned by default.

  Return type: same as the parameter type

  Syntax:

  ```
  NTH_VALUE(value any, nth integer)
  NTH_VALUE(value any, nth integer) IGNORE|RESPECT NULLS
  ```

  IGNORE|RESPECT NULLS specifies whether to ignore **NULL** values. **RESPECT NULLS** is the default value. If **IGNORE NULLS** is enabled, the system searches for non-NULL values in each group until the *n*th non-NULL value is found. If no *n*th non-NULL value is found until end of the current group, **NULL** is returned.

  Example 1: If **IGNORE NULLS** is disabled, **NULL** values are not ignored by default.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CREATE TABLE nth_value_t1(a int, b int);

gaussdb=# INSERT INTO nth_value_t1 VALUES(1, NULL),(1, NULL),(1, NULL),(1,1),(1,1),(1, 2),(1, 3),
(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,NTH_VALUE(b, 2) OVER(PARTITION BY a ORDER BY b) FROM nth_value_t1;
 a | b | nth_value
---+---+-----------
 1 | 1 |         1
 1 | 1 |         1
 1 | 2 |         1
 1 | 3 |         1
 1 |   |         1
 1 |   |         1
 1 |   |         1
 2 | 4 |
 2 | 5 |         5
 3 | 6 |
(10 rows)

gaussdb=# DROP TABLE nth_value_t1;
```

Example 2: **IGNORE NULLS** is enabled.

```
gaussdb=# CREATE TABLE nth_value_t1(a int, b int);

gaussdb=# INSERT INTO nth_value_t1 VALUES(1, NULL),(1, NULL),(1, NULL),(1,1),(1,1),(1, 2),(1, 3),
(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,NTH_VALUE(b, 2) IGNORE NULLS OVER(PARTITION BY a ORDER BY b DESC)
FROM nth_value_t1;
 a | b | nth_value
---+---+-----------
 1 |   |
 1 |   |
 1 |   |
 1 | 3 |
 1 | 2 |         2
 1 | 1 |         2
 1 | 1 |         2
 2 | 5 |
 2 | 4 |         4
 3 | 6 |
(10 rows)

-- IGNORE NULLS is disabled.
gaussdb=# SELECT a,b,NTH_VALUE(b, 2) OVER(PARTITION BY a ORDER BY b DESC) FROM
nth_value_t1;
 a | b | nth_value
---+---+-----------
 1 |   |
 1 |   |
 1 |   |
 1 | 3 |
 1 | 2 |
 1 | 1 |
 1 | 1 |
 2 | 5 |
 2 | 4 |         4
 3 | 6 |
(10 rows)

gaussdb=# DROP TABLE nth_value_t1;
```

- delta()

  Description: Returns the difference between the current row and the previous row.

  Parameter: numeric, as shown in **Table 1-97**.

  Return type: numeric

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-97 Parameters of the delta function

Parameter	Type	Description
field	Numeric	Column to be calculated.

NOTE

- When the delta() function is used as a window function, it must be used together with the over() statement, and the rows statement in the over() statement does not change the result of the delta() function. For example, the results returned by delta(value) over(order by time rows 1 preceding) and delta(value) over(order by time rows 3 preceding) are the same.
- The delta() function supports only numeric types that can be calculated; it does not support character string types.
- In the delta() function, **rows 1 preceding** in over() specifies the difference with the previous row. For other cases, the format is as follows:
  1. If the user uses the ORDER BY clause within over(), the calculation is for each sorted row compared to the first row.
  2. If the user uses the PARTITION BY clause within over(), the calculation is for the last row of each group compared to the first row of that group.
- The delta() function cannot be pushed down and requires data to be pulled from the CN for calculation. If the data volume is large, data will be flushed to disk, affecting performance.

Example:

```
gaussdb=# SELECT
gaussdb-#       delta(value) over (rows 1 preceding)
gaussdb-# FROM
gaussdb-#       (VALUES ('2019-07-12 00:00:00'::timestamptz, 1),('2019-07-12 00:01:00'::timestamptz,
2),('2019-07-12 00:02:00'::timestamptz, 3)) v(time,value);
 delta
-------
     0
     1
     1
(3 rows)
```

- RATIO_TO_REPORT(column_name)

  Description: Calculates the ratio of the value in a column to the total value in the group to which the column belongs.

  Parameter: numeric type, or any type that can be implicitly converted to the numeric type.

  Return type: For input parameter types float4 and float8, the return type is of the same value type. For other input parameter types, the return type is numeric.

  NOTE

  When RATIO_TO_REPORT(column_name) is used together with OVER(), the input parameters of OVER() support only **PARTITION BY** and **NULL**.

  Example 1:

```
gaussdb=# CREATE TABLE ratio_to_report_t1(a int, b int);

gaussdb=# INSERT INTO ratio_to_report_t1 VALUES(1,1),(1,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,RATIO_TO_REPORT(b) OVER(PARTITION BY a) FROM ratio_to_report_t1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 a | b |    ratio_to_report
---+---+-----------------------
 1 | 1 |  .14285714285714285714
 1 | 1 |  .14285714285714285714
 1 | 2 |  .28571428571428571429
 1 | 3 |  .42857142857142857143
 2 | 4 |  .44444444444444444444
 2 | 5 |  .55555555555555555556
 3 | 6 | 1.00000000000000000000
(7 rows)

gaussdb=# DROP TABLE ratio_to_report_t1;
```

Example 2: Nest this function with other functions.

```
gaussdb=# CREATE TABLE ratio_to_report_t1(a int, b int);

gaussdb=# INSERT INTO ratio_to_report_t1 VALUES(1,1),(1,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# SELECT a,b,TO_CHAR(RATIO_TO_REPORT(b) OVER(PARTITION BY a), '$999eeee') FROM
ratio_to_report_t1;
 a | b | to_char
---+---+---------
 1 | 1 |  1e-01
 1 | 1 |  1e-01
 1 | 2 |  3e-01
 1 | 3 |  4e-01
 2 | 4 |  4e-01
 2 | 5 |  6e-01
 3 | 6 |  1e+00
(7 rows)

gaussdb=# DROP TABLE ratio_to_report_t1;
```

Example 3: Call a stored procedure.

```
gaussdb=# CREATE TABLE ratio_to_report_t1(a int, b int);

gaussdb=# INSERT INTO ratio_to_report_t1 VALUES(1,1),(1,1),(1, 2),(1, 3),(2, 4),(2, 5),(3,6);

gaussdb=# CREATE OR REPLACE PROCEDURE ratio_to_report_proc IS CURSOR cur_1 IS SELECT
a,b,RATIO_TO_REPORT(b) OVER(PARTITION BY a) FROM ratio_to_report_t1;
BEGIN
FOR cur IN cur_1 LOOP
RAISE INFO '%', cur.ratio_to_report;
END LOOP;
END;
/

gaussdb=# CALL RATIO_TO_REPORT_PROC();
INFO:  .14285714285714285714
INFO:  .14285714285714285714
INFO:  .28571428571428571429
INFO:  .42857142857142857143
INFO:  .44444444444444444444
INFO:  .55555555555555555556
INFO:  1.00000000000000000000
 ratio_to_report_proc
----------------------

(1 row)

gaussdb=# DROP PROCEDURE ratio_to_report_proc;

gaussdb=# DROP TABLE ratio_to_report_t1;
```

- KEEP(DENSE_RANK { FIRST | LAST } ORDER BY expr)

  Description: Sorts data rows in a query result based on specified rules and returns the first or last group of values. This can be used in aggregate or window functions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax:

```
aggregate_function KEEP ( DENSE_RANK { FIRST | LAST }  ORDER BY expr [ DESC | ASC ] [ NULLS
{ FIRST | LAST } ]
    [, expr [ DESC | ASC ] [ NULLS { FIRST | LAST } ]]...)
OVER ( [query_partition_clause] ) ]
```

Return type: data type of the specified column in aggregate_function.

NOTE

- aggregate_function indicates an aggregate function. Currently, only MIN, MAX, SUM, AVG, COUNT, VARIANCE, and STDDEV are supported.
- The KEEP keyword is used for clearer semantics and is limited to aggregate functions and window functions.
- DENSE_RANK FIRST and DENSE_RANK LAST indicate that only rows after the minimum (FIRST) and maximum (LAST) dense ranking are aggregated, respectively.
- Vectorized plans cannot be generated for execution.

Example:

```
gaussdb=# CREATE TABLE keep_t1(a int, b int, c int, d int);

gaussdb=# INSERT INTO keep_t1 values(2, 111, 333, 10), (2, 11, 33, 10), (2, 11, 3, 1), (1, 22, 55, 10),
(1, 2, 5, 100);

gaussdb=#  SELECT MIN(d) KEEP(DENSE_RANK FIRST ORDER BY b) OVER (PARTITION BY a)
MIN_FIRST,
   MIN(b) KEEP(DENSE_RANK LAST ORDER BY b) OVER (PARTITION BY a) MIN_LAST,
   MAX(b) KEEP(DENSE_RANK FIRST ORDER BY b) OVER (PARTITION BY a) MAX_FIRST,
   MAX(b) KEEP(DENSE_RANK LAST ORDER BY b) OVER (PARTITION BY a) MAX_LAST FROM keep_t1;
 min_first | min_last | max_first | max_last
-----------+----------+-----------+----------
       100 |       22 |         2 |       22
       100 |       22 |         2 |       22
         1 |      111 |        11 |      111
         1 |      111 |        11 |      111
         1 |      111 |        11 |      111
(5 rows)
gaussdb=# DROP TABLE keep_t1;
```

- fill(value anyelement)

  Description: Performs interpolation on each group of empty data and supplements the empty values in the existing data. The rules are as follows:

  – If the current value is empty, the previous non-empty value is returned.

  – If the current data is not empty, the current value is returned.

  Parameters: See **Table 1-98**.

  Return type: anyelement

Table 1-98 Parameters of the fill function

Parameter	Type	Description
Value	anyelement	Value to be filled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The fill function must be used together with the OVER clause.
- The core algorithm is implemented through the COALESCE function. For details, see **Conditional Expression Functions**.

Example:

```
gaussdb=# SELECT
gaussdb-#    time,
gaussdb-#    CASE WHEN value is not null THEN value else fill(value) over(partition by time) END
gaussdb-# FROM
gaussdb-#    (VALUES ('2019-07-12 00:00:00'::timestamptz, 1),
gaussdb(#            ('2019-07-12 00:00:00'::timestamptz, NULL),
gaussdb(#            ('2019-07-12 00:02:00'::timestamptz, 3)) v(time,value);
         time          | fill
-----------------------+------
 2019-07-12 00:00:00+08 |    1
 2019-07-12 00:00:00+08 |    1
 2019-07-12 00:02:00+08 |    3
(3 rows)
```

1.6.20 Security Functions

Security Functions

- gs_encrypt_aes128(encryptstr,keystr)

  Description: Encrypts **encryptstr** strings using the key derived from **keystr** and returns encrypted strings. The value of **keystr** ranges from 8 to 16 bytes and contains at least three types of the following characters: uppercase letters, lowercase letters, digits, and special characters. Currently, the following types of data can be encrypted: numerals supported in the database; character type; RAW in binary type; and DATE, TIMESTAMP, and SMALLDATETIME in date/time type.

  Return type: text

  Length of the return value: At least 92 bytes and no more than (4*[*Len*/3]+68) bytes, where *Len* indicates the length of the data before encryption (unit: byte).

  Example:

  ```
  gaussdb=# SELECT gs_encrypt_aes128('MPPDB','1234@abc');
                                  gs_encrypt_aes128
  ----------------------------------------------------------------------------------------------
   OF1g3+70oeqFfyKiWlpxfYxPnpeitNc6+7nAe02Ttt37fZF8Q+bbEYhdw/YG+0c9tHKRWM6OcTzlB3HnqvX
  +1d8Bflo=
  (1 row)
  ```

  NOTE

  An encryption password is required during the execution of this function. For security purposes, the gsql tool does not record the SQL statements containing the function name in the execution history. That is, the execution history of this function cannot be found in **gsql** by paging up and down.

- gs_decrypt_aes128(decryptstr,keystr)

  Description: Decrypts **decryptstr** strings using the key derived from **keystr** and returns decrypted strings. The **keystr** used for decryption must be consistent with that used for encryption. **keystr** cannot be empty.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> This function needs to be used with the **gs_encrypt_aes128** encryption function.

Return type: text

Example:

```
gaussdb=# SELECT gs_decrypt_aes128('OF1g3+70oeqFfyKiWlpxfYxPnpeitNc6+7nAe02Ttt37fZF8Q
+bbEYhdw/YG+0c9tHKRWM6OcTzlB3HnqvX+1d8Bflo=','1234@abc');
 gs_decrypt_aes128
-------------------
 MPPDB
(1 row)
```

> **NOTE**
>
> An encryption password is required during the execution of this function. For security purposes, the gsql tool does not record the SQL statements containing the function name in the execution history. That is, the execution history of this function cannot be found in gsql by paging up and down.

- aes_encrypt(str, key_str, init_vector)

  Description: Encrypts the string **str** using the encryption password **key_str** and initialization vector **init_vector** based on the AES algorithm.

  Parameters in the command above are as follows:

  - **str**: character string to be encrypted. If **str** is set to **NULL**, the function returns **NULL**.
  - **key_str**: encryption password. If **key_str** is set to **NULL**, the function returns **NULL**. For security purposes, you are advised to use a 128-bit, 192-bit, or 256-bit secure random number as the key character string if the key length is 128 bits, 192 bits, or 256 bits (determined by the value of **block_encryption_mode**).
  - **init_vector**: An initialization variable is provided for the required block encryption mode. The length is greater than or equal to 16 bytes. Bytes greater than 16 bytes are automatically ignored. If neither **str** nor **key_str** is **NULL**, this parameter cannot be **NULL**. Otherwise, an error is reported. For security purposes, you are advised to ensure that the IV for each encryption is unique in OFB mode and that the IV for each encryption is unpredictable in CBC or CFB mode.

  Return type: text

  Example:

```
gaussdb=# SELECT aes_encrypt('huwei123','123456vfhex4dyu,vdaladhjsadad','1234567890123456');
 aes_encrypt
-------------
 u*8\x05c?0
(1 row)
```

NOTE

- This function is valid only when GaussDB is compatible with MySQL (that is, **sql_compatibility** is set to **'MYSQL'**).
- An encryption password is required during the execution of this function. For security purposes, the gsql tool does not record the SQL statements containing the function name in the execution history. That is, the execution history of this function cannot be found in gsql by paging up and down.
- Do not call this function during operations related to stored procedures, preventing the risk of sensitive information disclosure. In addition, when using the stored procedure that contains the function, you are advised to filter the parameter information of the function before providing the information for external maintenance personnel to locate the fault. Delete the logs after using them.
- Do not call the function when **debug_print_plan** is set to **on**, preventing the risk of sensitive information disclosure. You are advised to filter parameter information of the function in the log files generated when **debug_print_plan** is set to **on** before providing the log files to external maintenance engineers for fault locating. After you finish using the logs, delete them as soon as possible.
- The SQL_ASCII setting performs quite differently from other settings. If the character set of the server is SQL_ASCII, the server interprets the byte values 0 to 127 according to the ASCII standard. The byte values 128 to 255 are regarded as the characters that cannot be parsed. If this parameter is set to SQL_ASCII, no code conversion occurs. When this function calls the third-party OpenSSL library, the returned data is non-ASCII data. Therefore, when the character set of the database server is set to SQL_ASCII, the encoding of the client must also be set to SQL_ASCII. Otherwise, an error is reported. The database does not convert or verify non-ASCII characters.

- aes_decrypt(pass_str, key_str, init_vector)

  Description: Decrypts the string **str** using the decryption password **key_str** and initialization vector **init_vector** based on the AES algorithm.

  Parameters in the command above are as follows:

  – **pass_str**: character string to be decrypted. If **pass_str** is set to **NULL**, the function returns **NULL**.

  – **key_str**: decryption password. If **key_str** is set to **NULL**, the function returns **NULL**. For security purposes, you are advised to use a 128-bit, 192-bit, or 256-bit secure random number as the key character string if the key length is 128 bits, 192 bits, or 256 bits (determined by the value of **block_encryption_mode**).

  – **init_vector**: An initialization variable is provided for the required block decryption mode. The length is greater than or equal to 16 bytes. Bytes greater than 16 bytes are automatically ignored. If neither **pass_str** nor **key_str** is **NULL**, this parameter cannot be **NULL**. Otherwise, an error is reported. For security purposes, you are advised to ensure that the IV for each encryption is unique in OFB mode and that the IV for each encryption is unpredictable in CBC or CFB mode.

  Return type: text

  Example:

```
gaussdb=# SELECT
aes_decrypt(aes_encrypt('huwei123','123456vfhex4dyu,vdaladhjsadad','1234567890123456'),'123456vf
hex4dyu,vdaladhjsadad','1234567890123456');
 aes_decrypt
-------------
 huwei123
(1 row)
```

NOTE

- This function is valid only when GaussDB is compatible with MySQL (that is, **sql_compatibility** is set to **'MYSQL'**).
- A decryption password is required during the execution of this function. For security purposes, the gsql tool does not record the SQL statements containing the function name in the execution history. That is, the execution history of this function cannot be found in gsql by paging up and down.
- Do not call this function during operations related to stored procedures, preventing the risk of sensitive information disclosure. In addition, when using the stored procedure that contains the function, you are advised to filter the parameter information of the function before providing the information for external maintenance personnel to locate the fault. Delete the logs after using them.
- Do not call the function when **debug_print_plan** is set to **on**, preventing the risk of sensitive information disclosure. You are advised to filter parameter information of the function in the log files generated when **debug_print_plan** is set to **on** before providing the log files to external maintenance engineers for fault locating. After you finish using the logs, delete them as soon as possible.
- To ensure successful decryption, ensure that the values of **block_encryption_mode**, **key_str** and IV are the same as those during encryption.
- Due to encoding differences, encrypted data cannot be directly copied from the gsql client for decryption. In this scenario, the decryption result may not be the character string before encryption.
- The SQL_ASCII setting performs quite differently from other settings. If the character set of the server is SQL_ASCII, the server interprets the byte values 0 to 127 according to the ASCII standard. The byte values 128 to 255 are regarded as the characters that cannot be parsed. If this parameter is set to SQL_ASCII, no code conversion occurs. When this function calls the third-party OpenSSL library, the returned data is non-ASCII data. Therefore, when the character set of the database server is set to SQL_ASCII, the encoding of the client must also be set to SQL_ASCII. Otherwise, an error is reported. The database does not convert or verify non-ASCII characters.

- gs_digest(input_string, hash_algorithm)

  Description: Hashes the input string using the specified hash algorithm and returns a hexadecimal number.

  Parameters in the command above are as follows:

  – **input_string**: character string to be hashed. The value cannot be **NULL**.

  – **hash_algorithm**: specifies the hash algorithm. Currently, SHA-256, SHA-384, SHA-512, and SM3 are supported. Both uppercase and lowercase letters are supported. If an unsupported hash algorithm is used, an error is reported.

  Return type: text

  Example:

```
gaussdb=# SELECT pg_catalog.gs_digest('gaussdb', 'sha256');
                            gs_digest
------------------------------------------------------------------
 4dc50d746f4e04f9b446986b34a0050e358fbfb8bc1fba314c54b52a417b0b8e
(1 row)
```

- gs_password_deadline()

  Description: Indicates the number of remaining days before the password of the current user expires.

  Return type: interval

  Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT gs_password_deadline();
 gs_password_deadline
------------------------
 83 days 17:44:32.196094
(1 row)
```

- gs_password_notifytime()

  Description: Specifies the number of days prior to password expiration that a user will receive a reminder.

  Return type: int32

- gs_password_lifetime(roleName)

  Description: Displays the remaining time before the account password expires.

  Parameter: name of the account to be queried. If no parameter is passed, the current login account is used by default.

  Return type: string

  Example:

  If the account has no password expiration set, the following information is displayed:

  ```
  gaussdb=# SELECT gs_password_lifetime('user2');
   gs_password_lifetime
  ------------------------
   -
  (1 row)
  ```

  If the account has a password expiration set, the following information is displayed:

  ```
  gaussdb=# SELECT gs_password_lifetime('user1');
   gs_password_lifetime
  ------------------------
   9 days 23:38:30.509614
  (1 row)
  ```

  NOTE

  gs_password_deadline() queries the password expiration time set in **password_effect_time**, and gs_password_lifetime() queries the expiration time set for the account password.

- login_audit_messages(BOOLEAN)

  Description: Queries login information about a login user.

  Return type: tuple

  Example:

  – Check the date, time, and IP address of the last successful login.

    ```
    gaussdb=> SELECT * FROM login_audit_messages(true);
     username | database |       logintime        |    mytype     | result | client_conninfo
    ----------+----------+------------------------+---------------+--------+-----------------
     omm      | postgres | 2020-06-29 21:56:40+08 | login_success | ok     | gsql@[local]
    (1 row)
    ```

  – Check the number, date, and time of failed attempts since the previous successful login.

    ```
    gaussdb=>  SELECT * FROM login_audit_messages(false);
     username | database |       logintime        |    mytype    | result |  client_conninfo
    ----------+----------+------------------------+--------------+--------+-------------------
     omm      | postgres | 2020-06-29 21:57:55+08 | login_failed | failed | [unknown]@[local]
     omm      | postgres | 2020-06-29 21:57:53+08 | login_failed | failed | [unknown]@[local]
    (2 rows)
    ```

- login_audit_messages_pid()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Queries login information about a login user. Different from **login_audit_messages**, this function queries login information based on **backendid**. Information about subsequent logins of the same user does not alter the query result of previous logins and cannot be found using this function.

Return type: tuple

NOTE

When the thread pool is enabled, the **backendid** obtained in the same session may change due to thread switchover. As a result, the return values are different when the function is called for multiple times. You are advised not to call this function when the thread pool is enabled.

Example:

– Check the date, time, and IP address of the last successful login.

```
gaussdb=> SELECT * FROM login_audit_messages_pid(true);
 username | database |       logintime        |    mytype     | result | client_conninfo |    backendid
----------+----------+------------------------+---------------+--------+-----------------+-----------------
 omm      | postgres | 2020-06-29 21:56:40+08 | login_success | ok     | gsql@[local]    |
139823109633792
(1 row)
```

– Check the number, date, and time of failed attempts since the previous successful login.

```
gaussdb=> SELECT * FROM login_audit_messages_pid(false);
 username | database |       logintime        |    mytype    | result |  client_conninfo  |    backendid
----------+----------+------------------------+--------------+--------+-------------------+-----------------
 omm      | postgres | 2020-06-29 21:57:55+08 | login_failed | failed | [unknown]@[local] |
139823109633792
 omm      | postgres | 2020-06-29 21:57:53+08 | login_failed | failed | [unknown]@[local] |
139823109633792
(2 rows)
```

- inet_server_addr()

  Description: Displays the server IP address.

  Return type: inet

  Example:

```
gaussdb=# SELECT inet_server_addr();
 inet_server_addr
------------------
 10.10.0.13
(1 row)
```

  NOTE

  - The client IP address 10.10.0.50 and server IP address 10.10.0.13 are used as an example.
  - If a local connection is used, the value is empty.

- inet_client_addr()

  Description: Displays the client IP address.

  Return type: inet

  Example:

```
gaussdb=# SELECT inet_client_addr();
 inet_client_addr
------------------
 10.10.0.50
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The client IP address 10.10.0.50 and server IP address 10.10.0.13 are used as an example.
- If a local connection is used, the value is empty.

- gs_query_audit(starttime, endtime, audit_log)

  Description: Queries the traditional audit logs of the current CN.

  Parameters: **timestamptz starttime** and **timestamptz endtime[,audit_log]**

  Return type: record

  Table 1-99 describes the return fields.

  Table 1-99 Fields returned by gs_query_audit

Name	Type	Description
time	timestamp with time zone	Operation time
type	text	Operation
result	text	Operation result
userid	oid	User ID
username	text	Name of the user who performs the operation
database	text	Database name
client_conninfo	text	Client connection information
object_name	text	Object name
detail_info	text	Operation details
node_name	text	Node name
thread_id	text	Thread ID
local_port	text	Local port
remote_port	text	Remote port

  Example:

```
gaussdb=# SELECT * FROM gs_query_audit('2025-01-14 11:35:41','2025-01-14 11:35:42');
          time          |     type      | result | userid | username | database | client_conninfo | object_name
|                        detail_info                         |  node_name   |            thread_id
        | local_port | remote_port
------------------------+---------------+--------+--------+----------+----------+-----------------+-------------
+------------------------------------------------------------+--------------+-----------------------
---------+------------+-------------
 2025-01-14 11:35:41+08 | login_success | ok     | 10     |  omm     | postgres | gsql@[local]    |
postgres    | login db(postgres) success,the current user is:omm, SSL=off     | coordinator1 |
139779518232320@790140941594432 | 45002      | null
 2025-01-14 11:35:41+08 | user_logout   | ok     | 10     |  omm     | postgres | gsql@[local]    |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
postgres    | logout db(postgres) success                                  | coordinator1 |
139779518232320@790140941676326 | 45002      | null
 2025-01-14 11:35:41+08 | login_success | ok     | 10     |  omm     | postgres | gsql@[local]    |
postgres    | login db(postgres) success,the current user is:omm, SSL=off     | coordinator1 |
139779518232320@790140941809559 | 45002      | null
(3 rows)
```

- pg_query_audit(starttime, endtime, audit_log)

  Description: Queries the traditional audit logs of the current CN.

  Parameters: **timestamptz starttime** and **timestamptz endtime[,audit_log]**

  Return type: record

  Table 1-100 describes the return fields.

  Table 1-100 Fields returned by pg_query_audit

Name	Type	Description
time	timestamp with time zone	Operation time
type	text	Operation
result	text	Operation result
userid	oid	User ID
username	text	Name of the user who performs the operation
database	text	Database name
client_conninfo	text	Client connection information
object_name	text	Object name
detail_info	text	Operation details
node_name	text	Node name
thread_id	text	Thread ID
local_port	text	Local port
remote_port	text	Remote port

  Example:

```
gaussdb=# SELECT * FROM pg_query_audit('2025-01-14 11:35:41','2025-01-14 11:35:42');
          time          |     type      | result | userid | username | database | client_conninfo | object_name
|                       detail_info                        |  node_name   |           thread_id     | local_port |
remote_port
------------------------+---------------+--------+--------+----------+----------+-----------------+-------------
+----------------------------------------------------------------+--------------+-------------------------------
+------------+-------------
 2025-01-14 11:35:41+08 | login_success | ok     | 10     |  omm     | postgres | gsql@[local]    |
postgres    | login db(postgres) success,the current user is:omm, SSL=off     | coordinator1 |
139779518232320@790140941594432 | 45002      | null
 2025-01-14 11:35:41+08 | user_logout   | ok     | 10     |  omm     | postgres | gsql@[local]    |
postgres    | logout db(postgres) success                                  | coordinator1 |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
139779518232320@790140941676326 | 45002      | null
 2025-01-14 11:35:41+08 | login_success | ok     | 10     |  omm     | postgres | gsql@[local]    |
postgres    | login db(postgres) success,the current user is:omm, SSL=off     | coordinator1 |
139779518232320@790140941809559 | 45002      | null
(3 rows)
```

- gs_query_audit_dist(starttime, endtime)

  Description: Queries the traditional audit logs of all CNs. The return fields of this function are the same as those of the gs_query_audit function.

  Parameters: **timestamptz starttime** and **timestamptz endtime**

  Return type: record

  Example:

```
gaussdb=# SELECT * FROM gs_query_audit_dist('2025-01-14 11:35:41','2025-01-14 11:35:42');
          time          |     type      | result | userid | username | database | client_conninfo | object_name
|                       detail_info                        |  node_name   |           thread_id       | local_port |
remote_port
------------------------+---------------+--------+--------+----------+----------+-----------------+-------------
+----------------------------------------------------------------+--------------+---------------------------------
+------------+-------------
 2025-01-14 11:35:41+08 | login_success | ok     | 10     | omm      | postgres | gsql@[local]    |
postgres    | login db(postgres) success,the current user is:omm, SSL=off     | coordinator1 |
139779518232320@790140941594432 | 45002      | null
 2025-01-14 11:35:41+08 | user_logout   | ok     | 10     | omm      | postgres | gsql@[local]    |
postgres    | logout db(postgres) success                                     | coordinator1 |
139779518232320@790140941676326 | 45002      | null
 2025-01-14 11:35:41+08 | login_success | ok     | 10     | omm      | postgres | gsql@[local]    |
postgres    | login db(postgres) success,the current user is:omm, SSL=off     | coordinator1 |
139779518232320@790140941809559 | 45002      | null
 2025-01-14 11:35:41+08 | login_success | ok     | 10     | omm      | postgres | gsql@[local]    |
postgres    | login db(postgres) success,the current user is:omm, SSL=off     | coordinator2 |
140242642269952@790140941703387 | 45004      | null
 2025-01-14 11:35:41+08 | user_logout   | ok     | 10     | omm      | postgres | gsql@[local]    |
postgres    | logout db(postgres) success                                     | coordinator2 |
140242642269952@790140941784961 | 45004      | null
 2025-01-14 11:35:41+08 | login_success | ok     | 10     | omm      | postgres | coordinator1@::1 |
postgres    | login db(postgres) success,the current user is:omm, SSL=off     | coordinator2 |
140242642269952@790140941846544 | 45004      | 49014
(6 rows)
```

- pgxc_query_audit(starttime, endtime)

  Description: Queries the traditional audit logs of all CNs. The return fields of this function are the same as those of the gs_query_audit function.

  Parameters: **timestamptz starttime** and **timestamptz endtime**

  Return type: record

  Example:

```
gaussdb=# SELECT * FROM pgxc_query_audit('2025-01-14 11:35:41','2025-01-14 11:35:42');
          time          |     type      | result | userid | username | database | client_conninfo | object_name
|                       detail_info                        |  node_name   |           thread_id       | local_port |
remote_port
------------------------+---------------+--------+--------+----------+----------+-----------------+-------------
+----------------------------------------------------------------+--------------+---------------------------------
+------------+-------------
 2025-01-14 11:35:41+08 | login_success | ok     | 10     | omm      | postgres | gsql@[local]    |
postgres    | login db(postgres) success,the current user is:omm, SSL=off     | coordinator1 |
139779518232320@790140941594432 | 45002      | null
 2025-01-14 11:35:41+08 | user_logout   | ok     | 10     | omm      | postgres | gsql@[local]    |
postgres    | logout db(postgres) success                                     | coordinator1 |
139779518232320@790140941676326 | 45002      | null
 2025-01-14 11:35:41+08 | login_success | ok     | 10     | omm      | postgres | gsql@[local]    |
postgres    | login db(postgres) success,the current user is:omm, SSL=off     | coordinator1 |
139779518232320@790140941809559 | 45002      | null
 2025-01-14 11:35:41+08 | login_success | ok     | 10     | omm      | postgres | gsql@[local]    |
postgres    | login db(postgres) success,the current user is:omm, SSL=off     | coordinator2 |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
140242642269952@790140941703387 | 45004       | null
 2025-01-14 11:35:41+08 | user_logout   | ok     | 10     | omm       | postgres | gsql@[local]     |
postgres    | logout db(postgres) success                                  | coordinator2 |
140242642269952@790140941784961 | 45004       | null
 2025-01-14 11:35:41+08 | login_success | ok     | 10     | omm       | postgres | coordinator1@::1 |
postgres    | login db(postgres) success,the current user is:omm, SSL=off     | coordinator2 |
140242642269952@790140941846544 | 45004       | 49014
(6 rows)
```

- gs_query_unified_audit(starttime, endtime, audit_log)

  Description: Queries the unified audit logs of all CNs.

  Parameters: **timestamptz starttime** and **timestamptz endtime[,audit_log text]**

  Return type: record

  The following table describes the returned columns.

  Table 1-101 Unified audit log parameters

Name	Description
time	Operation time.
type	Operation type.
result	Operation result.
userid	User ID.
username	Name of the user who performs the operation.
database	Database name.
client_conninfo	Client connection information.
object_name	Operation object name.
detail_info	Operation details.
node_name	Node name.
thread_id	Thread ID.
local_port	Local port.
remote_port	Remote port.
policy_id	Unified audit policy ID.
unified_audit_type	Unified audit policy type.
unified_audit_policy	Unified audit policy information.

  Example:

```
gaussdb=# SELECT * FROM gs_query_unified_audit(sysdate-1,sysdate+1) WHERE type='audit_policy';
          time          |     type     | result | userid | username | database | client_conninfo |
object_name              |                        detail_info                         |  node_name   |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
          thread_id            | local_port | remote_port | policy_id | unified_audit_type |
unified_audit_policy
-----------------------+--------------+--------+--------+----------+----------+-----------------
+---------------------------------------+-------------------------------------------------------------------
+--------------+--
------------------------------+------------+-------------+-----------+--------------------
+-------------------------------------------------------------------
 2025-05-07 15:17:17+08 | audit_policy | ok     | 16397  | user001  | postgres | gsql@[local]     | table:
[public.table_security_auditing] | INSERT INTO table_security_auditing VALUES(1,'22');          |
coordinator1 | 1
40658670036736@799917437811493 | 45102      | 46946       | 16421     | access             | INSERT
 2025-05-07 15:17:17+08 | audit_policy | ok     | 16397  | user001  | postgres | gsql@[local]     | table:
[public.table_security_auditing] | UPDATE table_security_auditing SET name=234123 WHERE id=1;   |
coordinator1 | 1
40658670036736@799917437817363 | 45102      | 46946       | 16421     | access             | UPDATE
 2025-05-07 15:17:17+08 | audit_policy | ok     | 16397  | user001  | postgres | gsql@[local]     | table:
[public.table_security_auditing] | DELETE FROM table_security_auditing WHERE id=1;               |
coordinator1 | 1
40658670036736@799917437820418 | 45102      | 46946       | 16421     | access             | DELETE
 2025-05-07 15:17:19+08 | audit_policy | ok     | 16397  | user001  | postgres | gsql@[local]     | table:
[public.table_security_auditing] | TRUNCATE TABLE table_security_auditing;                       |
coordinator1 | 1
40658670036736@799917439207152 | 45102      | 46946       | 16421     | access             | TRUNCATE
 2025-05-07 15:17:23+08 | audit_policy | ok     | 16397  | user001  | postgres | gsql@[local]     |
public.table_security_auditing           | GRANT INSERT ON TABLE table_security_auditing TO user002;
| coordinator1 | 1
40658670036736@799917443968269 | 45102      | 46946       | 16406     | privilege          | GRANT ON
TABLE postgres.public.table_security_auditing TO user002
 2025-05-07 15:17:25+08 | audit_policy | ok     | 16397  | user001  | postgres | gsql@[local]     |
public.table_security_auditing           | REVOKE INSERT ON TABLE table_security_auditing FROM
user002; | coordinator1 | 1
40658670036736@799917445034551 | 45102      | 46946       | 16406     | privilege          | REVOKE
ON TABLE postgres.public.table_security_auditing FROM user002
(6 rows)
```

- gs_delete_audit(starttime, endtime)

  Description: Deletes audit logs in a specified period.

  Parameters: **timestamptz starttime** and **timestamptz endtime**

  Return type: void

  Example:

  ```
  gaussdb=# SELECT * FROM gs_delete_audit('2025-01-14 00:00:00','2025-01-14 00:01:00');
   gs_delete_audit
  -----------------

  (1 row)
  ```

- pg_delete_audit(starttime, endtime)

  Description: Deletes audit logs in a specified period.

  Parameters: **timestamptz starttime** and **timestamptz endtime**

  Return type: void

  Example:

  ```
  gaussdb=# SELECT * FROM pg_delete_audit('2025-01-14 00:00:00','2025-01-14 00:01:00');
   pg_delete_audit
  -----------------

  (1 row)
  ```

- alldigitsmasking()

  Description: Specifies the internal function of the masking policy, which is used to anonymize all characters.

  Parameter: col text, letter character default '0'

  Return type: text

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- creditcardmasking()

  Description: Specifies the internal function of the masking policy, which is used to anonymize all credit card information.

  Parameter: col text, letter character default 'x'

  Return type: text

- randommasking()

  Description: Specifies the internal function of the masking policy. The random policy is used.

  Parameter: col text

  Return type: text

- fullemailmasking()

  Description: Specifies the internal function of the masking policy, which is used to anonymize the text before the last period (.) except the at sign (@).

  Parameter: col text, letter character default 'x'

  Return type: text

- basicemailmasking()

  Description: Specifies the internal function of the masking policy, which is used to anonymize the text before the first at sign (@).

  Parameter: col text, letter character default 'x'

  Return type: text

- shufflemasking()

  Description: Specifies the internal function of the masking policy, which is used to sort characters out of order.

  Parameter: col text

  Return type: text

- regexpmasking()

  Description: Specifies the internal function of the masking policy, which is used to replace characters using a regular expression.

  Parameter: col text, reg text, replace_text text, pos INTEGER default 0, reg_len INTEGER default -1

  Return type: text

- gs_encrypt(encryptstr, keystr, encrypttype)

  Description: Encrypts **encryptstr** strings using **keystr** as the key and returns encrypted strings based on **encrypttype**.

  Return type: text

Parameter	Type	Description	Value Range
encryptstr	text	Data to be encrypted	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description	Value Range
keystr	text	Encryption password	The value ranges from 8 to 16 bytes and contains at least three types of the following characters: uppercase letters, lowercase letters, digits, and special characters.
encrypttype	text	Encryption/ Decryption type (case-insensitive)	aes128, sm4, aes128_cbc_sha256, aes256_cbc_sha256, aes128_gcm_sha256, aes256_gcm_sha256, and sm4_ctr_sm3

Example:

```
gaussdb=#  SELECT gs_encrypt('MPPDB', 'Asdf1234', 'sm4');
       gs_encrypt
------------------------------
 ZBzOmaGA4Bb+coyucJ0B8AkIShqc
(1 row)
```

NOTE

An encryption password is required during the execution of this function. For security purposes, the gsql tool does not record the SQL statements containing the function name in the execution history. That is, the execution history of this function cannot be found in **gsql** by paging up and down.

aes128 and **sm4** are compatible with earlier versions. The **aes128** encryption and decryption algorithm in CBC mode is used for AES-128 encryption and decryption, and **SHA1** is used for integrity check. The **sm4** encryption and decryption algorithm uses the SM4 CTR mode and does not perform integrity check.

- gs_decrypt(decryptstr, keystr, decrypttype)

  Description: Decrypts **decryptstr** strings using **keystr** as the decryption password and returns decrypted strings based on **decrypttype**. The **decrypttype** and **keystr** used for decryption must be consistent with **encrypttype** and **keystr** used for encryption. **keystr** cannot be empty.

  This function needs to be used with the **gs_encrypt** encryption function.

  Return type: text

Parameter	Type	Description	Value Range
decryptstr	text	Data to be decrypted	-
keystr	text	Decryption password	The value ranges from 8 to 16 bytes and contains at least three types of the following characters: uppercase letters, lowercase letters, digits, and special characters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description	Value Range
decrypttype	text	Encryption/Decryption type (case-insensitive)	aes128, sm4, aes128_cbc_sha256, aes256_cbc_sha256, aes128_gcm_sha256, aes256_gcm_sha256, and sm4_ctr_sm3

Example:

```
gaussdb=# SELECT gs_decrypt('ZBzOmaGA4Bb+coyucJ0B8AkIShqc', 'Asdf1234', 'sm4');
 gs_decrypt
------------
 MPPDB
(1 row)
```

NOTE

A decryption password is required during the execution of this function. For security purposes, the gsql tool does not record the SQL statements containing the function name in the execution history. That is, the execution history of this function cannot be found in gsql by paging up and down.

aes128 and **sm4** are compatible with earlier versions. The **aes128** encryption and decryption algorithm in CBC mode is used for AES-128 encryption and decryption, and **SHA1** is used for integrity check. The **sm4** encryption and decryption algorithm uses the SM4 CTR mode and does not perform integrity check.

- gs_encrypt_bytea(encryptstr, keystr, encrypttype)

  Description: Encrypts **encryptstr** strings using **keystr** as the encryption password and returns encrypted strings based on **encrypttype**.

  Return type: bytea

Parameter	Type	Description	Value Range
encryptstr	text	Data to be encrypted	-
keystr	text	Encryption password	The value ranges from 8 to 16 bytes and contains at least three types of the following characters: uppercase letters, lowercase letters, digits, and special characters.
encrypttype	text	Encryption/Decryption type (case-insensitive)	aes128_cbc_sha256, aes256_cbc_sha256, aes128_gcm_sha256, aes256_gcm_sha256, and sm4_ctr_sm3

Example:

```
gaussdb=#  SELECT gs_encrypt_bytea('MPPDB', 'Asdf1234', 'sm4_ctr_sm3');
                                gs_encrypt_bytea
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
----------------------------------------------------------------------------------------------------------------------------

\x90e286971c2c70410def0a2814af4ac44c737926458b66271d9d1547bc937395ca018d7755672fa9dc3c
dc6ec4a76001dc0e137f3bc5c8a5c51143561f1d09a848bfdebfec5e
(1 row)
```

NOTE

An encryption password is required during the execution of this function. For security purposes, the gsql tool does not record the SQL statements containing the function name in the execution history. That is, the execution history of this function cannot be found in gsql by paging up and down.

- gs_decrypt_bytea(decryptstr, keystr, decrypttype)

  Description: Decrypts **decryptstr** strings using **keystr** as the decryption password and returns decrypted strings based on **decrypttype**. The **decrypttype** and **keystr** used for decryption must be consistent with **encrypttype** and **keystr** used for encryption. **keystr** cannot be empty.

  This function needs to be used with the **gs_encrypt_bytea** encryption function.

  Return type: text

Parameter	Type	Description	Value Range
decryptstr	bytea	Data to be decrypted	-
keystr	text	Decryption password	The value ranges from 8 to 16 bytes and contains at least three types of the following characters: uppercase letters, lowercase letters, digits, and special characters.
decrypttype	text	Encryption/ Decryption type (case-insensitive)	aes128_cbc_sha256, aes256_cbc_sha256, aes128_gcm_sha256, aes256_gcm_sha256, and sm4_ctr_sm3

  Example:

```
gaussdb=# SELECT
gs_decrypt_bytea('\x90e286971c2c70410def0a2814af4ac44c737926458b66271d9d1547bc937395ca018
d7755672fa9dc3cdc6ec4a76001dc0e137f3bc5c8a5c51143561f1d09a848bfdebfec5e', 'Asdf1234',
'sm4_ctr_sm3');
 gs_decrypt_bytea
------------------
 MPPDB
(1 row)
```

NOTE

A decryption password is required during the execution of this function. For security purposes, the gsql tool does not record the SQL statements containing the function name in the execution history. That is, the execution history of this function cannot be found in gsql by paging up and down.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.6.21 Ledger Database Functions

- get_dn_hist_relhash(text, text)

  Description: Returns the hash value of table-level data in a specified tamper-proof user table. This function can be called only between distributed nodes or through EXECUTE DIRECT ON. A message indicating insufficient permission is displayed when all users call this function.

  Parameter type: text

  Return type: hash16

- ledger_hist_check(text, text)

  Description: Verifies the consistency between the hash value of table-level data in a specified tamper-proof user table and that in the corresponding history table.

  Parameter type: text

  Return type: Boolean

  Example:

  ```
  -- Create a schema.
  gaussdb=# CREATE SCHEMA ledgernsp WITH BLOCKCHAIN;
  -- Create a table.
  gaussdb=# CREATE TABLE ledgernsp.tab(a int, b text);
  -- Insert data.
  gaussdb=# INSERT INTO ledgernsp.tab values(generate_series(1, 10000), 'test');
  gaussdb=# SELECT ledger_hist_check('ledgernsp','tab');
   ledger_hist_check
  -------------------
   t
  (1 row)
  ```

- ledger_hist_repair(text, text)

  Description: Restores the hash value of the history table corresponding to the specified tamper-proof user table to be the same as that of the user table, and returns the hash difference.

  Parameter type: text

  Return type: hash16

  Example:

  ```
  gaussdb=# SELECT ledger_hist_repair('ledgernsp','tab');
   ledger_hist_repair
  --------------------
   0000000000000000
  (1 row)
  ```

- ledger_hist_archive(text, text)

  Description: Archives the history table corresponding to a specified tamper-proof user table to the **hist_back** folder in the audit log directory. If the name of a user history table contains a slash (/), the slash (/) is replaced with an underscore (_) in the name of the file generated during archiving.

  Parameter type: text

  Return type: Boolean

  Example:

  ```
  gaussdb=# EXECUTE DIRECT ON (datanode1) 'select pg_catalog.ledger_hist_archive(''ledgernsp'',
  ''tab'');';
   ledger_hist_archive
  ---------------------
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 t
(1 row)
```

- ledger_gchain_check(text, text)

  Description: Verifies the consistency between the history table hash corresponding to the specified tamper-proof user table and the relhash corresponding to the global history table.

  Parameter type: text

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT ledger_gchain_check('ledgernsp','tab');
   ledger_gchain_check
  ---------------------
   t
  (1 row)
  ```

- ledger_gchain_repair(text, text)

  Description: Restores relhash of a specified tamper-proof user table in the global history table so that the hash is the same as that in the history table, and returns the total hash value of the specified table.

  Parameter type: text

  Return type: hash16

  Example:

  ```
  gaussdb=# SELECT ledger_gchain_repair('ledgernsp','tab');
   ledger_gchain_repair
  ----------------------
   da30c1260af5be50
  (1 row)
  ```

- ledger_gchain_archive(void)

  Description: Archives global history tables to the **hist_back** folder in the audit log directory.

  Parameter type: void

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT ledger_gchain_archive();
   ledger_gchain_archive
  -----------------------
   t
  (1 row)
  -- Clean the environment.
  gaussdb=# DROP TABLE ledgernsp.tab;
  gaussdb=# DROP SCHEMA ledgernsp;
  ```

- hash16in(cstring)

  Description: Converts the input hexadecimal string into the internal hash16 format.

  Parameter type: cstring

  Return type: hash16

- hash16out(hash16)

  Description: Converts internal hash16 data to hexadecimal cstring data.

  Parameter type: hash16

  Return type: cstring

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- hash32in(cstring)

  Description: Converts the input hexadecimal string (32 characters) into the internal type hash32.

  Parameter type: cstring

  Return type: hash32
- hash32out(hash32)

  Description: Converts internal hash32 data to hexadecimal cstring data.

  Parameter type: hash32

  Return type: cstring

1.6.22 Encrypted Functions and Operators

- byteawithoutorderwithequalcolin(cstring)

  Description: Converts input data to the internal byteawithoutorderwithequalcol format.

  Parameter type: cstring

  Return type: byteawithoutorderwithequalcol
- byteawithoutorderwithequalcolout(byteawithoutorderwithequalcol)

  Description: Converts internal data of the byteawithoutorderwithequalcol type to data of the cstring type.

  Parameter type: byteawithoutorderwithequalcol

  Return type: cstring
- byteawithoutorderwithequalcolsend(byteawithoutorderwithequalcol)

  Description: Converts data of the byteawithoutorderwithequalcol type to data of the bytea type.

  Parameter type: byteawithoutorderwithequalcol

  Return type: bytea
- byteawithoutorderwithequalcolrecv(internal)

  Description: Converts data of the internal type to data of the byteawithoutorderwithequalcol type.

  Parameter type: internal

  Return type: byteawithoutorderwithequalcol
- byteawithoutorderwithequalcoltypmodin(cstring[])

  Description: Converts data of the cstring[] type to data of the byteawithoutorderwithequalcol type.

  Parameter type: cstring[]

  Return type: int4
- byteawithoutorderwithequalcoltypmodout(int4)

  Description: Converts data of the int4 type into data of the cstring type.

  Parameter type: int4

  Return type: cstring
- byteawithoutordercolin(cstring)

  Description: Converts input data to the internal byteawithoutordercolin format.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: cstring

Return type: byteawithoutordercol

- byteawithoutordercolout(byteawithoutordercol)

  Description: Converts internal data of the byteawithoutordercol type to data of the cstring type.

  Parameter type: byteawithoutordercol

  Return type: cstring

- byteawithoutordercolsend(byteawithoutordercol)

  Description: Converts data of the byteawithoutordercol type to data of the bytea type.

  Parameter type: byteawithoutordercol

  Return type: bytea

- byteawithoutordercolrecv(internal)

  Description: Converts data of the internal type to data of the byteawithoutordercol type.

  Parameter type: internal

  Return type: byteawithoutordercol

- byteawithoutorderwithequalcolcmp(byteawithoutorderwithequalcol, byteawithoutorderwithequalcol)

  Description: Compares two byteawithoutorderwithequalcol data sizes. If the first data size is smaller than the second one, **-1** is returned. If the first data size is equal to the second one, **0** is returned. If the first data size is larger than the second one, **1** is returned.

  Parameter type: byteawithoutorderwithequalcol, byteawithoutorderwithequalcol

  Return type: int4

- byteawithoutorderwithequalcolcmpbytear(byteawithoutorderwithequalcol, bytea)

  Description: Compares the byteawithoutorderwithequalcol and bytea data sizes. If the first data size is smaller than the second one, **-1** is returned. If the first data size is equal to the second one, **0** is returned. If the first data size is larger than the second one, **1** is returned.

  Parameter type: byteawithoutorderwithequalcol, bytea

  Return type: int4

- byteawithoutorderwithequalcolcmpbyteal(bytea, byteawithoutorderwithequalcol)

  Description: Compares the bytea and byteawithoutorderwithequalcol data sizes. If the first data size is smaller than the second one, **-1** is returned. If the first data size is equal to the second one, **0** is returned. If the first data size is larger than the second one, **1** is returned.

  Parameter type: bytea, byteawithoutorderwithequalcol

  Return type: int4

- byteawithoutorderwithequalcoleq(byteawithoutorderwithequalcol, byteawithoutorderwithequalcol)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Compares two byteawithoutorderwithequalcol data records. If they are the same, **true** is returned. Otherwise, **false** is returned.

Parameter type: byteawithoutorderwithequalcol, byteawithoutorderwithequalcol

Return type: Boolean

- byteawithoutorderwithequalcoleqbyteal(bytea, byteawithoutorderwithequalcol)

  Description: Compares the bytea and byteawithoutorderwithequalcol data records. If they are the same, **true** is returned. Otherwise, **false** is returned.

  Parameter type: bytea, byteawithoutorderwithequalcol

  Return type: Boolean

- byteawithoutorderwithequalcoleqbytear(byteawithoutorderwithequalcol, bytea)

  Description: Compares the byteawithoutorderwithequalcol and bytea data records. If they are the same, **true** is returned. Otherwise, **false** is returned.

  Parameter type: byteawithoutorderwithequalcol, bytea

  Return type: Boolean

- byteawithoutorderwithequalcolne(byteawithoutorderwithequalcol, byteawithoutorderwithequalcol)

  Description: Compares two byteawithoutorderwithequalcol data records. If they are different, **true** is returned. Otherwise, **false** is returned.

  Parameter type: byteawithoutorderwithequalcol, byteawithoutorderwithequalcol

  Return type: Boolean

- byteawithoutorderwithequalcolnebyteal(bytea, byteawithoutorderwithequalcol)

  Description: Compares the bytea and byteawithoutorderwithequalcol data records. If they are the same, **true** is returned. Otherwise, **false** is returned.

  Parameter type: bytea, byteawithoutorderwithequalcol

  Return type: Boolean

- byteawithoutorderwithequalcolnebytear(byteawithoutorderwithequalcol, bytea)

  Description: Compares the byteawithoutorderwithequalcol and bytea data records. If they are the same, **true** is returned. Otherwise, **false** is returned.

  Parameter type: byteawithoutorderwithequalcol, bytea

  Return type: Boolean

- hll_hash_byteawithoutorderwithequalcol(byteawithoutorderwithequalcol)

  Description: Returns the hll hash value of byteawithoutorderwithequalcol.

  Parameter type: byteawithoutorderwithequalcol

  Return type: hll_hashval

- tee_sortsupport(internal)

  Description: Function used to support sorting, which is internally called during sorting and cannot be called by users.

  Parameter type: internal

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: void

- teegtsel(internal, oid, internal, integer)

  Description: Selectivity of ciphertext greater than and greater than or equal to. The default value **0.3333333333333333** is returned. It cannot be called by common users.

  Parameter type: internal, oid, internal, integer

  Return type: double precision

- teeltsel(internal, oid, internal, integer)

  Description: Selectivity of ciphertext less than and less than or equal to. The default value **0.3333333333333333** is returned. It cannot be called by common users.

  Parameter type: internal, oid, internal, integer

  Return type: double precision

- teelikesel(internal, oid, internal, integer)

  Description: Selectivity of ciphertext fuzzy match. The default value **0.005** is returned. It cannot be called by common users.

  Parameter type: internal, oid, internal, integer

  Return type: double precision

- teenlikesel(internal, oid, internal, integer)

  Description: Selectivity of ciphertext non-fuzzy-match. The default value **0.005** is returned. It cannot be called by common users.

  Parameter type: internal, oid, internal, integer

  Return type: double precision

- sum(byteawithoutorderwithequalcol)

  Description: Aggregates the ciphertext sum.

  Parameter type: byteawithoutorderwithequalcol

  Return type: byteawithoutorderwithequalcol

- avg(byteawithoutorderwithequalcol)

  Description: Aggregates the ciphertext average value.

  Parameter type: byteawithoutorderwithequalcol

  Return type: byteawithoutorderwithequalcol

- min(byteawithoutorderwithequalcol)

  Description: Aggregates the ciphertext minimum value.

  Parameter type: byteawithoutorderwithequalcol

  Return type: byteawithoutorderwithequalcol

- max(byteawithoutorderwithequalcol)

  Description: Aggregates the ciphertext maximum value.

  Parameter type: byteawithoutorderwithequalcol

  Return type: byteawithoutorderwithequalcol

- stddev_samp(byteawithoutorderwithequalcol)

  Description: Aggregates the standard deviation of ciphertext samples.

  Parameter type: byteawithoutorderwithequalcol

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: byteawithoutorderwithequalcol

- stddev_pop(byteawithoutorderwithequalcol)

  Description: Aggregates the total standard deviation of ciphertext.

  Parameter type: byteawithoutorderwithequalcol

  Return type: byteawithoutorderwithequalcol

- var_samp(byteawithoutorderwithequalcol)

  Description: Aggregates the variance of ciphertext samples.

  Parameter type: byteawithoutorderwithequalcol

  Return type: byteawithoutorderwithequalcol

- var_pop(byteawithoutorderwithequalcol)

  Description: Aggregates the total variance of ciphertext.

  Parameter type: byteawithoutorderwithequalcol

  Return type: byteawithoutorderwithequalcol

- tee_trans(byteawithoutorderwithequalcol, byteawithoutorderwithequalcol)

  Description: Aggregates the internal processing function for ciphertext. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, byteawithoutorderwithequalcol

  Return type: byteawithoutorderwithequalcol

- tee_collect(byteawithoutorderwithequalcol, byteawithoutorderwithequalcol)

  Description: Aggregates the internal processing function for ciphertext. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, byteawithoutorderwithequalcol

  Return type: byteawithoutorderwithequalcol

- tee_final(byteawithoutorderwithequalcol)

  Description: Aggregates the internal processing function for ciphertext. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_deterministic(text, oid)

  Description: Encrypts the plaintext of the TEXT type to the encrypted equivalent structure. The key OID is the value of **column_key_distributed_id** in the gs_column_keys system catalog. This function is not supported in the current version.

  Parameter type: text, oid

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_deterministic(int1, oid)

  Description: Encrypts the plaintext of the int1 type to the encrypted equivalent structure. The key OID is the value of **column_key_distributed_id** in the gs_column_keys system catalog. This function is not supported in the current version.

  Parameter type: int1, oid

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: byteawithoutorderwithequalcol

- ce_encrypt_deterministic(int2, oid)

  Description: Encrypts the plaintext of the int2 type to the encrypted equivalent structure. The key OID is the value of **column_key_distributed_id** in the gs_column_keys system catalog. This function is not supported in the current version.

  Parameter type: int2, oid

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_deterministic(int4, oid)

  Description: Encrypts the plaintext of the int4 type to the encrypted equivalent structure. The key OID is the value of **column_key_distributed_id** in the gs_column_keys system catalog. This function is not supported in the current version.

  Parameter type: int4, oid

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_deterministic(int8, oid)

  Description: Encrypts the plaintext of the int8 type to the encrypted equivalent structure. The key OID is the value of **column_key_distributed_id** in the gs_column_keys system catalog. This function is not supported in the current version.

  Parameter type: int8, oid

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_deterministic(float4, oid)

  Description: Encrypts the plaintext of the float4 type to the encrypted equivalent structure. The key OID is the value of **column_key_distributed_id** in the gs_column_keys system catalog. This function is not supported in the current version.

  Parameter type: float4, oid

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_deterministic(float8, oid)

  Description: Encrypts the plaintext of the float8 type to the encrypted equivalent structure. The key OID is the value of **column_key_distributed_id** in the gs_column_keys system catalog. This function is not supported in the current version.

  Parameter type: float8, oid

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_deterministic(numeric, oid)

  Description: Encrypts the plaintext of the numeric type to the encrypted equivalent structure. The key OID is the value of **column_key_distributed_id** in the gs_column_keys system catalog. This function is not supported in the current version.

  Parameter type: numeric, oid

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_deterministic(clob, oid)

  Description: Encrypts the plaintext of the CLOB type to the encrypted equivalent structure. The key OID is the value of **column_key_distributed_id**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

in the gs_column_keys system catalog. This function is not supported in the current version.

Parameter type: clob, oid

Return type: byteawithoutorderwithequalcol

- ce_encrypt_deterministic(varchar, oid)

  Description: Encrypts the plaintext of the varchar type to the encrypted equivalent structure. The key OID is the value of **column_key_distributed_id** in the gs_column_keys system catalog. This function is not supported in the current version.

  Parameter type: varchar, oid

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_deterministic(nvarchar2, oid)

  Description: Encrypts the plaintext of the nvarchar2 type to the encrypted equivalent structure. The key OID is the value of **column_key_distributed_id** in the gs_column_keys system catalog. This function is not supported in the current version.

  Parameter type: nvarchar2, oid

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_deterministic(bpchar, oid)

  Description: Encrypts the plaintext of the bpchar type to the encrypted equivalent structure. The key OID is the value of **column_key_distributed_id** in the gs_column_keys system catalog. This function is not supported in the current version.

  Parameter type: bpchar, oid

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_deterministic(bytea, oid)

  Description: Encrypts the plaintext of the bytea type to the encrypted equivalent structure. The key OID is the value of **column_key_distributed_id** in the gs_column_keys system catalog. This function is not supported in the current version.

  Parameter type: bytea, oid

  Return type: byteawithoutorderwithequalcol

- ce_decrypt_deterministic(byteawithoutorderwithequalcol, oid)

  Description: Decrypts the ciphertext of the encrypted equivalent structure to the plaintext data of the text type. The key OID is the value of **column_key_distributed_id** in the gs_column_keys system catalog. This function is not supported in the current version.

  Parameter type: byteawithoutorderwithequalcol, oid

  Return type: text

NOTE

The following functions starting with ce_encrypt/ce_decrypt exist in the pg_catalog namespace.

- ce_encrypt_text(text, int4, boolean, int4, internal)

  Description: Encryption function for fully-encrypted type conversion. It cannot be called by common users.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: text, int4, Boolean, int4, internal

Return type: byteawithoutorderwithequalcol

- ce_encrypt_varchar(varchar, int4, boolean, int4, internal)

  Description: Encryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: varchar, int4, Boolean, int4, internal

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_nvarchar2(nvarchar2, int4, boolean, int4, internal)

  Description: Encryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: nvarchar2, int4, Boolean, int4, internal

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_bpchar(bpchar, int4, boolean, int4, internal)

  Description: Encryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: bpchar, int4, Boolean, int4, internal

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_int1(int1, int4, boolean, int4, internal)

  Description: Encryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: int1, int4, Boolean, int4, internal

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_int2(int2, int4, boolean, int4, internal)

  Description: Encryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: int2, int4, Boolean, int4, internal

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_int4(int4, int4, boolean, int4, internal)

  Description: Encryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: int4, int4, Boolean, int4, internal

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_int8(int8, int4, boolean, int4, internal)

  Description: Encryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: int8, int4, Boolean, int4, internal

  Return type: byteawithoutorderwithequalcol

- ce_encrypt_float4(float4, int4, boolean, int4, internal)

  Description: Encryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: float4, int4, Boolean, int4, internal

  Return type: byteawithoutorderwithequalcol

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- ce_encrypt_float8(float8, int4, boolean, int4, internal)

  Description: Encryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: float8, int4, Boolean, int4, internal

  Return type: byteawithoutorderwithequalcol
- ce_encrypt_numeric(numeric, int4, boolean, int4, internal)

  Description: Encryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: numeric, int4, Boolean, int4, internal

  Return type: byteawithoutorderwithequalcol
- ce_encrypt_bytea(bytea, int4, boolean, int4, internal)

  Description: Encryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: bytea, int4, Boolean, int4, internal

  Return type: byteawithoutorderwithequalcol
- ce_encrypt_clob(clob, int4, boolean, int4, internal)

  Description: Encryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: clob, int4, Boolean, int4, internal

  Return type: byteawithoutorderwithequalcol
- ce_decrypt_text(byteawithoutorderwithequalcol, int4, boolean, int4, internal)

  Description: Decryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, int4, Boolean, int4, internal

  Return type: text
- ce_decrypt_varchar(byteawithoutorderwithequalcol, int4, boolean, int4, internal)

  Description: Decryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, int4, Boolean, int4, internal

  Return type: varchar
- ce_decrypt_nvarchar2(byteawithoutorderwithequalcol, int4, boolean, int4, internal)

  Description: Decryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, int4, Boolean, int4, internal

  Return type: nvarchar2
- ce_decrypt_bpchar(byteawithoutorderwithequalcol, int4, boolean, int4, internal)

  Description: Decryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, int4, Boolean, int4, internal

  Return type: bpchar

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- ce_decrypt_int1(byteawithoutorderwithequalcol, int4, boolean, int4, internal)

  Description: Decryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, int4, Boolean, int4, internal

  Return type: int1
- ce_decrypt_int2(byteawithoutorderwithequalcol, int4, boolean, int4, internal)

  Description: Decryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, int4, Boolean, int4, internal

  Return type: int2
- ce_decrypt_int4(byteawithoutorderwithequalcol, int4, boolean, int4, internal)

  Description: Decryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, int4, Boolean, int4, internal

  Return type: int4
- ce_decrypt_int8(byteawithoutorderwithequalcol, int4, boolean, int4, internal)

  Description: Decryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, int4, Boolean, int4, internal

  Return type: int8
- ce_decrypt_float4(byteawithoutorderwithequalcol, int4, boolean, int4, internal)

  Description: Decryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, int4, Boolean, int4, internal

  Return type: float4
- ce_decrypt_float8(byteawithoutorderwithequalcol, int4, boolean, int4, internal)

  Description: Decryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, int4, Boolean, int4, internal

  Return type: float8
- ce_decrypt_numeric(byteawithoutorderwithequalcol, int4, boolean, int4, internal)

  Description: Decryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, int4, Boolean, int4, internal

  Return type: numeric
- ce_decrypt_bytea(byteawithoutorderwithequalcol, int4, boolean, int4, internal)

  Description: Decryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, int4, Boolean, int4, internal

  Return type: bytea

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- ce_decrypt_clob(byteawithoutorderwithequalcol, int4, boolean, int4, internal)

  Description: Decryption function for fully-encrypted type conversion. It cannot be called by common users.

  Parameter type: byteawithoutorderwithequalcol, int4, Boolean, int4, internal

  Return type: CLOB

Examples

Encrypted equality functions such as byteawithoutorderwithequalcolin and byteawithoutorderwithequalcolout are read/write format conversion functions such as in, out, send, and recv specified by the data type byteawithoutorderwithequalcol in the database kernel. For details, see the byteain and byteaout functions of the bytea type. However, the local CEK is verified, and the function can be successfully executed only when the encrypted column contains a cekoid that exists on the localhost.

```
-- For example, use an encrypted client to connect to the database and create an encrypted table int_type.
int_col2 is the encrypted column.
gaussdb=# \key_info keyType=user_token,password=*********
gaussdb=# CREATE CLIENT MASTER KEY MyCMK WITH (KEY_STORE = user_token, ALGORITHM =
AES_256_CBC);
gaussdb=# CREATE COLUMN ENCRYPTION KEY cek1 WITH VALUES (CLIENT_MASTER_KEY = MyCMK,
ALGORITHM = AEAD_AES_256_CBC_HMAC_SHA256);
gaussdb=# CREATE TABLE IF NOT EXISTS int_type (
id INT,
int_col1 TINYINT,
int_col2 smallint ENCRYPTED WITH (COLUMN_ENCRYPTION_KEY = cek1, ENCRYPTION_TYPE =
DETERMINISTIC)
);
CREATE TABLE
gaussdb=# INSERT INTO int_type VALUES (2, 100, 200);
INSERT 0 1

-- Use a non-encrypted client to connect to the database and query the ciphertext of the encrypted column.
gaussdb=# SELECT int_col2 FROM int_type;
                                                
```

```
gaussdb=# SELECT * FROM
byteawithoutorderwithequalcolsend('\x907219912381298461289346129'::byteawithoutorderwithequalcol);
ERROR:  cek with OID 596711794 not found
LINE 1: SELECT * FROM byteawithoutorderwithequalcolsend('\x907219912...
                                                        ^
gaussdb=# SELECT * FROM byteawithoutordercolout('\x907219019999999999999123812984612893461 29');
ERROR:  cek with OID 2566986098 not found
LINE 1: SELECT * FROM byteawithoutordercolout('\x9072190199999999999...

                                              ^
gaussdb=# SELECT * FROM
byteawithoutorderwithequalcolsend('\x907219019999999999999123812984612893461 29'::byteawithoutorde
rwithequalcol);
ERROR:  cek with OID 2566986098 not found
LINE 1: SELECT * FROM byteawithoutorderwithequalcolsend('\x907219019...
                                                        ^
```

1.6.23 Set Return Functions

NOTE

- If the input parameters of any function are set return functions (SRFs), expressions in the input parameter list of that function will be verified, and an error will be reported upon detecting multiple SRFs.

  Taking **SELECT div(srf(n,1),srf(m,2));** as an example:

- If the target list of the SELECT statement includes **srf**, multiple rows of data will be returned.
- If the target list of the SELECT statement includes multiple layers of nested SRFs, such as **srf(n, srf(m, 1))**, $n \times m$ rows of data will be returned. Even if strict functions are utilized, they will not be optimized as long as they are not SRFs.

  For example, **SELECT srf(n, div(srf(m, 1), NULL));div** is a strict function. If the input parameters of the function do not include **srf** but include **NULL**, the function will return **NULL**. If the input parameters include **srf**, the strict function will not be optimized and will return *m* rows of **NULL**.
- If the input parameters of an SRF include an expression whose return value is **NULL**, the SRF will return 0 rows if it is strict, and return the designated number of rows of **NULL** if it is not strict.

  For example, **srf(10, 'a')** indicates that 10 rows of **'a'** need to be returned. If **srf(10, NULL)** is passed, a strict SRF will return 0 rows, whereas a non-strict SRF will return 10 rows of **NULL**.

Series Generating Functions

- generate_series(start, stop)

  Description: Generates a series of values, from **start** to **stop** with a step size of one.

  Parameter types: int, bigint, and numeric.

  Return types: setof int, setof bigint, and setof numeric (same as the parameter types).

- generate_series(start, stop, step)

  Description: Generates a series of values, from **start** to **stop** with a step size of **step**.

  Parameter types: int, bigint, and numeric.

  Return types: setof int, setof bigint, and setof numeric (same as the parameter types).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- generate_series(start, stop, step interval)

  Description: Generates a series of values, from **start** to **stop** with a step size of **step**.

  Parameter type: timestamp or timestamp with time zone.

  Return type: setof timestamp or setof timestamp with time zone (same as the parameter type).

When **step** is positive, zero rows are returned if **start** is greater than **stop**. Conversely, when **step** is negative, zero rows are returned if **start** is less than **stop**. Zero rows are also returned for **NULL** inputs. It is an error for **step** to be zero.

Example:

```
gaussdb=# SELECT * FROM generate_series(2,4);
 generate_series
-----------------
               2
               3
               4
(3 rows)

gaussdb=# SELECT * FROM generate_series(5,1,-2);
 generate_series
-----------------
               5
               3
               1
(3 rows)

gaussdb=# SELECT * FROM generate_series(4,3);
 generate_series
-----------------
(0 rows)

-- This example applies to the date-plus-integer operator.
gaussdb=# SELECT current_date + s.a AS dates FROM generate_series(0,14,7) AS s(a);
   dates
------------
 2017-06-02
 2017-06-09
 2017-06-16
(3 rows)

gaussdb=# SELECT * FROM generate_series('2008-03-01 00:00'::timestamp, '2008-03-04 12:00', '10 hours');
   generate_series
---------------------
 2008-03-01 00:00:00
 2008-03-01 10:00:00
 2008-03-01 20:00:00
 2008-03-02 06:00:00
 2008-03-02 16:00:00
 2008-03-03 02:00:00
 2008-03-03 12:00:00
 2008-03-03 22:00:00
 2008-03-04 08:00:00
(9 rows)
```

- gs_search_function_with_name(funcname)

  Description: Obtains the OID of the funcname function.

  Parameter type: cstring.

  Return type: setof oid.

  ```
  -- Basic usage
  CREATE OR REPLACE FUNCTION proc_plpgsql(a int,b int, c int)
  RETURNS int AS $BODY$
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DECLARE
BEGIN
RETURN $1 + $2;
END;
$BODY$ language plpgsql;
CREATE OR REPLACE FUNCTION proc_plpgsql(int,int)
RETURNS int AS $BODY$
DECLARE
BEGIN
RETURN $1 + $2;
END;
$BODY$ language plpgsql;
gaussdb=# SELECT gs_search_function_with_name('proc_plpgsql');
 gs_search_function_with_name
------------------------------
                        16776
                        24576
(2 rows)
```

Index Generating Functions

- generate_subscripts(array anyarray, dim int)

  Description: Generates a series comprising the given array's indexes.

  Return type: setof int.

- generate_subscripts(array anyarray, dim int, reverse boolean)

  Description: Generates a series comprising the given array's indexes. When **reverse** is true, the series is returned in reverse order.

  Return type: setof int.

generate_subscripts is a function that generates the set of valid indexes for the specified dimension of the given array. Zero rows are returned for arrays that do not have the requested dimension, or for NULL arrays (but valid indexes are returned for NULL array elements). Example:

```
-- Basic usage
gaussdb=# SELECT generate_subscripts('{NULL,1,NULL,2}'::int[], 1) AS s;
 s
---
 1
 2
 3
 4
(4 rows)
-- Unnest a 2D array.
gaussdb=# CREATE OR REPLACE FUNCTION unnest2(anyarray)
RETURNS SETOF anyelement AS $$
SELECT $1[i][j]
   FROM generate_subscripts($1,1) g1(i),
        generate_subscripts($1,2) g2(j);
$$ LANGUAGE sql IMMUTABLE;

gaussdb=# SELECT * FROM unnest2(ARRAY[[1,2],[3,4]]);
 unnest2
---------
       1
       2
       3
       4
(4 rows)

-- Delete the function.
gaussdb=# DROP FUNCTION unnest2;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.6.24 Overloaded Query Functions

- gs_search_function_with_name_and_arg(funcname_and_argtypes cstring)

  Description: Returns the information about the overloaded function that is actually called based on the function name and input parameter type.

  Return type: text

  NOTE

  - This function is supported only in Oracle-compatible mode.
  - To query the overloading information of a function whose input parameter contains **NULL**, replace **NULL** with **UNKNOWN**.
  - If the input parameter of the function is a window function, add **over()** at the end.
  - Some functions are implemented using syntax keywords and cannot be queried using this function. For details about the mapping, see **Table 1-102**.
  - If the function to be searched for is changed or the function selection logic is changed, the result of the function may change. For details about the function selection logic, see **Functions**.

Table 1-102 Internal function calling relationship of syntax functions

Syntax Function	Internal Function
CURRENT_CATALOG	current_database()
CURRENT_DATE	The matching priority is from top to bottom. • When **a_format_date_timestamp** is set to **on** and **mapping_date_to_datea** is set to **off**, pg_systimestamp()::timestamp(0) is used. • When **mapping_date_to_datea** is set to **on**, pg_systimestamp()::datea is used. • When **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s2** or later version, 'now'::text::timestamp(0) is used. • In other scenarios, text_date('now'::text) is used.
CURRENT_ROLE	current_user()
CURRENT_SCHEMA	current_schema()
CURRENT_TIME[(n)]	'now'::text::timetz[(n)]

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax Function	Internal Function
CURRENT_TIMESTAMP	• For CURRENT_TIMESTAMP, pg_systimestamp() is used. • CURRENT_TIMESTAMP(int) – When **a_format_date_timestamp** is set to **on**, pg_systimestamp()::timestamptz(int) is used. – When **a_format_date_timestamp** is set to **off**, 'now'::text::timestamptz(int) is used. • For CURRENT_TIMESTAMP(float), current_timestamp(numeric) is used only when **a_format_version** is set to **'10c'** and **a_format_dev_version** is set to **'s2'** or later version.
CURRENT_USER	current_user()
DBTIMEZONE	Supported only when **a_format_version** is set to **'10c'** and **a_format_dev_version** is set to **'s2'** or later version. In this case, dbtimezone() is used.
DECODE	• If the number of parameters is 2, DECODE(A, B) corresponds to decode(A, B). • If the number of parameters is greater than 2, a case is constructed. For details, see **UNION, CASE, and Related Constructs**.
EXTRACT (field FROM source)	date_part(field, source)
GROUP_CONCAT(arg1[, …] SEPARATOR s)	group_concat(s, arg1[, …])
LOCALTIME[(n)]	'now'::text::time[(n)]

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax Function	Internal Function
LOCALTIMESTAMP	• LOCALTIMESTAMP – When **a_format_date_timestamp** is set to **on**, pg_systimestamp()::timestamp is used. – When **a_format_date_timestamp** is set to **off**, 'now'::text::timestamp is used. • LOCALTIMESTAMP(int) – When **a_format_date_timestamp** is set to **on**, pg_systimestamp()::timestamp(int) is used. – When **a_format_date_timestamp** is set to **off**, 'now'::text::timestamp(int) is used. • LOCALTIMESTAMP(float) is supported only when **a_format_version** is set to **'10c'** and **a_format_dev_version** is set to **'s2'** or a later version. – When **a_format_date_timestamp** is set to **on**, current_timestamp(numeric)::timestamp is used. – When **a_format_date_timestamp** is set to **off**, there is no mapping.
NOW()	now()
OVERLAY	• OVERLAY(A PLACING B FROM C FOR D) corresponds to overlay(A, B, C, D). • OVERLAY(A PLACING B FROM C) corresponds to overlay(A, B, C).
POSITION (A IN B)	position(B, A)
SESSION_USER	session_user()
SESSIONTIMEZONE	session_time_zone()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax Function	Internal Function
SUBSTR	• SUBSTR(str, pos) corresponds to substr(str, pos). • SUBSTR(str FROM pos) corresponds to substr(str, pos). • SUBSTR(str, pos, len) corresponds to substr(str, pos, len). • SUBSTR(str FROM pos FOR len) corresponds to substr(str, pos, len). • SUBSTR(str FOR len FROM pos) corresponds to substr(str, pos, len). • SUBSTR(str FOR len) corresponds to substr(str, 1::int4, len::int4).
SUBSTRING	• SUBSTRING(str, pos) corresponds to substring(str, pos). • SUBSTRING(str FROM pos) corresponds to substring(str, pos). • SUBSTRING(str, pos, len) corresponds to substring(str, pos, len). • SUBSTRING(str FROM pos FOR len) corresponds to substring(str, pos, len). • SUBSTRING(str FOR len FROM pos) corresponds to substring(str, pos, len). • SUBSTRING(str FOR len) corresponds to substring(str, 1::int4, len::int4).
SYSDATE	• When **mapping_date_to_datea** is set to **off**, current_sysdate() is used. • When **mapping_date_to_datea** is set to **on**, sysdate_a() is used.
TRIM(BOTH [A | A [FROM]] B)	btrim(B[, A])
TRIM(LEADING [A | A [FROM]] B)	ltrim(B[, A])
TRIM(TRAILING [A | A [FROM]] B)	rtrim(B[, A])
USER	current_user()
XMLEXISTS(A PASSING [BY REF] B [BY REF])	xmlexists(A, B)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> For a syntax function that has an internal mapping function, the projection alias in the query result actually displays the name of the implementation function internally called, not the syntax keyword itself.

```
-- Basic usage.
gaussdb=# SELECT gs_search_function_with_name_and_arg('abs(int)');
     gs_search_function_with_name_and_arg
-----------------------------------------------
 pg_catalog.int4 pg_catalog.abs(pg_catalog.int4)
(1 row)

gaussdb=# SELECT gs_search_function_with_name_and_arg('pg_catalog.avg(int16)');
       gs_search_function_with_name_and_arg
-------------------------------------------------------
 pg_catalog.float8 pg_catalog.avg(pg_catalog.float8)
(1 row)

gaussdb=# SELECT gs_search_function_with_name_and_arg('lag(datea, int, unknown) over()');
                              gs_search_function_with_name_and_arg
-------------------------------------------------------------------------------------------------------------
 pg_catalog.anyelement pg_catalog.lag(pg_catalog.anyelement, pg_catalog.int4,
pg_catalog.anyelement)
(1 row)
```

- gs_search_operator_with_name_and_arg(ltype cstring, opname cstring, rtype cstring)

  Description: Returns the information about the overloaded operator that is actually called based on the operator name and the data types of the left and right operands.

  Return type: text

  > **NOTE**
  >
  > - This function is supported only in Oracle-compatible mode.
  > - Only binary operators can be queried.
  > - To query the overloading information of an operator whose input parameter contains **NULL**, replace **NULL** with **UNKNOWN**.
  > - To query the overloading information of the != or ^= operator, use <> instead.
  > - If the operator to be searched for is changed or the operator selection logic is changed, the result of the function may change. For details about the operator selection logic, see **Operators**.

```
-- Basic usage.
gaussdb=# SELECT gs_search_operator_with_name_and_arg('int', '+', 'int');
            gs_search_operator_with_name_and_arg
-------------------------------------------------------------
 pg_catalog.int4 (pg_catalog.int4 pg_catalog.+ pg_catalog.int4)
(1 row)

gaussdb=# SELECT gs_search_operator_with_name_and_arg('int', '+', 'float4');
               gs_search_operator_with_name_and_arg
-----------------------------------------------------------------------
 pg_catalog.float8 (pg_catalog.float8 pg_catalog.+ pg_catalog.float4)
(1 row)
```

1.6.25 Conditional Expression Functions

Conditional Expression Functions

- coalesce(expr1, expr2, ..., exprn)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Returns the first non-NULL value in the parameter list. **COALESCE(expr1, expr2)** is equivalent to **CASE WHEN expr1 IS NOT NULL THEN expr1 ELSE expr2 END**.

Example:

```
gaussdb=# SELECT coalesce(NULL,'hello');
 coalesce
----------
 hello
(1 row)
```

Note:

- **NULL** is returned only if all parameters are **NULL**.
- It is often used to replace **NULL** with the default value.
- Like a **CASE** expression, **COALESCE** only evaluates the parameters that are needed to determine the result. That is, parameters to the right of the first non-NULL parameter are not evaluated.

- decode(base_expr, compare1, value1, compare2,value2, ... default)

  Description: Compares **base_expr** with each **compare(n)** and returns **value(n)** if they are matched. If no match occurs, **default** is returned.

  Example:

  ```
  gaussdb=# SELECT decode('A','A',1,'B',2,0);
   case
  ------
   1
  (1 row)
  ```

  Note: Operations on the XML data are not supported.

- nullif(expr1, expr2)

  Description: Returns **NULL** only when **expr1** is equal to **expr2**. Otherwise, **expr1** is returned. This function is logically equivalent to CASE WHEN expr1 = expr2 THEN NULL ELSE expr1 END.

  NOTE

  The nullif(expr1, expr2) function is a mapping function. Therefore, the corresponding function definition cannot be found in the pg_proc system catalog.

  Example:

  ```
  gaussdb=# SELECT nullif('hello','world');
   nullif
  --------
   hello
  (1 row)
  ```

  Note: Operations on the XML data are not supported.

  Assume the two parameter data types are different:

  - If implicit conversion exists between the two data types, implicitly convert the parameter of lower priority to this data type using the data type of higher priority. If the conversion succeeds, computation is performed. Otherwise, an error is reported. Example:

    ```
    gaussdb=# SELECT nullif('1234'::VARCHAR,123::INT4);
     nullif
    --------
       1234
    (1 row)
    gaussdb=# SELECT nullif('1234'::VARCHAR,'2012-12-24'::DATE);
    ERROR:  invalid input syntax for type timestamp: "1234"
    ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - If implicit conversion is not applied between two data types, an error is displayed. Example:

```
gaussdb=# SELECT nullif(1::bit, '1'::MONEY);
ERROR:  operator does not exist: bit = money
LINE 1: SELECT nullif(1::bit, '1'::MONEY);
                  ^
HINT:  No operator matches the given name and argument type(s). You might need to add explicit type casts.
CONTEXT:  referenced column: nullif
```

- nvl(expr1 , expr2)

  Description:

  - If the value of **expr1** is **NULL**, the value of **expr2** is returned.
  - If the value of **expr1** is not **NULL**, the value of **expr1** is returned.

  Example:

```
gaussdb=# SELECT nvl('hello','world');
  nvl
-------
 hello
(1 row)
```

  Note: **expr1** and **expr2** can be of any data type. If they have different data types, the NVL function will check whether **expr2** can be implicitly converted into **expr1**. If **expr2** can be implicitly converted into **expr1**, the function will return the data type of **expr1**. If **expr2** cannot be implicitly converted into **expr1** but **expr1** can be implicitly converted into **expr2**, the function will return the data type of **expr2**. If no implicit type conversion exists between the two parameters and the parameters are different data types, an error is reported.

- nvl2(expr1, expr2, expr3)

  Description:

  - If *expr1* is **NULL**, *expr3* is returned.
  - If *expr1* is not **NULL**, *expr2* is returned.

    NOTE

    This function takes effect in ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s1**.

  Example:

```
gaussdb=# SELECT nvl2('hello','world','other');
  case
-------
 world
(1 row)
```

  Note: The *expr2* and *expr3* parameters can be of any type. If the last two parameters of NVL2 are of different types, check whether *expr3* can be implicitly converted to *expr2*. If *expr3* cannot be implicitly converted to *expr2*, an error is returned. If the first parameter is of the numeric type, the function converts this parameter and other parameters to the numeric type, and then compares them. If the parameters cannot be converted, an error message is displayed. If the first parameter is of another type, the function converts other parameters to the type of the first parameter for comparison. If the parameters cannot be converted, an error message is displayed.

- greatest(expr1 [, ...])

  Description: Selects the largest value from a list of any number of expressions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT greatest(1*2,2-3,4-1);
 greatest
----------
        3
(1 row)
gaussdb=# SELECT greatest('HARRY', 'HARRIOT', 'HAROLD');
 greatest
----------
 HARRY
(1 row)
```

Note: Operations on the XML data are not supported.

NOTE

In ORA-compatible mode, when **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1**:

1. If the value of any parameter is null, the function returns null.
2. If the first parameter is of the numeric type, the function converts this parameter and other parameters to the numeric type, and then compares them. If the parameters cannot be converted, an error message is displayed. If the first parameter is of another type, the function converts other parameters to the type of the first parameter for comparison. If the parameters cannot be converted, an error message is displayed.

- least(expr1 [, ...])

  Description: Selects the smallest value from a list of any number of expressions.

  Example:

```
gaussdb=# SELECT least(1*2,2-3,4-1);
 least
-------
    -1
(1 row)
gaussdb=# SELECT least('HARRY','HARRIOT','HAROLD');
 least
--------
 HAROLD
(1 row)
```

  Note: Operations on the XML data are not supported.

  NOTE

  In ORA-compatible mode, when **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1**:

  1. If the value of any parameter is null, the function returns null.
  2. If the first parameter is of the numeric type, the function converts this parameter and other parameters to the numeric type, and then compares them. If the parameters cannot be converted, an error message is displayed. If the first parameter is of another type, the function converts other parameters to the type of the first parameter for comparison. If the parameters cannot be converted, an error message is displayed.

- EMPTY_BLOB()

  Description: Initiates a BLOB variable in an INSERT or an UPDATE statement to a **NULL** value.

  Return type: BLOB

  Example:

```
-- Create a table.
gaussdb=# CREATE TABLE blob_tb(b blob,id int) DISTRIBUTE BY REPLICATION;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Insert data.
gaussdb=# INSERT INTO blob_tb VALUES (empty_blob(),1);
-- Drop the table.
gaussdb=# DROP TABLE blob_tb;
```

Note: The length is 0 obtained using DBE_LOB.GET_LENGTH.

- EMPTY_CLOB()

  Description: Initiates a CLOB variable in an INSERT or UPDATE statement to a null value.

  > NOTE
  >
  > This function is valid in ORA-compatible mode only when **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1**.

  Return type: CLOB

  Example:

```
-- Create a table.
gaussdb=# CREATE TABLE clob_tb(c clob,id int);
-- Insert data.
gaussdb=# INSERT INTO clob_tb VALUES (empty_clob(),1);
-- Drop the table.
gaussdb=# DROP TABLE clob_tb;
```

  Note: The length is 0 obtained using DBE_LOB.GET_LENGTH.

- lnnvl(condition)

  Description: Checks the condition in the WHERE clause of a query statement. If the condition is true, **false** is returned. If the condition is unknown or false, **true** is returned.

  condition: The value must be a logical expression. However, it cannot be used in composite conditions with keywords such as AND, OR, and BETWEEN.

  Return type: Boolean

  Example:

```
-- Create a table.
gaussdb=# CREATE TABLE student_demo (name VARCHAR2(20), grade  NUMBER(10,2));
CREATE TABLE

-- Insert data.
gaussdb=# INSERT INTO student_demo VALUES ('name0',0);
INSERT 0 1
gaussdb=# INSERT INTO student_demo VALUES ('name1',1);
INSERT 0 1
gaussdb=# INSERT INTO student_demo VALUES ('name2',2);
INSERT 0 1

-- Call lnnvl.
gaussdb=# SELECT * FROM student_demo WHERE LNNVL(name = 'name1');
 name  | grade
-------+-------
 name0 |  0.00
 name2 |  2.00
(2 rows)

-- Drop the table.
gaussdb=# drop table student_demo;
DROP TABLE
```

  > NOTE
  >
  > The lnnvl function is supported in ORA-compatible mode only when **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s2**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- isnull(expr)

  Description: Checks whether **expr** is **NULL**. If **expr** is **NULL**, **true** is returned. Otherwise, **false** is returned. The isnull function is compatible in full mode.

  Parameter: Any type of input is supported.

  Return type: Boolean

  Example:

```
gaussdb=# SELECT isnull(null);
 isnull
--------
 t
(1 row)

gaussdb=# SELECT isnull(1);
 isnull
--------
 f
(1 row)
```

- if(expr1, expr2, expr3)

  Description: Checks the value of **expr1**. If **expr1** is **true**, **expr2** is returned. Otherwise, **expr3** is returned. This function is logically equivalent to CASE WHEN expr1 THEN expr2 ELSE expr3 END.

  Parameters: See **Table 1-103**.

  Table 1-103 Parameter types

Parameter	Valid Input Parameter Type	Description
expr1	Boolean	Determines the return value of the if() function.
expr2	Any type	If **expr1** is **true**, **expr2** is returned.
expr3	Any type	If **expr1** is **false**, **expr3** is returned.

  Return value type: related to the input parameter type. For details about the output rules, see the description.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- if(expr1, expr2, expr3) is supported only in MySQL-compatible mode.
- When the compatibility parameters **b_format_version** is set to **''** and **b_format_dev_version** is set to **''**, the output result type of the if(expr1, expr2, expr3) function is the same as that of the CASE WHEN expr1 THEN expr2 ELSE expr3 END function. When the compatibility parameters **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s1'**, the return value type is deduced according to the following rules in descending order:
  - If the two input parameters are of the same type, the return value type is the same as the input parameter type.
  - If one of the input parameters is of the SET type, the return value is of the TEXT type.
  - If one of the input parameters is of the BLOB type, the return value is of the BLOB type.
  - If one of the input parameters is of the TEXT type, the return value is of the TEXT type.
  - If one of the input parameters is of the STRING type, the return value is of the TEXT type.
  - If the two input parameters are of the time type, the return value is of the TIMESTAMPTZ type; if one of the input parameters is of the time type but the other is not, the return value is of the TEXT type.
  - If both input parameters are signed integers and one input parameter is of the BIGINT type, the return value is of the BIGINT type; otherwise, the return value is of the INT type.
  - If both input parameters are unsigned integers and one input parameter is of the UNSIGNED BIGINT type, the return value is of the UNSIGNED BIGINT type; otherwise, the return value is of the UNSIGNED INT type.
  - If the two input parameters are of the floating-point type and integer type respectively, the return value is of the floating-point type.
  - Except the preceding NUMERIC type combinations, the return value for other NUMERIC type combinations is of the NUMERIC type.
  - If no implicit conversion function exists between the input parameter type and return value type, an error is reported.
- If the input parameter **expr1** is **NULL**, the returned result is the same as that when **expr1** is **FALSE**.
- The if(expr1, expr2, expr3) function is a mapping function. Therefore, the corresponding function definition cannot be found in the pg_proc system catalog.

Example:

```
-- expr1 is an expression.
gaussdb=# select if(2>3, 'true', 'false');
 case
-------
 false
(1 row)

-- If the input parameter expr1 is NULL, the result is the same as that when expr1 is false, and the value of expr3 is returned.
gaussdb=# select if(null, 'not null', 'is null');
  case
---------
 is null
(1 row)
```

- ifnull(expr1, expr2)

  Description: Returns the value of **expr2** if **expr1** is **NULL**; otherwise, returns the value of **expr1**. This function is logically equivalent to nvl(expr1, expr2).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter: any type.

Return value type: related to the input parameter type. For details about the output rules, see the description.

NOTE

- ifnull(expr1, expr2) is supported only in MySQL-compatible mode.
- When the compatibility parameters **b_format_version** is set to **''** and **b_format_dev_version** is set to **''**, the output result type of the ifnull(expr1, expr2) function is the same as that of the nvl(expr1, expr2) function. When the compatibility parameters **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s1'**, the return value type is deduced according to the following rules in descending order:
  - If the two input parameters are of the same type, the return value type is the same as the input parameter type.
  - If one of the input parameters is of the SET type, the return value is of the TEXT type.
  - If one of the input parameters is of the BLOB type, the return value is of the BLOB type.
  - If one of the input parameters is of the TEXT type, the return value is of the TEXT type.
  - If one of the input parameters is of the STRING type, the return value is of the TEXT type.
  - If the two input parameters are of the time type, the return value is of the TIMESTAMPTZ type; if one of the input parameters is of the time type but the other is not, the return value is of the TEXT type.
  - If one input parameter is of the FLOAT4 type and the other is of any numeric type, the return type is DOUBLE.
  - If both input parameters are of the NUMERIC type, the return value is of the type with higher precision. For example, if the input parameter types are TINYINT and INT, the return value type is INT.
  - If no implicit conversion function exists between the input parameter type and return value type, an error is reported.
- If both input parameters are **NULL**, the return value is **NULL**; otherwise, the first value that is not **NULL** is returned.
- The ifnull(expr1, expr2) function is a mapping function. Therefore, the corresponding function definition cannot be found in the pg_proc system catalog.

Example:

```
-- If the input parameter is an empty string, the output is an empty string instead of NULL.
gaussdb=# SELECT ifnull('', null) is null as a;
 a
---
 f
(1 row)

-- If both input parameters are NULL, the output is NULL.
gaussdb=# SELECT ifnull(null, null) is null as a;
 a
---
 t
(1 row)

-- If the input parameters are NULL and a character string, the output is the first non-null value.
gaussdb=# SELECT ifnull(null, 'A') as a;
 a
---
 A
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.6.26 System Information Functions

Session Information Functions

- SYS_CONTEXT()

  Description: Returns the value of the parameter associated with the context namespace at the current time. This function has the same function as **USERENV()** when the first input parameter is **userenv**.

  Currently, the following parameters can be used as the second input parameter.

  'current_edition_name', 'current_schema', 'current_sql', 'current_user', 'current_userid', 'database_role', 'ip_address', 'is_apply_server', 'isdba', 'nls_currency', 'nls_date_format', 'nls_date_language', 'server_host', 'service_name', 'session_user', 'session_userid', 'db_name', 'db_unique_name', 'instance', 'instance_name', 'current_edition_id', 'current_schemaid', 'lang', 'language', and 'sid'

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT SYS_CONTEXT('userenv','NLS_CURRENCY');
   sys_context
  -------------
   $
  (1 row)

  gaussdb=# SELECT SYS_CONTEXT('userenv','NLS_DATE_FORMAT');
    sys_context
  ---------------
   ISO, MDY
  (1 row)

  gaussdb=# SELECT SYS_CONTEXT('userenv','NLS_DATE_LANGUAGE');
   sys_context
  -------------
   en_US.UTF-8
  (1 row)
  ```

- USERENV()

  Description: Returns the value of the parameter associated with the context namespace at the current time. This function has the same function as **SYS_CONTEXT()** when the first input parameter is **userenv**.

  Currently, the following input parameters are supported:

  'current_edition_name', 'current_schema', 'current_sql', 'current_user', 'current_userid', 'database_role', 'ip_address', 'is_apply_server', 'isdba', 'nls_currency', 'nls_date_format', 'nls_date_language', 'server_host', 'service_name', 'session_user', 'session_userid', 'db_name', 'db_unique_name', 'instance', 'instance_name', 'current_edition_id', 'current_schemaid', 'lang', 'language', and 'sid'

  Currently, the following input parameters are not supported:

  'action', 'is_application_root', 'is_application_pdb', 'audited_cursorid', 'authenticated_identity', 'authentication_data', 'authentication_method', 'cdb_domain', 'cdb_name', 'client_identifier', 'con_id', 'con_name', 'current_sql_length', 'db_domain', 'db_supplemental_log_level', 'dblink_info', 'drain_status', 'entryid', 'enterprise_identity', 'fg_job_id', 'global_uid', 'identification_type', 'is_dg_rolling_upgrade', 'ldap_server_type', 'module',

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

'network_protocol', 'nls_calendar', 'nls_sort', 'nls_territory', 'oracle_home', 'os_user', 'platform_slash', 'policy_invoker', 'proxy_enterprise_identity', 'proxy_user', 'proxy_userid', 'scheduler_job', 'session_edition_id', 'session_edition_name', 'sessionid', 'statementid', 'terminal', 'unified_audit_sessionid', 'session_default_collation', 'client_info', 'bg_job_id', 'client_program_name', 'current_bind', 'global_context_memory', 'host', and 'current_sqln'

Return type: text.

Example:

```
gaussdb=# SELECT USERENV('sid');
 userenv
---------
 955
(1 row)

gaussdb=# SELECT USERENV('instance');
 userenv
---------
 14524
(1 row)

gaussdb=# SELECT USERENV('action');
 userenv
---------

(1 row)

gaussdb=# SELECT USERENV('actijj');
ERROR:  The configuration parameter actijj cannot be identified.
CONTEXT:  PL/pgSQL function userenv(text) line 114 at RETURN
referenced column: userenv
```

NOTE

If the input parameter is not supported, NULL is returned. If the input parameter is invalid, an error is reported.

- current_catalog()

  Description: Name of the current database (called "catalog" in the SQL standard)

  Return type: name.

  Example:

  ```
  testdb=# SELECT current_catalog;
   current_database
  ------------------
   testdb
  (1 row)
  ```

- current_database()

  Description: Name of the current database

  Return type: name.

  Example:

  ```
  testdb=# SELECT current_database();
   current_database
  ------------------
   testdb
  (1 row)
  ```

- current_query()

  Description: Text of the currently executing query committed by the client (which might contain more than one statement)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: text.

Example:

```
gaussdb=# SELECT current_query();
      current_query
--------------------------
 SELECT current_query();
(1 row)
```

- current_schema[()]

  Description: Name of the current schema

  Return type: name.

  Example:

  ```
  gaussdb=# SELECT current_schema();
   current_schema
  ----------------
   public
  (1 row)
  ```

  Note: **current_schema** returns the first valid schema name in the search path. (If the search path is empty or contains no valid schema name, **NULL** is returned.) This is the schema that will be used for any tables or other named objects that are created without specifying a target schema.

- current_schemas(Boolean)

  Description: Name of a schema in the search path

  Return type: name[].

  Example:

  ```
  gaussdb=# SELECT current_schemas(true);
    current_schemas
  ---------------------
   {pg_catalog,public}
  (1 row)
  ```

  Note:

  current_schemas(Boolean) returns an array of the names of all schemas in the search path. The Boolean option specifies whether implicitly included system schemas such as **pg_catalog** are included in the returned search path.

  NOTE

  The search path can be altered at runtime. The command is as follows:

  ```
  SET search_path TO schema [, schema, ...]
  ```

- database()

  Description: Returns the name of the current schema.

  Return type: name.

  Example:

  ```
  gaussdb=# SELECT database();
   database
  ----------
   public
  (1 row)
  ```

  NOTE

  This function is valid in a MySQL-compatible database when the GUC parameters **b_format_version** and **b_format_dev_version** are set to **'5.7'** and **'s1'**, respectively.

- current_user()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Username in the current operating environment.

Return type: name.

Example:

```
gaussdb=# SELECT current_user;
 current_user
--------------
 omm
(1 row)
```

Note: **current_user** is the user identifier used for permission check. Generally, it indicates a session user, which can be modified by **SET ROLE**. During function execution, it is affected by the SECURITY DEFINER attribute.

- definer_current_user()

  Description: Username in the current operating environment.

  Return type: name.

  Example:

  ```
  gaussdb=# SELECT definer_current_user();
   definer_current_user
  ----------------------
   omm
  (1 row)
  ```

  Note: Generally, the results of **definer_current_user** and **current_user** are the same. However, when this function is executed in a stored procedure, the name of user who defines the current stored procedure is returned.

- pg_current_sessionid()

  Description: Session ID in the current operating environment.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT pg_current_sessionid();
      pg_current_sessionid
  ----------------------------
   1579228402.140190434944768
  (1 row)
  ```

  Note: pg_current_sessionid() is used to obtain the session ID in the current operating environment. The format of the value is *Timestamp.Session ID*. When **enable_thread_pool** is disabled, the actual session ID is the thread ID.

- pg_current_sessid()

  Description: Session ID in the current operating environment.

  Return type: text.

  Example:

  ```
  gaussdb=# select pg_current_sessid();
  pg_current_sessid
  -------------------
  140308875015936
  (1 row)
  ```

  Note: In thread pool mode, the session ID of the current session is obtained. In non-thread pool mode, the background thread ID of the current session is obtained.

- pg_current_userid()

  Description: Current user ID.

  Return type: text.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT pg_current_userid();
pg_current_userid
-------------------
10
(1 row)
```

- tablespace_oid_name(oid)

  Description: Queries the tablespace name based on the tablespace OID.

  Return type: text.

  Example:

  ```
  gaussdb=# select tablespace_oid_name(1663);
   tablespace_oid_name
  ---------------------
   pg_default
  (1 row)
  ```

- inet_client_addr()

  Description: Remote connection address. **inet_client_addr** returns the IP address of the current client.

  > NOTE
  >
  > This function takes effect only in remote connection mode.

  Return type: inet.

  Example:

  ```
  gaussdb=# SELECT inet_client_addr();
   inet_client_addr
  ------------------
   10.10.0.50
  (1 row)
  ```

- inet_client_port()

  Description: Remote connection port. **inet_client_port** returns the port number of the current client.

  > NOTE
  >
  > This function takes effect only in remote connection mode.

  Return type: int.

  Example:

  ```
  gaussdb=# SELECT inet_client_port();
   inet_client_port
  ------------------
            33143
  (1 row)
  ```

- inet_server_addr()

  Description: Local connection address. **inet_server_addr** returns the IP address on which the server accepts the current connection.

  > NOTE
  >
  > This function takes effect only in remote connection mode.

  Return type: inet.

  Example:

  ```
  gaussdb=# SELECT inet_server_addr();
   inet_server_addr
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
------------------
 10.10.0.13
(1 row)
```

- inet_server_port()

  Description: Local connection port. **inet_server_port** returns the number of the port receiving the current connection. All these functions return **NULL** if the current connection is via a UDS.

  > NOTE
  >
  > This function takes effect only in remote connection mode.

  Return type: int.

  Example:

```
gaussdb=# SELECT inet_server_port();
 inet_server_port
------------------
             8000
(1 row)
```

- pg_backend_pid()

  Description: ID of the worker thread connected to the current session.

  Return type: int.

  Example:

```
gaussdb=# SELECT pg_backend_pid();
 pg_backend_pid
-----------------
 140229352617744
(1 row)
```

- pg_conf_load_time()

  Description: Configures load time. **pg_conf_load_time** returns the timestamp when the server configuration files were last loaded.

  Return type: timestamp with time zone.

  Example:

```
gaussdb=# SELECT pg_conf_load_time();
      pg_conf_load_time
-------------------------------
 2017-09-01 16:05:23.89868+08
(1 row)
```

- pg_my_temp_schema()

  Description: OID of the temporary schema of a session. The value is **0** if the OID does not exist.

  Return type: oid.

  Example:

```
gaussdb=# SELECT pg_my_temp_schema();
 pg_my_temp_schema
-------------------
                 0
(1 row)
```

  Note: **pg_my_temp_schema** returns the OID of the current session's temporary schema, or **0** if it has no temporary schemas (because no temporary tables are created). **pg_is_other_temp_schema** returns **true** if the given OID is the OID of another session's temporary schema.

- pg_is_other_temp_schema(oid)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Specifies whether the schema is the temporary schema of another session.

Return type: Boolean

Example:

```
gaussdb=# SELECT pg_is_other_temp_schema(25356);
 pg_is_other_temp_schema
-------------------------
 f
(1 row)
```

- pg_listening_channels()

  Description: Name of the channel that the session is currently listening on.

  Return type: setof text.

  Example:

  ```
  gaussdb=# SELECT pg_listening_channels();
   pg_listening_channels
  -----------------------
  (0 rows)
  ```

  Note: **pg_listening_channels** returns a set of names of channels that the current session is currently listening on.

- pg_postmaster_start_time()

  Description: Server start time. **pg_postmaster_start_time** returns the timestamp with time zone when the server is started.

  Return type: timestamp with time zone.

  Example:

  ```
  gaussdb=# SELECT pg_postmaster_start_time();
    pg_postmaster_start_time
  ------------------------------
   2017-08-30 16:02:54.99854+08
  (1 row)
  ```

- sessionid2pid()

  Description: Obtains PID information from a session ID (for example, the **sessid** column in **pv_session_stat**).

  Return type: int8

  Example:

  ```
  gaussdb=# select sessionid2pid(sessid::cstring) from pv_session_stat limit 2;
    sessionid2pid
  -----------------
   139973107902208
   139973107902208
  (2 rows)
  ```

- session_context('namespace' , 'parameter')

  Description: Obtains and returns the parameter values of a specified namespace.

  Return type: VARCHAR.

  Example:

  ```
  gaussdb=# SELECT session_context('USERENV', 'CURRENT_SCHEMA');
   session_context
  -----------------
   public
  (1 row)
  ```

  The result varies according to the current actual schema.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Note: Currently, only the SESSION_CONTEXT('USERENV', 'CURRENT_SCHEMA') and SESSION_CONTEXT('USERENV', 'CURRENT_USER') formats are supported.

- pg_trigger_depth()

  Description: Nesting level of triggers.

  Return type: int.

  Example:

```
gaussdb=# SELECT pg_trigger_depth();
 pg_trigger_depth
------------------
                0
(1 row)
```

- opengauss_version()

  Description: Referenced openGauss kernel version.

  Return type: text.

  The following is an example. Replace *x.x.x* in the query result with the actual value.

```
gaussdb=# SELECT opengauss_version();
 opengauss_version
--------------------
 x.x.x
(1 row)
```

- gs_deployment()

  Description: Deployment mode of the current system. For a distributed system, **Distribute** is returned.

  Return type: text.

  Example:

```
gaussdb=# SELECT gs_deployment();
 gs_deployment
---------------
 Distribute
(1 row)
```

- session_user

  Description: Session username.

  Return type: name.

  Example:

```
gaussdb=# SELECT session_user;
 session_user
--------------
 omm
(1 row)
```

  Note: **session_user** usually specifies the initial user connected to the current database, but system administrators can change this setting by using **SET SESSION AUTHORIZATION**.

- user

  Description: Equivalent to current_user.

  Return type: name.

  Example:

```
gaussdb=# SELECT user;
 current_user
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------------
 omm
(1 row)

-- Create an ORA-compatible database and set the GUC parameter a_format_func_col_name to
'user'.
gaussdb=# SET a_format_func_col_name = 'user';
SET

gaussdb=# SELECT user;
 user
------
 omm
(1 row)
```

> **NOTE**
>
> In ORA-compatible mode, if the GUC parameter **a_format_func_col_name** is set to **'user'**, the projection column name returned by the user function is **user**.

- get_shard_oids_byname()

  Description: Returns the OID of the node when the node name is entered.

  Return type: oid.

  Example:

```
gaussdb=# SELECT get_shard_oids_byname('datanode1');
 get_shard_oids_byname
-----------------------
 {16385}
(1 row)
```

- getpgusername()

  Description: Obtains the database username.

  Return type: name.

  Example:

```
gaussdb=# SELECT getpgusername();
 getpgusername
---------------
 GaussDB_userna
(1 row)
```

- getdatabaseencoding()

  Description: Obtains the database encoding mode.

  Return type: name.

  Example:

```
gaussdb=# SELECT getdatabaseencoding();
 getdatabaseencoding
---------------------
 SQL_ASCII
(1 row)
```

- version()

  Description: Version information. **version** returns a string describing a server's version.

  Return type: text.

  Example:

```
gaussdb=# SELECT version();
                                                
```

```
release
(1 row)
```

- working_version_num()

  Description: Version number. It returns a version number related to system compatibility.

  Return type: int.

  Example:

```
gaussdb=# SELECT working_version_num();
 working_version_num
---------------------
               92231
(1 row)
```

- get_hostname()

  Description: Returns the host name of the current node.

  Return type: text.

  Example:

```
gaussdb=# SELECT get_hostname();
 get_hostname
--------------
 linux-user
(1 row)
```

- get_nodename()

  Description: Returns the name of the current node.

  Return type: text.

  Example:

```
gaussdb=# SELECT get_nodename();
 get_nodename
--------------
 coordinator1
(1 row)
```

- get_nodeinfo(text)

  Description: Returns the value of the corresponding node information based on the search attribute. Currently, the search attributes include **node_name** and **node_type**.

  Return type: text.

  Example:

```
gaussdb=# SELECT get_nodeinfo('node_type');
 get_nodeinfo
--------------
 DIS_CN
(1 row)
gaussdb=# SELECT get_nodeinfo('node_name');
 get_nodeinfo
--------------
 coordinator1
(1 row)
```

- get_schema_oid(cstring)

  Description: Returns the OID of the queried schema.

  Return type: oid.

  Example:

```
gaussdb=# SELECT get_schema_oid('public');
 get_schema_oid
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
----------------
          2200
(1 row)
```

- pgxc_parse_clog(OUT xid int8, OUT nodename text, OUT status text)

  Description: Returns the status of all transactions in the current cluster.

  Return type: SETOF record

  Example:

  ```
  gaussdb=# SELECT pgxc_parse_clog();
   pgxc_parse_clog
  ----------------
  (0,dn_6004_6005_6006,INPROGRESS)
  (1,dn_6004_6005_6006,COMMITTED)
  (2,dn_6004_6005_6006,INPROGRESS)
  (3 row)
  ```

- pgxc_parse_clog(IN int2 bucketid, OUT xid int8, OUT nodename text, OUT status text)

  Description: Returns the statuses of all transactions with the specified bucket IDs in the current cluster. The input parameter is the bucket IDs of the hash bucket table. The value range is [–1,1023]. For an ordinary table, the value is **–1**.

  Return type: SETOF record

  Example:

  ```
  gaussdb=# SELECT pgxc_parse_clog('-1');
   pgxc_parse_clog
  ----------------
  (0,dn_6004_6005_6006,INPROGRESS)
  (1,dn_6004_6005_6006,COMMITTED)
  (2,dn_6004_6005_6006,INPROGRESS)
  (3 row)
  ```

- pgxc_prepared_xact()

  Description: Returns the list of transaction GIDs at the prepared stage in the cluster.

  Return type: set of text

  Example:

  ```
  gaussdb=# SELECT pgxc_prepared_xact();
   pgxc_prepared_xact
  --------------------
  (0 row)
  ```

- pgxc_xacts_iscommitted()

  Description: Returns the status of the transaction with the specified XID in the cluster. **t** indicates the committed state, **f** indicates the aborted state, and **null** indicates other states. Users must have the SYSADMIN or MONADMIN permission.

  Return type: SETOF record

  Example:

  ```
  gaussdb=# SELECT pgxc_xacts_iscommitted(1);
   pgxc_xacts_iscommitted
  --------------------
  (dn_6004_6005_6006,t)
  (cn_5001,t)
  (cn_5002,t)
  (dn_6001_6002_6003,t)
  (4 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pgxc_total_memory_detail()

  Description: Displays the memory usage in the cluster. Users must have the SYSADMIN or MONADMIN permission.

  > **NOTE**
  >
  > If the GUC parameter **enable_memory_limit** is set to **off**, this function cannot be used.

  Return type: set of pv_total_memory_detail

  Example:

```
gaussdb=# SELECT pgxc_total_memory_detail();
pgxc_total_memory_detail
--------------------
(dn_6004_6005_6006,max_process_memory,81920)
(dn_6004_6005_6006,process_used_memory,72747)
(dn_6004_6005_6006,max_dynamic_memory,12096)
(dn_6004_6005_6006,dynamic_used_memory,1530)
(4 row)
```

- pv_total_memory_detail()

  Description: Collects statistics on memory usage of the current database node in the unit of MB.

  > **NOTE**
  >
  > If the GUC parameter **enable_memory_limit** is set to **off**, this function cannot be used.

  Return type: record.

  Table 1-104 Return values

Name	Type	Description
nodename	text	Node name.

Name	Type	Description
memorytype	text	Memory type. The value must be one of the following: • **max_process_memory**: memory occupied by a GaussDB cluster instance • **process_used_memory**: memory occupied by a GaussDB thread • **max_dynamic_memory**: maximum dynamic memory • **dynamic_used_memory**: used dynamic memory • **dynamic_peak_memory**: dynamic peak memory • **dynamic_used_shrctx**: maximum dynamic shared memory context • **dynamic_peak_shrctx**: dynamic peak value of the shared memory context • **max_shared_memory**: maximum shared memory • **shared_used_memory**: used shared memory • **high_backend_thread_memory**: memory used by high-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **high_backend_thread_max_memory**: memory reserved for high-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **medium_backend_thread_memory**: memory used by medium-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **medium_backend_thread_max_memory**: memory reserved for medium-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **low_backend_thread_memory**: memory used by low-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
		• **low_backend_thread_max_memory**: memory reserved for low-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **max_sctpcomm_memory**: maximum memory allowed for the communications library. • **sctpcomm_used_memory**: memory used by the communications library. • **sctpcomm_peak_memory**: memory peak of the communications library. • **other_used_memory**: other used memory
memorymbytes	integer	Size of allocated memory-typed memory

- get_client_info()

  Description: Returns client information.

  Return type: record.

- pv_session_memory_detail()

  Description: Collects session memory usage by the MemoryContext node.

  Return type: record.

Table 1-105 Return values

Name	Type	Description
sessid	text	• When the thread pool is disabled (**enable_thread_pool = off**), this column indicates the thread start time+session ID (string: **timestamp.sessionid**). • When the thread pool is enabled (**enable_thread_pool = on**): If the memory context is at the thread level, this column indicates the thread start time +thread ID (string: **timestamp.threadid**). If the memory context is at the session level, the column indicates the thread start time +session ID (string: **timestamp.sessionid**).
threadid	bigint	Thread ID bound to a session.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
contextname	text	Name of the memory context.
level	smallint	Hierarchy of the memory context.
parent	text	Name of the parent memory context.
totalsize	bigint	Total size of the memory context, in bytes.
freesize	bigint	Total size of released memory in the memory context, in bytes.
usedsize	bigint	Size of used memory in the memory context, in bytes. For **TempSmallContextGroup**, this parameter specifies the number of collected memory contexts.

Access Permission Query Functions

Permissions on DDL statements, including ALTER, DROP, COMMENT, INDEX, and VACUUM, are inherent permissions implicitly owned by the owner.

NOTE

For the preceding implicitly owned permissions, the owner and users who inherit the owner's permissions cannot directly obtain them through the following access permission query functions. However, the actual owner and users who inherit the owner's permissions have these permissions, and the execution is not affected. To obtain permissions through the following query functions, you need to explicitly grant the permissions using GRANT before querying.

The following access permission query function only specifies whether a user has a certain permission on an object. That is, the permission on the object recorded in the **acl** column of the system catalog is returned.

- has_any_column_privilege(user, table, privilege)

  Description: Queries whether a specified user has permission to access any column in a table.

  Table 1-106 Parameter type description

Parameter	Valid Input Parameter Type	Description	Value Range
user	name, oid	A specific user	Username or ID

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Valid Input Parameter Type	Description	Value Range
table	text, oid	Tables	Table name or ID
privilege	text	Permission	• **SELECT**: allows the SELECT statement to be executed on any column of a specified table. • **INSERT**: allows the INSERT statement to be executed on any column of a specified table. • **UPDATE**: allows the UPDATE statement to be executed on any column of a specified table. • **REFERENCES**: allows a foreign key constraint (not supported in distributed mode). • **COMMENT**: allows the COMMENT statement to be executed on any column of a specified table.

Return type: Boolean

Example:

```
gaussdb=# CREATE USER test_user PASSWORD '********';
gaussdb=# CREATE TABLE test_table(col int);
gaussdb=# SELECT has_any_column_privilege('test_user','test_table','INSERT');
has_any_column_privilege
--------------------------
f
(1 row)

gaussdb=# DROP USER test_user;
```

- has_any_column_privilege(table, privilege)

  Description: Queries whether the current user has the permission to access any column of a table. For details about the valid parameter types, see **Table 1-106**.

  Return type: Boolean

  Note: **has_any_column_privilege** checks whether a user can access any column of a table in a particular way. Its parameter possibilities are analogous to those of **has_table_privilege**, except that the desired access permission type must be some combination of SELECT, INSERT, UPDATE, or REFERENCES.

  Example:

```
gaussdb=# SELECT has_any_column_privilege('test_table','INSERT');
has_any_column_privilege
--------------------------
t
(1 row)

gaussdb=# DROP TABLE test_table;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> Note that having any of these permissions at the table level indicates that the permission is implicitly granted for each column of the table. Therefore, **has_any_column_privilege** always returns **true** if **has_table_privilege** has the same parameters. A success message is also returned if a column-level permission is granted for at least one column.

- has_column_privilege(user, table, column, privilege)

  Description: Queries whether a specified user has permission to access columns.

  Table 1-107 Parameter type description

Parameter	Valid Input Parameter Type	Description	Value Range
user	name, oid	A specific user	Username or ID.
table	text, oid	Table name	Table name or ID.
column	text, smallint	Column name	Column name or attribute number.
privilege	text	Permission	• **SELECT**: allows the SELECT statement to be executed on specified columns of a table. • **INSERT**: allows the INSERT statement to be executed on specified columns of a table. • **UPDATE**: allows the UPDATE statement to be executed on specified columns of a table. • **REFERENCES**: allows a foreign key constraint (not supported in distributed mode). • **COMMENT**: allows the COMMENT statement to be executed on specified columns of a table.

  Return type: Boolean

- has_column_privilege(table, column, privilege)

  Description: Queries whether the current user has permission to access columns. For details about the valid parameter types, see **Table 1-107**.

  Return type: Boolean

  has_column_privilege checks whether a user can access a column in a particular way. Its parameter possibilities are analogous to those of

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

has_table_privilege, with the addition that the column can be specified either by name or attribute number. The desired access permission type must be some combination of SELECT, INSERT, UPDATE, or REFERENCES.

NOTE

Note that having any of these permissions at the table level indicates that the permission is implicitly granted for each column of the table.

- has_cek_privilege(user, cek, privilege)

  Description: Specifies whether a specified user has permission on CEKs.

  Table 1-108 Parameter type description

Parameter	Valid Input Parameter Type	Description	Value Range
user	name, oid	A specific user	Username or ID
cek	text, oid	CEK	Name or ID of a CEK
privilege	text	Permission	• **USAGE**: allows users to use the specified CEK. • **DROP**: allows users to delete the specified CEK.

  Return type: Boolean.

- has_cmk_privilege(user, cmk, privilege)

  Description: Specifies whether a specified user has permission on CMKs.

  Table 1-109 Parameter type description

Parameter	Valid Input Parameter Type	Description	Value Range
user	name, oid	A specific user	Username or ID
cmk	text, oid	CMK	Name or ID of the CMK
privilege	text	Permission	• **USAGE**: allows users to use the specified CMK. • **DROP**: allows users to delete the specified CMK.

  Return type: Boolean.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- has_database_privilege(user, database, privilege)

  Description: Queries whether a specified user has permission to access databases.

  Table 1-110 Parameter type description

Parameter	Valid Input Parameter Type	Description	Value Range
user	name, oid	A specific user	Username or ID
database	text, oid	Database	Database name or ID
privilege	text	Permission	• **CREATE**: For databases, allows new schemas to be created within the database. • **TEMPORARY**: allows users to create temporary tables when the database is used. • **TEMP**: allows users to create temporary tables when the database is used. • **CONNECT**: allows users to access specified databases. • **ALTER**: allows users to modify the attributes of a specified object. • **DROP**: allows users to delete a specified object. • **COMMENT**: allows users to define or modify comments of a specified object.

  Return type: Boolean

- has_database_privilege(database, privilege)

  Description: Queries whether the current user has the permission to access a database. For details about valid parameter types, see **Table 1-110**.

  Return type: Boolean

  Note: **has_database_privilege** checks whether a user can access a database in a particular way. Its parameter possibilities are analogous to those of **has_table_privilege**. The desired access permission type must be some combination of CREATE, CONNECT, TEMPORARY, or TEMP (which is equivalent to TEMPORARY).

- has_directory_privilege(user, directory, privilege)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-111 Parameter type description

Parameter	Valid Input Parameter Type	Description	Value Range
user	name, oid	A specific user	Username or ID
directory	text, oid	Directory	Directory name or OID
privilege	text	Permission	• **READ**: allows read operations on the directory. • **WRITE**: allows write operations on the directory.

Description: Specifies whether a specified user has permission on directories.

Return type: Boolean

- has_directory_privilege(directory, privilege)

  Description: Queries whether the current user has the permission to access a directory. For details about valid parameter types, see **Table 1-111**.

  Return type: Boolean

- has_foreign_data_wrapper_privilege(user, fdw, privilege)

Table 1-112 Parameter type description

Parameter	Valid Input Parameter Type	Description	Value Range
user	name, oid	A specific user	Username or ID
fdw	text, oid	Foreign data wrapper	Name or ID of the foreign data wrapper
privilege	text	Permission	**USAGE**: allows access to the foreign data wrapper.

Description: Queries whether a specified user has permission to access foreign data wrappers.

Return type: Boolean

- has_foreign_data_wrapper_privilege(fdw, privilege)

  Description: Queries whether the current user has permissions on foreign data wrappers. For details about valid parameter types, see **Table 1-112**.

  Return type: Boolean

  Note: **has_foreign_data_wrapper_privilege** checks whether a user can access a foreign data wrapper in a particular way. Its parameter possibilities are

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

analogous to those of **has_table_privilege**. The desired access permission type must be **USAGE**.

- has_function_privilege(user, function, privilege)

  Table 1-113 Parameter type description

Parameter	Valid Input Parameter Type	Description	Value Range
user	name, oid	A specific user	Username or ID
function	text, oid	Function	Function name or ID
privilege	text	Permission	**EXECUTE**: allows users to use specified functions and the operators that are realized by the functions. • **ALTER**: allows users to modify the attributes of a specified object. • **DROP**: allows users to delete a specified object. • **COMMENT**: allows users to define or modify comments of a specified object.

  Description: Queries whether a specified user has permission to access functions.

  Return type: Boolean

- has_function_privilege(function, privilege)

  Description: Queries whether the current user has permission to access functions. For details about valid parameter types, see **Table 1-113**.

  Return type: Boolean

  Note: **has_function_privilege** checks whether a user can access a function in a particular way. Its parameter possibilities are analogous to those of **has_table_privilege**. When a function is specified by a text string rather than by an OID, the allowed input is the same as that for the regprocedure data type (see **OID Types**). The desired access permission type must be EXECUTE.

- has_language_privilege(user, language, privilege)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-114 Parameter type description

Parameter	Valid Input Parameter Type	Description	Value Range
user	name, oid	A specific user	Username or ID
language	text, oid	Language	Language name or ID
privilege	text	Permission	**USAGE**: allows users to specify a procedural language when creating a function.

Description: Queries whether a specified user has permission to access languages.

Return type: Boolean

- has_language_privilege(language, privilege)

  Description: Queries whether the current user has permission to access languages. For details about valid parameter types, see **Table 1-114**.

  Return type: Boolean

  Note: **has_language_privilege** checks whether a user can access a procedural language in a particular way. Its parameter possibilities are analogous to those of **has_table_privilege**. The desired access permission type must be **USAGE**.

- has_nodegroup_privilege(user, nodegroup, privilege)

  Description: Checks whether a user has permission to access a cluster node.

  Return type: Boolean

Table 1-115 Parameter type description

Parameter	Valid Input Parameter Type	Description	Value Range
user	name, oid	A specific user	Existing username or ID.
nodegroup	text, oid	Cluster node	Existing cluster node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Valid Input Parameter Type	Description	Value Range
privilege	text	Permission	• **USAGE**: For sub-clusters, allows users who can access objects contained in the schema to access tables in the sub-cluster. • **CREATE**: For sub-clusters, allows users to create tables within the sub-cluster. • **COMPUTE**: allows elastic computing in the sub-cluster. • **ALTER**: allows users to modify the attributes of a specified object. • **DROP**: allows users to delete a specified object.

- has_nodegroup_privilege(nodegroup, privilege)

  Description: Checks whether a user has permission to access a cluster node.

  Return type: Boolean

- has_schema_privilege(user, schema, privilege)

  Description: Queries whether a specified user has permission to access schemas.

  Return type: Boolean

- has_schema_privilege(schema, privilege)

  Description: Queries whether the current user has permission to access schemas.

  Return type: Boolean

  Note: **has_schema_privilege** checks whether a user can access a schema in a particular way. Its parameter possibilities are analogous to those of **has_table_privilege**. The desired access permission type must be a combination among CREATE, USAGE, ALTER, DROP, and COMMENT. If the check type contains the CREATE permission and the checked schema is a schema with the same name as the user, the function returns **TRUE** only when the user has the OWNER permission on the schema due to the special constraint of the schema with the same name.

- has_sequence_privilege(user, sequence, privilege)

  Description: Queries whether a specified user has permissions on sequences.

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-116 Parameter type description

Parameter	Valid Input Parameter Type	Description	Value Range
user	name, oid	A specific user	Existing username or ID.
sequence	text, oid	Sequence	Existing sequence name or ID.
privilege	text	Permission	• **USAGE**: For sequences, allows users to use the nextval function. • **SELECT**: allows users to create a sequence. • **UPDATE**: allows users to execute the **UPDATE** statement. • **ALTER**: allows users to modify the attributes of a specified object. • **DROP**: allows users to delete a specified object. • **COMMENT**: allows users to define or modify comments of a specified object.

- has_sequence_privilege(sequence, privilege)

  Description: Queries whether the current user has permissions on sequences.

  Return type: Boolean
- has_server_privilege(user, server, privilege)

  Description: Queries whether a specified user has permission to access foreign servers.

  Return type: Boolean
- has_server_privilege(server, privilege)

  Description: Queries whether the current user has permission to access foreign servers.

  Return type: Boolean

  Note: **has_server_privilege** checks whether a user can access a foreign server in a particular way. Its parameter possibilities are analogous to those of **has_table_privilege**. The access permission type must be **USAGE**, **ALTER**, **DROP**, or **COMMENT**.
- has_table_privilege(user, table, privilege)

  Description: Queries whether a specified user has permission to access tables.

  Return type: Boolean
- has_table_privilege(table, privilege)

  Description: Queries whether the current user has permission to access tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: Boolean

Note: **has_table_privilege** checks whether a user can access a table in a particular way. The user can be specified by name or by OID (**pg_authid.oid**), or be set to **public** which indicates public pseudo roles. If this parameter is omitted, **current_user** is used. The table can be specified by name or by OID. When it is specified by name, the name can be schema-qualified if necessary. If the desired access permission type is specified by a text string, the text string must be one of the values **SELECT**, **INSERT**, **UPDATE**, **DELETE**, **TRUNCATE**, **REFERENCESTRIGGER**, **ALTER**, **DROP**, **COMMENT**, **INDEX**, or **VACUUM**. Optionally, **WITH GRANT OPTION** can be added to a permission type to test whether the permission is held with the grant option. Also, multiple permission types can be separated by commas (,), in which case the result will be **true** if any of the listed permissions is held.

Example:

```
gaussdb=# CREATE SCHEMA tpcds;
gaussdb=# CREATE TABLE tpcds.web_site(col int);
gaussdb=# CREATE USER test_user PASSWORD '********';
gaussdb=# SELECT has_table_privilege('tpcds.web_site', 'select');
 has_table_privilege
---------------------
 t
(1 row)

gaussdb=# SELECT has_table_privilege('test_user', 'tpcds.web_site', 'select,INSERT WITH GRANT
OPTION ');
 has_table_privilege
---------------------
 f
(1 row)

gaussdb=# DROP USER test_user;
gaussdb=# DROP TABLE tpcds.web_site;
gaussdb=# DROP SCHEMA tpcds;
```

- has_tablespace_privilege(user, tablespace, privilege)

  Description: Queries whether a specified user has permission to access tablespaces.

  Return type: Boolean

- has_tablespace_privilege(tablespace, privilege)

  Description: Queries whether the current user has permission to access tablespaces.

  Return type: Boolean

  Note: **has_tablespace_privilege** checks whether a user can access a tablespace in a particular way. Its parameter possibilities are analogous to those of **has_table_privilege**. The access permission type must be **CREATE**, **ALTER**, **DROP**, or **COMMENT**.

- pg_has_role(user, role, privilege)

  Description: Queries whether a specified user has permission on roles.

  Return type: Boolean

- pg_has_role(role, privilege)

  Description: Queries whether the current user has permission to access a role.

  Return type: Boolean

  Note: **pg_has_role** checks whether a user can access a role in a particular way. Its parameter possibilities are analogous to those of **has_table_privilege**,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

except that **public** cannot be used as a username. The desired access permission type must be some combination of **MEMBER** or **USAGE**. **MEMBER** denotes direct or indirect membership in the role (that is, permission **SET ROLE**), while **USAGE** denotes the usage permission on the role that is available without the **SET ROLE** permission.

- has_any_privilege(user, privilege)

  Description: Queries whether a specified user has certain ANY permission. If multiple permissions are queried at the same time, **true** is returned as long as one permission is obtained.

  Return type: Boolean

  Table 1-117 Parameter type description

Parameter	Valid Input Parameter Type	Description	Value Range
user	name	A specific user	Existing username

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Valid Input Parameter Type	Description	Value Range
privilege	text	ANY permission	The options are as follows: • CREATE ANY TABLE [WITH ADMIN OPTION] • ALTER ANY TABLE [WITH ADMIN OPTION] • DROP ANY TABLE [WITH ADMIN OPTION] • SELECT ANY TABLE [WITH ADMIN OPTION] • INSERT ANY TABLE [WITH ADMIN OPTION] • UPDATE ANY TABLE [WITH ADMIN OPTION] • DELETE ANY TABLE [WITH ADMIN OPTION] • TRUNCATE ANY TABLE [WITH ADMIN OPTION] • CREATE ANY SEQUENCE [WITH ADMIN OPTION] • CREATE ANY INDEX [WITH ADMIN OPTION] • CREATE ANY FUNCTION [WITH ADMIN OPTION] • EXECUTE ANY FUNCTION [WITH ADMIN OPTION] • CREATE ANY PACKAGE [WITH ADMIN OPTION] • EXECUTE ANY PACKAGE [WITH ADMIN OPTION] • CREATE ANY TYPE [WITH ADMIN OPTION]

Schema Visibility Query Functions

Each function performs the visibility check on one type of database objects. For functions and operators, an object in the search path is visible if there is no object of the same name and parameter data type earlier in the path. For operator classes, both name and associated index access methods are considered.

All these functions require OIDs to identify the objects to be checked. If you want to test an object by name, it is convenient to use the OID alias type (regclass, regtype, regprocedure, regoperator, regconfig, or regdictionary).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

For example, a table is said to be visible if the schema where the table is located is in the search path and no table of the same name appears earlier in the search path. This is equivalent to the statement that the table can be referenced by name without explicit schema qualification. To list the names of all visible tables, run the following command:

```
gaussdb=# SELECT relname FROM pg_class WHERE pg_table_is_visible(oid);
```

- pg_collation_is_visible(collation_oid)

  Description: Queries whether the collation is visible in search path.

  Return type: Boolean
- pg_conversion_is_visible(conversion_oid)

  Description: Queries whether the conversion is visible in the search path.

  Return type: Boolean
- pg_function_is_visible(function_oid)

  Description: Queries whether the function is visible in the search path.

  Return type: Boolean
- pg_opclass_is_visible(opclass_oid)

  Description: Queries whether the operator class is visible in the search path.

  Return type: Boolean
- pg_operator_is_visible(operator_oid)

  Description: Queries whether the operator is visible in the search path.

  Return type: Boolean
- pg_opfamily_is_visible(opclass_oid)

  Description: Queries whether the operator family is visible in the search path.

  Return type: Boolean
- pg_table_is_visible(table_oid)

  Description: Queries whether the table is visible in the search path.

  Return type: Boolean
- pg_ts_config_is_visible(config_oid)

  Description: Queries whether the text search configuration is visible in the search path.

  Return type: Boolean
- pg_ts_dict_is_visible(dict_oid)

  Description: Queries whether the text search dictionary is visible in the search path.

  Return type: Boolean
- pg_ts_parser_is_visible(parser_oid)

  Description: Queries whether the text search parser is visible in the search path.

  Return type: Boolean
- pg_ts_template_is_visible(template_oid)

  Description: Queries whether the text search template is visible in the search path.

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_type_is_visible(type_oid)

  Description: Queries whether the type (or domain) is visible in the search path.

  Return type: Boolean

System Catalog Information Functions

- format_type(type_oid, typemod)

  Description: Obtains the SQL name of a data type.

  Return type: text.

  Note: **format_type** returns the SQL name of a data type based on the OID of the data type and possible modifiers. If the specific modifier is unknown, pass **NULL** at the position of the modifier. Modifiers are generally meaningful only for data types with length restrictions. The SQL name returned by **format_type** contains the length of the data type, which can be calculated by taking sizeof(int32) from actual storage length [actual storage len – sizeof(int32)] in the unit of bytes. 32-bit space is required to store the customized length set by users. Therefore, the actual storage length contains 4 bytes more than the customized length. In the following example, the SQL name returned by **format_type** is character varying(6), indicating the length of the varchar type is 6 bytes. Therefore, the actual storage length of the varchar type is 10 bytes.

  ```
  gaussdb=# SELECT format_type((SELECT oid FROM pg_type WHERE typname='varchar'), 10);
      format_type
  ----------------------
   character varying(6)
  (1 row)
  ```

- pg_check_authid(role_oid)

  Description: Checks whether a role name with given OID exists.

  Return type: Boolean.

  ```
  gaussdb=# select pg_check_authid(1);
  pg_check_authid
  -----------------
  f
  (1 row)
  ```

- pg_describe_object(catalog_id, object_id, object_sub_id)

  Description: Obtains the description of a database object.

  Return type: text.

  Note: **pg_describe_object** returns the description of a database object specified by a catalog OID, an object OID, and a (possibly zero) sub-object ID. This is useful to determine the identity of an object stored in the **pg_depend** catalog.

- pg_get_constraintdef(constraint_oid)

  Description: Obtains definition of a constraint.

  Return type: text.

- pg_get_constraintdef(constraint_oid, pretty_bool)

  Description: Obtains definition of a constraint.

  Return type: text.

  Note: **pg_get_constraintdef** and **pg_get_indexdef** respectively reconstruct the creation command for a constraint and an index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_get_expr(pg_node_tree, relation_oid)

  Description: Decompiles internal form of an expression, assuming that any Vars in it refer to the relationship indicated by the second parameter.

  Return type: text.

- pg_get_expr(pg_node_tree, relation_oid, pretty_bool)

  Description: Decompiles internal form of an expression, assuming that any Vars in it refer to the relationship indicated by the second parameter.

  Return type: text.

  Note: **pg_get_expr** decompiles the internal form of an individual expression, such as the default value of a column. This helps to check the content of system catalogs. If the expression contains keywords, specify the OID of the relationship that the keywords refer to as the second parameter; if no keywords are contained, use zero.

- pg_get_functiondef(func_oid)

  Description: Obtains the definition of a function.

  Return type: text.

  Example:

```
gaussdb=# SELECT * FROM pg_get_functiondef(598);
 headerlines |                    definition
-------------+-----------------------------------------------------
           4 | CREATE OR REPLACE FUNCTION pg_catalog.abbrev(inet)+
             |  RETURNS text                                     +
             |  LANGUAGE internal                                +
             |  IMMUTABLE STRICT NOT FENCED NOT SHIPPABLE        +
             | AS $function$inet_abbrev$function$                +
             |
(1 row)
```

- pg_get_function_arguments(func_oid)

  Description: Obtains argument list of function's definition (with default values).

  Return type: text.

  Note: **pg_get_function_arguments** returns the parameter list of a function, in the form it would need to appear in **CREATE FUNCTION**.

- pg_get_function_identity_arguments(func_oid)

  Description: Obtains the parameter list to identify a function (without default values).

  Return type: text.

  Note: **pg_get_function_identity_arguments** returns the parameter list required to identify a function, in the form it would need to appear in **ALTER FUNCTION**. This form omits default values.

- pg_get_function_result(func_oid)

  Description: Obtains the RETURNS clause for a function.

  Return type: text.

  Note: **pg_get_function_result** returns the appropriate **RETURNS** clause for the function.

- pg_get_indexdef(index_oid)

  Description: Obtains the **CREATE INDEX** command for an index.

  Return type: text.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# CREATE TABLE test3 (a int, b int);
gaussdb=# CREATE INDEX test3_b_index ON test3 USING btree(b) TABLESPACE pg_default;
gaussdb=# SELECT pg_get_indexdef(oid) FROM (SELECT oid FROM pg_class WHERE relname =
'test3_b_index') tmp;
                                        pg_get_indexdef
-----------------------------------------------------------------------------------------------------
 CREATE INDEX test3_b_index ON test3 USING ubtree (b) WITH (storage_type=USTORE) TABLESPACE
pg_default
(1 row)
```

- pg_get_indexdef(index_oid, dump_schema_only)

  Description: Obtains the **CREATE INDEX** command for indexes in dump scenarios. In the current version, the value of **dump_schema_only** does not affect the function output.

  Return type: text.

  Example:

```
gaussdb=# SELECT pg_get_indexdef(oid, true) FROM (SELECT oid FROM pg_class WHERE relname =
'test3_b_index') tmp;
                                        pg_get_indexdef
-----------------------------------------------------------------------------------------------------
 CREATE INDEX test3_b_index ON test3 USING ubtree (b) WITH (storage_type=USTORE) TABLESPACE
pg_default
(1 row)
```

- pg_get_indexdef(index_oid, column_no, pretty_bool)

  Description: Obtains the **CREATE INDEX** command for an index, or definition of just one index column when the value of **column_no** is not zero.

  Return type: text.

  Example:

```
gaussdb=# SELECT pg_get_indexdef(oid, 0, false) FROM (SELECT oid FROM pg_class WHERE relname
= 'test3_b_index') tmp;
                                        pg_get_indexdef
-----------------------------------------------------------------------------------------------------
 CREATE INDEX test3_b_index ON test3 USING ubtree (b) WITH (storage_type=USTORE) TABLESPACE
pg_default
(1 row)


gaussdb=# SELECT pg_get_indexdef(oid, 1, false) FROM (SELECT oid FROM pg_class WHERE relname
= 'test3_b_index') tmp;
 pg_get_indexdef
-----------------
 b
(1 row)

gaussdb=# DROP INDEX test3_b_index;
gaussdb=# DROP TABLE test3;
```

  Note: pg_get_idexdef returns a complete CREATE OR REPLACE INDEX statement for a function.

- pg_get_keywords()

  Description: Obtains the list of SQL keywords and their categories.

  Return type: SETOF record.

  Note: **pg_get_keywords** returns a set of records describing the SQL keywords recognized by the server. The **word** column contains the keywords. The **catcode** column contains a category code: **U** for unreserved, **C** for column name, **T** for type or function name, or **R** for reserved. The **catdesc** column contains a possibly-localized string describing the category. The keyword range and type vary depending on the compatibility mode, and the return

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

result of pg_get_keywords also varies. The records returned by the pg_get_keywords function do not contain the options configured in the **disable_keyword_options** parameter. For details about the **disable_keyword_options** parameter, see **Platform and Client Compatibility**.

- pg_get_ruledef(rule_oid)

  Description: Obtains the **CREATE RULE** command for a rule.

  Return type: text.

- pg_get_ruledef(rule_oid, pretty_bool)

  Description: Obtains the **CREATE RULE** command for a rule.

  Return type: text.

- pg_get_userbyid(role_oid)

  Description: Obtains the role name with a given OID.

  Return type: name.

  Note: **pg_get_userbyid** extracts a role's name given its OID.

- pg_check_authid(role_id)

  Description: Checks whether a user exists based on **role_id**.

  Return type: text.

  ```
  gaussdb=# SELECT pg_check_authid(20);
  pg_check_authid
  -----------------
  f
  (1 row)
  ```

- pg_get_viewdef(view_name)

  Description: Obtains the underlying **SELECT** command for a view.

  Return type: text.

- pg_get_viewdef(view_name, pretty_bool)

  Description: Obtains the underlying **SELECT** command for a view. Lines with columns are wrapped to 80 columns if **pretty_bool** is set to **true**.

  Return type: text.

  Note: **pg_get_viewdef** reconstructs the SELECT query that defines a view. Most of these functions come in two forms. When the function has the **pretty_bool** parameter and the value is **true**, it can optionally "pretty-print" the result. The pretty-printed format is more readable. The other one is the default format which is more likely to be interpreted in the same way by future versions. Avoid using pretty-printed output for dump purposes. Passing **false** to the pretty-print parameter generates the same result as a variant without this parameter.

- pg_get_viewdef(view_oid)

  Description: Obtains the underlying SELECT command for a view.

  Return type: text.

- pg_get_viewdef(view_oid, pretty_bool)

  Description: Obtains the underlying **SELECT** command for a view. Lines with columns are wrapped to 80 columns if **pretty_bool** is set to **true**.

  Return type: text.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_get_viewdef(view_oid, wrap_column_int)

  Description: Obtains the underlying **SELECT** command for a view. Lines with columns are wrapped to the specified number of columns and printing is implicit.

  Return type: text.

- pg_get_tabledef(table_oid)

  Description: Obtains the definition of a table based on **table_oid**.

  Return type: text.

  Example:

```
gaussdb=# CREATE TABLE t1 (c1 serial);
gaussdb=# SELECT pg_get_tabledef(oid) FROM (SELECT oid FROM pg_class WHERE relname = 't1')
tmp;
                          pg_get_tabledef
------------------------------------------------------------------------------
 SET search_path = public;                                                   +
 CREATE TABLE t1 (                                                           +
     c1 integer DEFAULT nextval('t1_c1_seq1790791646'::regclass) NOT NULL+
 )                                                                           +
 WITH (orientation=row, compression=no, storage_type=USTORE, segment=off)+
 DISTRIBUTE BY HASH(c1)                                                      +
 TO GROUP group_version1;
(1 row)
```

- pg_get_tabledef(table_name)

  Description: Obtains the definition of a table based on **table_name**.

  Return type: text.

  Example:

```
gaussdb=# SELECT * FROM pg_get_tabledef('t1');
                          pg_get_tabledef
------------------------------------------------------------------------------
 SET search_path = public;                                                   +
 CREATE TABLE t1 (                                                           +
     c1 integer DEFAULT nextval('t1_c1_seq1790791646'::regclass) NOT NULL+
 )                                                                           +
 WITH (orientation=row, compression=no, storage_type=USTORE, segment=off)+
 DISTRIBUTE BY HASH(c1)                                                      +
 TO GROUP group_version1;
(1 row)
```

  Note: **pg_get_tabledef** reconstructs the CREATE statement of the definition of the table, including the table definition, index information, comments, and ILM policy. You need to create the group, schema, tablespace, and server information on which the table object depends. If the ILM policy is inherited from the parent object, pg_get_tabledef does not return the ILM policy creation statement of the data object.

- pg_options_to_table(reloptions)

  Description: Obtains the set of storage option name/value pairs.

  Return type: SETOF record.

  Note: **pg_options_to_table** returns the set of storage option name/value pairs (**option_name**/**option_value**) when **pg_class.reloptions** or **pg_attribute.attoptions** is passed.

- pg_tablespace_databases(tablespace_oid)

  Description: Obtains the set of database OIDs that have objects in the specified tablespace.

  Return type: SETOF oid.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Note: **pg_tablespace_databases** allows a tablespace to be checked. It returns the set of OIDs of databases that have objects stored in the tablespace. If this function returns any rows of data, the tablespace is not empty and cannot be dropped. To display the specific objects in the tablespace, you need to connect to the databases identified by **pg_tablespace_databases** and query the pg_class system catalogs.

- pg_tablespace_location(tablespace_oid)

  Description: Obtains the path in the file system that this tablespace is located in.

  Return type: text.

- pg_typeof(any)

  Description: Obtains the data type of any value.

  Return type: regtype.

  Note: pg_typeof returns the OID of the data type of the value that is passed to it. This can be helpful for troubleshooting or dynamically constructing SQL queries. It is declared that the function returns regtype, which is an OID alias type (see **OID Types**). This means that it is the same as an OID for comparison purposes but displays as a type name.

  Example:

```
gaussdb=# SELECT pg_typeof(33);
 pg_typeof
-----------
 integer
(1 row)

gaussdb=# SELECT typlen FROM pg_type WHERE oid = pg_typeof(33);
 typlen
--------
      4
(1 row)
```

- collation for (any)

  Description: Obtains the collation of the parameter.

  Return type: text.

  Note: The expression **collation for** returns the collation of the value that is passed to it. Example:

```
gaussdb=# SELECT collation for (description) FROM pg_description LIMIT 1;
 pg_collation_for
------------------
 "default"
(1 row)
```

  The value might be quoted and schema-qualified. If no collation is derived for the parameter expression, then a null value is returned. If the argument is not of a collatable data type, then an error message is displayed.

- getdistributekey(table_name)

  Description: Obtains a distribution key for a hash table.

  Return type: text.

  Example:

```
gaussdb=# SELECT getdistributekey('t1');
 getdistributekey
------------------
 c1
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_extension_update_paths(name)

  Description: Returns the version update path of the specified extension. This function can be called only by system administrators.

  Return type: text (source text), text (target text), text (path text)

- pg_get_serial_sequence(tablename, colname)

  Description: Obtains the sequence of the corresponding table name and column name.

  Return type: text.

  Example:

```
gaussdb=# SELECT * FROM pg_get_serial_sequence('t1', 'c1');
   pg_get_serial_sequence
----------------------------
 public.t1_c1_seq1790791646
(1 row)
```

- pg_sequence_parameters(sequence_oid, OUT start_value int16, OUT minimum_value int16, OUT maximum_value int16, OUT increment int16, OUT cycle_option Boolean, OUT order_option Boolean)

  Description: Obtains the parameters of a specified sequence, including the start value, minimum value, maximum value, increment, cycle option, and order option.

  Return type: int16, int16, int16, int16, Boolean, Boolean.

  Example:

```
-- Use the name of the sequence automatically generated when the t1 table is created.
gaussdb=# SELECT oid FROM pg_class WHERE relname = 't1_c1_seq1790791646';
  oid
-------
 17120
(1 row)

-- Use the OID obtained from the preceding query.
gaussdb=# SELECT * FROM  pg_sequence_parameters(17120);
 start_value | minimum_value |     maximum_value     | increment | cycle_option | order_option
-------------+---------------+-----------------------+-----------+--------------+--------------
           1 |             1 | 9223372036854775807 |         1 | f            | f
(1 row)
```

- pgxc_get_variable_info()

  Description: Obtains variable values on the node, including **nodeName**, **nextOid**, **nextXid**, **oldestXid**, **xidVacLimit**, **oldestXidDB**, **lastExtendCSNLogpage**, **startExtendCSNLogpage**, **nextCommitSeqNo**, **latestCompleteXid**, and **startupMaxXid**.

  Return type: set of pg_variable_info

  Example:

```
gaussdb=# SELECT pgxc_get_variable_info( );
                        pgxc_get_variable_info
-----------------------------------------------------------------------
(dn_6004_6005_6006,25617,141396349,2073,20000002073,15808,138111,0,127154152,141396348,104
433004)
(1 row)
```

- gs_get_index_status(schema_name, index_name)

  Description: Obtains the index status information on all nodes, including whether an index can be inserted and whether an index is available. This function is mainly used to check the index status during online index creation or when the index creation fails. The return values include **node_name**, **indisready**, and **indisvalid**. Only when **indisready** and **indisvalid** of indexes

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

on all nodes are set to **true** and the index state is not changed to unusable, the current index is available.

Return type: text, Boolean, Boolean

Example:

```
gaussdb=# CREATE INDEX index1 ON t1(c1);
gaussdb=# SELECT * FROM gs_get_index_status('public', 'index1');
   node_name    | indisready | indisvalid
----------------+------------+------------
 datanode1      | t          | t
 datanode2      | t          | t
 coordinator1   | t          | t
(3 row)

gaussdb=# DROP INDEX index1;
gaussdb=# DROP TABLE t1;
```

Comment Information Functions

- col_description(table_oid, column_number)

  Description: Obtains the comment for a table column.

  Return type: text.

  Note: **col_description** returns the comment for a table column, which is specified by the OID of its table and its column number.

- obj_description(object_oid, catalog_name)

  Description: Obtains the comment for a database object.

  Return type: text.

  Note: The two-parameter form of **obj_description** returns the comment for a database object specified by its OID and the name of the system catalog to which it belongs. For example, **obj_description(123456,'pg_class')** would retrieve the comment for the table with OID 123456. The one-parameter form of **obj_description** requires only the OID.

  obj_description cannot be used for table columns since columns do not have OIDs of their own.

- obj_description(object_oid)

  Description: Obtains the comment for a database object.

  Return type: text.

- shobj_description(object_oid, catalog_name)

  Description: Obtains the comment for a shared database object.

  Return type: text.

  Note: **shobj_description** is used just like **obj_description**, except that the former is used for shared objects. Some system catalogs are global to all databases in the cluster, and the comments for objects in them are stored globally as well.

XIDs and Snapshots

Internal XIDs are 64 bits. **txid_snapshot**, data type used by these functions, stores information about XID visibility at a particular moment in time. **Table 1-118** describes the components.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-118 Snapshot components

Name	Description
xmin	Earliest XID (**txid**) that is still active. All earlier transactions will either be committed and visible, or rolled back.
xmax	First as-yet-unassigned **txid**. All **txids** greater than or equal to this are not yet started as of the time of the snapshot, so they are invisible.
xip_list	Active **txids** at the time of the snapshot. The list includes only those active **txids** between **xmin** and **xmax**; there might be active **txids** higher than **xmax**. A **txid** that is greater than or equal to **xmin** and less than **xmax** and that is not in this list was already completed at the time of the snapshot, and is either visible or rolled back according to its commit status. The list does not include **txids** of subtransactions.

The textual representation of **txid_snapshot** is **xmin:xmax:xip_list**.

For example, **10:20:10,14,15** means **xmin=10, xmax=20, xip_list=10, 14, 15**.

The following functions provide server transaction information in an exportable form. These functions are mainly used to determine which transactions were committed between two snapshots.

- pgxc_is_committed(transaction_id)

  Description: Specifies whether the given XID is committed or ignored. **NULL** indicates the unknown status (it can be running, ready, frozen, or other status). In maintenance mode (when the GUC parameter **xc_maintenance_mode** is set to **on**), the statuses of all transactions with the specified bucket IDs and ordinary tables are traversed. If a status conflict occurs, for example, the status of the same transaction is **committed** in bucket 1 and **rolled back** in bucket 2, an error is reported. Exercise caution when enabling the mode. It is used by maintenance engineers for troubleshooting. Common users should not use the mode.

  Return type: Boolean.

- pgxc_is_committed(transaction_id, bucketid)

  Description: Commits or ignores the given transaction (gxid) if **bucketid** is specified. **NULL** indicates that the status is unknown (running, ready, or frozen). **bucketid** is the physical bucket ID of the hash bucket table. The value is **–1** for an ordinary table. In maintenance mode (when the GUC parameter **xc_maintenance_mode** is set to **on**), if **bucketid** is set to **–1**, the statuses of all transactions with the specified bucket IDs and ordinary tables are traversed. If a status conflict occurs, for example, the status of the same transaction is **committed** in bucket 1 and **rolled back** in bucket 2, an error is reported. Exercise caution when enabling the mode. It is used by maintenance engineers for troubleshooting. Common users should not use the mode.

  Return type: Boolean.

- txid_current()

  Description: Obtains the current XID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: bigint.

- gs_txid_oldestxmin()

  Description: Obtains the minimum XID (specified by **oldestxmin**).

  Return type: bigint.

- txid_current_snapshot()

  Description: Obtains the current snapshot.

  Return type: txid_snapshot.

- txid_snapshot_xip(txid_snapshot)

  Description: Obtains in-progress XIDs in a snapshot.

  Return type: setof bigint.

- txid_snapshot_xmax(txid_snapshot)

  Description: Obtains **xmax** of snapshots.

  Return type: bigint.

- txid_snapshot_xmin(txid_snapshot)

  Description: Obtains **xmin** of snapshots.

  Return type: bigint.

- txid_visible_in_snapshot(bigint, txid_snapshot)

  Description: Specifies whether the XID is visible in a snapshot (do not use subtransaction IDs).

  Return type: Boolean

- get_local_prepared_xact()

  Description: Obtains the two-phase residual transaction information of the current node, including the XID, GID of the two-phase transaction, prepared time, owner OID, database OID, and node name of the current node.

  Return types: xid, text, timestamptz, oid, and text.

- get_remote_prepared_xacts()

  Description: Obtains the two-phase residual transaction information of all remote nodes, including the XID, GID of the two-phase transaction, prepared time, owner name, database name, and node name.

  Return types: xid, text, timestamptz, name, and text.

- global_clean_prepared_xacts(text, text)

  Description: Concurrently cleans two-phase residual transactions. Only the gs_clean tool can call this function for cleaning. In other situations, **false** is returned.

  Return type: Boolean

- pgxc_stat_get_wal_senders()

  Description: Returns the sent logs of all primary DNs and the received logs of their corresponding standby DNs in the cluster. Only users with the system admin or monitor admin permission can use this function.

  Return values: See **Table 1-119**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-119 pgxc_stat_get_wal_senders parameters

Column	Description
nodename	Instance name.
sender_pid	PID of the thread for sending logs.
local_role	Instance role.
peer_role	Role of the instance on the receiver.
peer_state	Status of the instance on the receiver.
state	Synchronization status between instances.
sender_sent_location	Location where the sender sends logs.
sender_write_location	Location where the sender writes logs.
sender_flush_location	Location where the sender flushes logs to disk.
sender_replay_location	Log location of the current instance. If the DN is a primary DN, the location is the value of **sender_flush_location**. Otherwise, the location where the current instance log is replayed is used.
receiver_received_location	Location where the receiver receives logs.
receiver_write_location	Location where the receiver writes logs.
receiver_flush_location	Location where the receiver flushes logs to disk.
receiver_replay_location	Location where the receiver replays logs.

- pgxc_stat_get_wal_senders_status()

  Description: Returns the receiving status of transaction logs on all nodes. Only users with the system admin or monitor admin permission can use this function.

  Return values: See **Table 1-120**.

Table 1-120 pgxc_stat_get_wal_senders_status parameters

Column	Description
nodename	Name of the primary node.
source_ip	IP address of the primary node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
source_port	Port of the primary node.
dest_ip	IP address of the standby node.
dest_port	Port of the standby node.
sender_pid	PID of the sending thread.
local_role	Type of the primary node.
peer_role	Type of the standby node.
peer_state	Status of the standby node.
state	WAL sender status.
sender_sent_location	Sending position of the primary node.
sender_write_location	Writing position of the primary node.
sender_replay_location	Redo position of the primary node.
receiver_received_location	Receiving position of the standby node.
receiver_write_location	Writing position of the standby node.
receiver_flush_location	Flushing location of the standby node.
receiver_replay_location	Redo location of the standby node.

- gs_get_next_xid_csn()

  Description: Returns the values of **next_xid** and **next_csn** on all nodes globally.

  For details about the return values, see **Table 1-121**.

  Table 1-121 gs_get_next_xid_csn columns

Column	Description
nodename	Node name.
next_xid	ID of the next transaction on the current node.
next_csn	Next CSN of the current node.

- pg_control_system()

  Description: Returns the status of the system control file.

  Return type: SETOF record.

- pg_control_checkpoint()

  Description: Returns the system checkpoint status.

  Return type: SETOF record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- get_prepared_pending_xid()

  Description: Returns **nextxid** when restoration is complete.

  Return type: text.

- pg_clean_region_info()

  Description: Clears the region map.

  Return type: character varying

- pg_get_replication_slot_name()

  Description: Obtains the slot name.

  Return type: text.

- pg_get_running_xacts()

  Description: Obtains running **xact**.

  Return type: handle integer, gxid xid, state tinyint, node text, xmin xid, vacuum Boolean, timeline bigint, prepare_xid xid, pid bigint, next_xid xid, dbid oid

- pg_get_variable_info()

  Description: Obtains the shared memory variable cache.

  Return type: node_name text, nextOid oid, nextXid xid, oldestXid xid, xidVacLimit xid, oldestXidDB oid, lastExtendCSNLogpage xid, startExtendCSNLogpage xid, nextCommitSeqNo xid, latestCompletedXid xid, and startupMaxXid xid

- pg_get_xidlimit()

  Description: Obtains XID information from the shared memory.

  Return type: nextXid xid, oldestXid xid, xidVacLimit xid, xidWarnLimit xid, xidStopLimit xid, xidWrapLimit xid, oldestXidDB oid

- pg_relation_compression_ratio(text)

  Description: Queries the compression ratio of a table. By default, **1.0** is returned.

  Parameter: text.

  Return type: real.

- pg_relation_with_compression(text)

  Description: Specifies whether a table is compressed.

  Parameter: text.

  Return type: Boolean.

- pg_stat_file_recursive(location text[, get_turbo_info boolean])

  Description: Lists all files in a directory. Only the initial user can execute this function.

  Parameter: **location text[, get_turbo_info boolean]**.

  Return types: path text, filename text, size bigint, and isdir boolean[, isturbocompress boolean].

- pg_stat_get_activity_for_temptable()

  Description: Returns records of background threads related to the temporary table.

  Return type: datid oid, timelineid integer, tempid integer, and sessionid bigint

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_stat_get_activity_ng(pid bigint)

  Description: Returns records of background threads related to nodegroup.

  Parameter: pid bigint

  Return type: datid oid, pid bigint, sessionid bigint, and node_group text
- pg_stat_get_cgroup_info()

  Description: Returns Cgroup information.

  Return type: cgroup_name text, percent integer, usage_percent integer, shares bigint, usage bigint, cpuset text, relpath text, valid text, node_group text
- pg_stat_get_realtime_info_internal(oid, oid, bigint, cstring, oid)

  Description: Returns real-time information. Currently, this API is unavailable. **FailedToGetSessionInfo** is returned.

  Parameter: oid, oid, bigint, cstring, oid

  Return type: text.
- pg_stat_get_wlm_session_iostat_info()

  Description: Returns the session load I/O information.

  Return types: threadid bigint, maxcurr_iops integer, mincurr_iops integer, maxpeak_iops integer, minpeak_iops integer, iops_limits integer, io_priority integer, and curr_io_limits integer.
- pg_test_err_contain_err(integer)

  Description: Tests the error type and return information.

  Parameter: integer

  Return type: void
- pv_session_memory_detail_tp()

  Description: Returns the memory usage of the session. For details, see **pv_session_memory_detail**.

  Return types: sessid text, sesstype text, contextname text, level smallint, parent text, totalsize bigint, freesize bigint, and usedsize bigint.
- gs_get_table_distribution(table_name text, schema_name text)

  Description: Returns the distribution of table data on each DN.

  Parameter: table_name text, schema_name text

  Return type: text.
- pv_builtin_functions()

  Description: Displays information about all built-in system functions.

  Return types: proname name, pronamespace oid, proowner oid, prolang oid, procost real, prorows real, provariadic oid, protransform regproc, proisagg boolean, proiswindow boolean, prosecdef boolean, proleakproof boolean, proisstrict boolean, proretset boolean, provolatile "char", pronargs smallint, pronargdefaults smallint, prorettype oid, proargtypes oidvector, proallargtypes integer[], proargmodes "char"[], proargnames text[], proargdefaults pg_node_tree, prosrc text, probin text, proconfig text[], proacl aclitem[], prodefaultargpos int2vector, fencedmode boolean, proshippable boolean, propackage boolean, and oid oid.
- pv_thread_memory_detail()

  Description: Returns the memory information of each thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return types: threadid text, tid bigint, thrdtype text, contextname text, level smallint, parent text, totalsize bigint, freesize bigint, and usedsize bigint.

- pg_shared_memory_detail()

  Description: Returns usage information about all generated shared memory contexts. For details about each column, see **DBE_PERF.SHARED_MEMORY_DETAIL**.

  Return types: contextname text, level smallint, parent text, totalsize bigint, freesize bigint, and usedsize bigint.

- pgxc_get_running_xacts()

  Description: Returns information about running transactions on each node in the cluster. The column content is the same as that in the PGXC_RUNNING_XACTS view (see **PGXC_RUNNING_XACTS**). Only users with the system admin or monitor admin permission can view the information.

  Return type: SETOF record

- pgxc_snapshot_status()

  Description: Returns the status of key memory information in the GTM in GTM mode for fault locating. GTM-Free and GTM-Lite do not support this function.

  Return type: xmin xid, xmax xid, xcnt int, oldestxmin xid, next_xid xid, timeline int, active_thread_num int, max_active_thread_num int, snapshot_num int, snapshot_totalsize bigint

  For details about the return values, see **Table 1-122**.

Table 1-122 pgxc_snapshot_status return parameters

Column	Description
xmin	Minimum active XID in the GTM.
xmax	Largest XID committed in the GTM plus 1. Transactions whose IDs are greater than or equal to this value are active.
xcnt	Number of active transactions in the GTM.
oldestxmin	ID of the earliest accessed transaction in the GTM.
next_xid	ID of the next transaction allocated by the GTM.
timeline	Current timeline in the GTM.
active_thread_num	Number of active worker threads in the GTM.
max_active_thread_num	Peak number of worker threads in the GTM within one minute.
snapshot_num	Number of snapshots delivered by the GTM within one minute.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
snapshot_totalsize	Total size of snapshots delivered by the GTM within one minute.

- get_gtm_lite_status()

  Description: Returns the backup XID and CSN in the GTM for fault locating. This system function is not supported in GTM-Free mode.

  Return values: See **Table 1-123**.

  Table 1-123 get_gtm_lite_status return parameters

Column	Description
backup_xid	Backup XID in the GTM.
csn	Latest CSN issued by the GTM.

- adm_hist_snapshot_func()

  Description: Returns information about the snapshot execution time. To access this function, set the **enable_wdr_snapshot** parameter to **on** and obtain the permission to access the **snapshot schema**, **snapshot table**, and **tables_snap_timestamp** tables.

  Return types: snap_id bigint, dbid oid, begin_interval_time timestamp(3), end_interval_time timestamp(3), flush_elapsed interval day(5) to second(1), begin_interval_time_tz timestamp(3) with time zone, and end_interval_time_tz timestamp(3) with time zone.

- global_sql_patch_func()

  Description: SQL patch information on each global node, which is used to return the result of the global_sql_patch view.

  Return values: See **Table 1-124**.

  Table 1-124 global_sql_patch_func return parameters

Name	Type	Description
node_name	text	Name of the node where the SQL patch is located.
patch_name	name	Patch name.
unique_sql_id	bigint	Global unique query ID.
owner	oid	ID of the user who creates the patch.
enable	Boolean	Determines whether the patch takes effect.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
status	"char"	Specifies whether the SQL patch is created based on the unique SQL ID (value: **'d'**) or SQL hash value (value: **'h'**).
abort	Boolean	Determines whether the value is **AbortHint**.
hint_string	text	Hint text.
description	text	Patch description.
parent_unique_sql_id	bigint	Globally unique ID of the outer statement of the SQL statement for which the patch takes effect. The value of this parameter is **0** for statements outside a stored procedure. For statements inside the stored procedure, the value of this parameter is the globally unique ID of the statement that calls the stored procedure.
original_query	text	Original statement.

- gs_get_current_version()

  Description: Returns the current compilation mode based on the current compilation macro. **'M'** is returned.

  Return type: char.

- gs_get_kernel_info()

  Description: Global transaction information on each primary CN and DN.

  Return values: See **Table 1-125**.

Table 1-125 gs_get_kernel_info columns

Name	Type	Description
node_name	text	Node name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
module	text	Module name, including: • **XACT** (transaction module) • **STANDBY** (standby module) • **UNDO** (undo module) • **HOTPATH** (hot patch module) • **FULL_SQL** • **LOCK** (lock module) • **SQL** • **SEC** (security module)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
name	text	Name of the key data in the memory state, including: • **startup_max_xid** (maximum XID when a thread is started) • **recent_local_xmin** (minimum XID of a local active transaction) **recent_global_xmin** (minimum XID of a global active transaction) • **standby_xmin** (minimum XID of active transactions on the standby node) • **standby_redo_cleanup_xmin** (minimum XID of cleanup logs during redo on the standby node) • **standby_redo_cleanup_xmin_lsn** (LSN of the minimum XID of cleanup logs during redo on the standby node) • **local_csn_min** (minimum CSN of local active transactions) • **replication_slot_xmin** (minimum XID of replication slots) • **replication_slot_catalog_xmin** (minimum XID of catalog replication slots) • **global_recycle_xid** (minimum XID of global undo recycling transactions) • **global_frozen_xid** (minimum XID of global frozen transactions) • **recent_xmin** (minimum XID of active transactions in the current session) • **next_csn** (CSN of the next transaction) • **hotpatch_additional_info** (reserved column for hot patches) • **stmt_session_discard_records** (data volume discarded by full SQL statements due to full slots supported by the kernel) • **stmt_shm_flush_discard_records** (data volume discarded by full SQL statements due to full ringbuf supported by the kernel) • **idle_in_trans_timeout_records** (accumulated data volume of idle transaction timeout counts supported by the kernel) • **total_lock_usage** (**LOCK**: number of locks applied for through the primary lock table)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
		• **total_lock_usage_rate** (**LOCK**: usage of the prim+ary lock tablespace) • **fastpath_strong_relation_locks_count** (**LOCK**: partition where FASTPATH strong lock conflicts exist and the number of strong locks corresponding to the partition) • **fastpath_failure_rate** (**LOCK**: percentage of locks that cannot be applied for through FASTPATH due to strong lock conflicts in the corresponding partition) • **lock_acquire_lack_fastpath_count** (**LOCK**: total number of times that locks fail to be applied for through FASTPATH because FASTPATH slots are used up) • **n_soft_parse** (**SQL**: number of soft parsing times) • **n_hard_parse** (**SQL**: number of hard parsing times) • **hsm_access_success_count** (**SEC**: number of successful accesses to the KMS) • **hsm_access_failure_count** (**SEC**: number of failed accesses to the KMS)
value	text	Value of the key data in the memory state.

Example:

```
gaussdb=# SELECT * FROM gs_get_kernel_info();
  node_name   | module  |              name                      |  value
--------------+---------+----------------------------------------+---------
 coordinator1 | XACT    | startup_max_xid                        | 16488
 coordinator1 | XACT    | recent_local_xmin                      | 16504
 coordinator1 | XACT    | recent_global_xmin                     | 16503
 coordinator1 | STANDBY | standby_xmin                           | 0
 coordinator1 | STANDBY | standby_redo_cleanup_xmin              | 0
 coordinator1 | STANDBY | standby_redo_cleanup_xmin_lsn          | 0/0
 coordinator1 | XACT    | local_csn_min                          | 6014225
 coordinator1 | XACT    | replication_slot_xmin                  | 0
 coordinator1 | XACT    | replication_slot_catalog_xmin          | 0
 coordinator1 | UNDO    | global_recycle_xid                     | 16501
 coordinator1 | XACT    | global_frozen_xid                      | 0
 coordinator1 | XACT    | recent_xmin                            | 16504
 coordinator1 | XACT    | next_csn                               | 6014226
 coordinator1 | HOTPATH | hotpatch_additional_info               |
 coordinator1 | FULL_SQL| stmt_session_discard_records           | 0
 coordinator1 | FULL_SQL| stmt_shm_flush_discard_records         | 0
 coordinator1 | XACT    | idle_in_trans_timeout_records          | 0
 coordinator1 | LOCK    | total_lock_usage                       | 0
 coordinator1 | LOCK    | total_lock_usage_rate                  | 0.00%
 coordinator1 | LOCK    | fastpath_strong_relation_locks_count |
 coordinator1 | LOCK    | fastpath_failure_rate                  | 0.00%
 coordinator1 | LOCK    | lock_acquire_lack_fastpath_count       | 0
 coordinator1 | SQL     | n_soft_parse                           | 21809
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
coordinator1 | SQL     | n_hard_parse                      | 234111
coordinator1 | SEC     | hsm_access_success_count          | 0
coordinator1 | SEC     | hsm_access_failure_count          | 0
datanode1    | XACT    | startup_max_xid                   | 16488
datanode1    | XACT    | recent_local_xmin                 | 15805
datanode1    | XACT    | recent_global_xmin                | 15805
datanode1    | STANDBY | standby_xmin                      | 0
datanode1    | STANDBY | standby_redo_cleanup_xmin         | 0
datanode1    | STANDBY | standby_redo_cleanup_xmin_lsn     | 0/0
datanode1    | XACT    | local_csn_min                     | 6014225
datanode1    | XACT    | replication_slot_xmin             | 0
datanode1    | XACT    | replication_slot_catalog_xmin     | 0
datanode1    | UNDO    | global_recycle_xid                | 15805
datanode1    | XACT    | global_frozen_xid                 | 0
datanode1    | XACT    | recent_xmin                       | 15805
datanode1    | XACT    | next_csn                          | 6014226
datanode1    | HOTPATH | hotpatch_additional_info          |
datanode1    | FULL_SQL| stmt_session_discard_records      | 0
datanode1    | FULL_SQL| stmt_shm_flush_discard_records    | 0
datanode1    | XACT    | idle_in_trans_timeout_records     | 0
datanode1    | LOCK    | total_lock_usage                  | 0
datanode1    | LOCK    | total_lock_usage_rate             | 0.00%
datanode1    | LOCK    | fastpath_strong_relation_locks_count |
datanode1    | LOCK    | fastpath_failure_rate             | 0.00%
datanode1    | LOCK    | lock_acquire_lack_fastpath_count  | 0
datanode1    | SQL     | n_soft_parse                      | 0
datanode1    | SQL     | n_hard_parse                      | 8958
datanode1    | SEC     | hsm_access_success_count          | 0
datanode1    | SEC     | hsm_access_failure_count          | 0
(52 row)
```

- gs_get_single_node_group(object_type, object_name1, object_name2, queryonly)

  Description: Obtains the default single-node information of the target table or schema.

  Parameters: See **Table 1-126**.

  Return type: text.

  Table 1-126 gs_get_single_node_group parameters

Parameter	Type	Description	Value Range
object_type	text	Object type.	• **schema** • **table**
object_name1	text	Schema name.	Name of an existing schema.
object_name2	text	Table name.	• Table name. • **NULL** (default).
queryonly	bool	Specifies whether to query only.	• **true** (default). • **false**.

  Example:

```
gaussdb=# CREATE SCHEMA s1 SET NODE GROUP sys_single_group1;

gaussdb=# CREATE TABLE single_table(c1 int, c2 int) TO GROUP sys_single_group2;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT gs_get_single_node_group('schema', 's1');
 gs_get_single_node_group
--------------------------
 sys_single_group1
(1 row)

gaussdb=# SELECT gs_get_single_node_group('table', 'public', 'single_table');
 gs_get_single_node_group
--------------------------
 sys_single_group2
(1 row)

gaussdb=# DROP TABLE single_table;
DROP TABLE
gaussdb=# DROP SCHEMA s1 CASCADE;
DROP SCHEMA
```

1.6.27 System Administration Functions

1.6.27.1 Configuration Setting Functions

Configuration setting functions are used for querying and modifying GUC parameters.

- current_setting(setting_name)

  Description: Obtains the current value of **setting_name** by query. It is equivalent to the SQL statement SHOW.

  Return type: text

  Example:

  ```
  gaussdb=# SELECT current_setting('datestyle');

   current_setting
  -----------------
   ISO, MDY
  (1 row)
  ```

- set_working_grand_version_num_manually(tmp_version)

  Description: Upgrades new features of GaussDB by switching the authorization version.

  Return type: void

- shell_in(type)

  Description: Processes the input route of the shell type (that is, the type that has not been filled).

  Return type: void

- shell_out(type)

  Description: Processes the output route of the shell type (that is, the type that has not been filled).

  Return type: void

- set_config(setting_name, new_value, is_local)

  Description: Sets a specified configuration parameter and returns the parameter value. This function does not allow you to set the **ROLE** parameter.

  Parameters:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - **setting_name**: name of the parameter to be set.
  - **new_value**: value of the parameter to be set.
  - **is_local**: application scope of the parameter value.
    - If **is_local** is **true**, the **new_value** applies only to the current transaction.
    - If **is_local** is **false**, the **new_value** applies to the current session, which is equivalent to the SQL statement SET.

  Return type: text

  Example:

```
gaussdb=# SELECT set_config('log_statement_stats', 'off', false);

 set_config
------------
 off
(1 row)
```

1.6.27.2 Universal File Access Functions

Universal file access functions provide local access APIs for files on a database server. Only files in the database cluster directory and the **log_directory** directory can be accessed. A relative path is used for files in the cluster directory, and a path matching the **log_directory** configuration setting is used for log files. Only the database initial user can use these functions.

- pg_ls_dir(dirname text)

  Description: Lists files in a directory.

  Return type: setof text

  Note: pg_ls_dir returns all the names in the specified directory, except the special entries "." and "..".

  Example:

```
gaussdb=# SELECT pg_ls_dir('./');
      pg_ls_dir
----------------------
 .gaussdb.conf.swp
 gaussdb.conf
 pg_tblspc
 PG_VERSION
 gs_ident.conf
 core
 server.crt
 pg_serial
 pg_twophase
 gaussdb.conf.lock
 pg_stat_tmp
 pg_notify
 pg_subtrans
 pg_ctl.lock
 pg_xlog
 pg_clog
 base
 pg_snapshots
 postmaster.opts
 postmaster.pid
 server.key.rand
 server.key.cipher
 pg_multixact
 pg_errorinfo
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 server.key
 gs_hba.conf
 pg_replslot
 .gs_hba.conf.swp
 cacert.pem
 gs_hba.conf.lock
 global
 gaussdb.state
(32 rows)
```

- pg_read_file(filename text, offset bigint, length bigint)

  Description: Returns the content of a text file. It can return a maximum of length bytes from *offset*. The actual value is smaller than the length if the end of the file is reached at first. If *offset* is negative, it is the length rolled back from the file end. If *offset* and *length* are omitted, the entire file is returned.

  Return type: text

  Example:

```
gaussdb=# SELECT pg_read_file('postmaster.pid',0,100);
           pg_read_file
----------------------------------------
 53078                            +
/srv/BigData/testdir/data1/coordinator+
 1500022474                       +
 8000                            +
 /var/run/FusionInsight           +
 localhost                        +
  2
(1 row)
```

- pg_read_binary_file(filename text [, offset bigint, length bigint,missing_ok boolean[, compress boolean]])

  Description: Returns the content of a binary file. The function is similar to pg_read_file, but the return value is of the bytea type. Therefore, no encoding check is performed. This function can be used with the convert_from function to read a file in a specified encoding.

  Return type: bytea

```
gaussdb=# SELECT convert_from(pg_read_binary_file('filename'), 'UTF8');
```

- pg_read_binary_file_blocks(path text, block_begin bigint, block_count bigint, bitmap bigint)

  Description: Reads the binary content of a specified page area in a cross-page transparent compression file. This function can be called only by the initial database user.

  Return type: bytea

- pg_stat_file(filename text)

  Description: Returns a record with the status information of the text file, including the file size, last access timestamp, last modification timestamp, last file status change timestamp, and a **Boolean** value indicating if it is a directory.

  Return type: record

  Typical usages are as follows:

```
gaussdb=# SELECT * FROM pg_stat_file('filename');
gaussdb=# SELECT (pg_stat_file('filename')).modification;
```

  Example:

```
gaussdb=# SELECT convert_from(pg_read_binary_file('postmaster.pid'), 'UTF8');
           convert_from
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
---------------------------------------
 4881                                 +
 /srv/BigData/gaussdb/data1/coordinator+
 1496308688                           +
 25108                                +
 /opt/huawei/Bigdata/gaussdb/gaussdb_tmp +
 *                                    +
  25108001  43352069                  +

(1 row)
gaussdb=# SELECT * FROM pg_stat_file('postmaster.pid');

 size |         access         |       modification       |         change
| creation | isdir
------+------------------------+--------------------------+--------------------------
+----------+-------
  117 | 2017-06-05 11:06:34+08 | 2017-06-01 17:18:08+08 | 2017-06-01 17:18:08+08
|          | f
(1 row)
gaussdb=# SELECT (pg_stat_file('postmaster.pid')).modification;
      modification
-------------------------
 2017-06-01 17:18:08+08
(1 row)
```

1.6.27.3 Server Signal Functions

Server signal functions send control signals to other server threads. Only the system administrator has the permission to execute the following functions:

- pg_cancel_backend(pid int)

  Description: Cancels a statement that is being executed by a backend thread.

  Return type: Boolean

  Note: pg_cancel_backend sends a query cancellation (SIGINT) signal to the backend thread identified by **pid**. The PID of an active backend thread can be found in the **pid** column of the PG_STAT_ACTIVITY view (see **PG_STAT_ACTIVITY**), or can be found by listing the database thread using **ps** on the server. A user with the SYSADMIN permission, the owner of the database connected to the backend thread, the owner of the backend thread, or a user who inherits permissions of the built-in role gs_role_signal_backend can use this function.

- pg_cancel_session(pid bigint, sessionid bigint)

  Description: Cancels the statement that is being executed by an active session in thread pool mode.

  Return type: Boolean

  Note: The input parameters of pg_cancel_session can be queried using the **pid** and **sessionid** columns in pg_stat_activity. The input parameters can be used to cancel the statements that are being executed by active sessions in thread pool mode. When the input parameters **pid** and **sessionid** are the same and both are thread IDs, the function of pg_cancel_backend is the same as that of pg_cancel_backend.

- pg_cancel_invalid_query()

  Description: Cancels the invalid query of a backend.

  Return type: Boolean

  Note: Only the system administrator has the permission to cancel queries that are running in the backend of a degraded GTM.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_reload_conf()

  Description: Sends a SIGHUP signal to the server to trigger all server threads to reload their configuration files.

  Return type: Boolean

- pg_rotate_logfile()

  Description: Rotates the log files of the server.

  Return type: Boolean

  Note: pg_rotate_logfile sends a signal to the log file manager, instructing the manager to immediately switch to a new output file. This function works only when redirect_stderr is used for log output. Otherwise, no log file manager sub-thread is generated.

- pg_terminate_session(pid bigint, sessionid bigint)

  Description: Terminates a backend session in thread pool mode.

  Return type: Boolean

  Note: The input parameters of this function can be queried using the **pid** and **sessionid** fields in pg_stat_activity. A user with the SYSADMIN permission, the owner of the database connected to the session, the owner of the session, or a user who inherits permissions of the built-in role gs_role_signal_backend has the permission to use this function. If the target session fails to be terminated after this function is executed, the socket connection between the session and the client is forcibly closed.

  NOTICE

  - This function has two execution modes based on the relationship between the input parameters **pid** and **sessionid**. The details are as follows:
    - If the parameter values are the same (**pid** = **sessionid**, and the value is the thread ID), the function terminates threads by thread. Non-thread pool threads and active thread pool threads can be terminated, but inactive thread pool threads cannot be terminated.
    - If the parameter values are different (**pid** ≠ **sessionid**), the function terminates sessions by session. Active sessions can be terminated, and inactive sessions and their client socket connections can be cleared.
  - In the following scenarios, the target thread or session may not be terminated immediately after this function is executed:
    - The thread or session is waiting for a lock.
    - The thread or session is blocked or does not respond due to a hardware fault (for example, a storage exception, network interruption, or resource unavailability).
    - The session is in the network I/O (for example, **recv/send**) state.
    - The session is in the idle state when the system CPU load is high or the thread pool is full.

- pg_terminate_active_session_socket(pid int64, sessionid int64)

  Description: Closes the socket connection between an active session and the client.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: Boolean

Note: Each of these functions returns **true** if they are successful and **false** otherwise. A user with the SYSADMIN permission, the owner of the database connected to the backend thread, the owner of the backend thread, or a user who inherits permissions of the built-in role gs_role_signal_backend can use this function.

- pg_terminate_backend(pid int)

  Description: Terminates a backend thread. Only the system administrator and thread owner can use this function.

  Return type: Boolean

  Note: Each of these functions returns **true** if they are successful and **false** otherwise. A user with the SYSADMIN permission, the owner of the database connected to the backend thread, the owner of the backend thread, or a user who inherits permissions of the built-in role gs_role_signal_backend can use this function. If the target session fails to be terminated after this function is executed, the socket connection between the session and the client is forcibly closed.

  NOTICE

  - This function can terminate non-thread pool threads and active thread pool threads, but cannot terminate inactive thread pool threads.
  - In the following scenarios, the target thread may not be terminated immediately after this function is executed:
    - The thread is waiting for a lock.
    - The thread is blocked or does not respond due to a hardware fault (for example, a disk exception or resource unavailability).

  Example:

```
gaussdb=# SELECT pid from pg_stat_activity;
       pid
-----------------
 140657876268816
(1 rows)

gaussdb=# SELECT pg_terminate_backend(140657876268816);
 pg_terminate_backend
----------------------
 t
(1 row)
```

- gs_cancel_invalid_gtm_query(cstring)

  Description: Disconnects all sessions connected to the GTM with a specified address. The function parameter is a comma-separated string. The format of each separated substring is IP[:port] (Content in square parentheses ([]) is optional.).

  Return type: void

  Remarks: Only the system administrator has the permission to execute this function. This function is automatically called by the CM component in a specific process. You are advised not to manually call this function.

- terminate_session_has_temp_file()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Interrupts all sessions where temporary files are being opened.

Return type: void

Remarks: Only the system administrator has the permission to execute this function. This function is automatically called by the CM component in a specific process. You are advised not to manually call this function.

1.6.27.4 Backup and Restoration Control Functions

Backup Control Functions

Backup control functions help with online backup.

- pg_create_restore_point(name text)

  Description: Creates a named point for performing a restore (requires an administrator role).

  Return type: text

  Note: pg_create_restore_point creates a named transaction log record that can be used as a restoration target, and returns the corresponding transaction log location. The given name can then be used with **recovery_target_name** to specify the point up to which restoration will proceed. Avoid creating multiple restoration points with the same name, since restoration will stop at the first one whose name matches the restoration target.

- pg_current_xlog_location()

  Description: Obtains the write position of the current transaction log.

  Return type: text

  Note: pg_current_xlog_location displays the write position of the current transaction log in the same format as those of the previous functions. Read-only operations do not require SYSADMIN permissions.

- pg_current_xlog_insert_location()

  Description: Obtains the insert position of the current transaction log.

  Return type: text

  Note: pg_current_xlog_insert_location displays the insert position of the current transaction log. The insertion point is the logical end of the transaction log at any instant, while the write location is the end of what has been written out from the server's internal buffers. The write position is the end that can be detected externally from the server. This operation can be performed to archive only some of completed transaction log files. The insert position is mainly used for commissioning the server. Read-only operations do not require SYSADMIN permissions.

- gs_current_xlog_insert_end_location()

  Description: Obtains the insert position of the current transaction log.

  Return type: text

  Note: **gs_current_xlog_insert_end_location** displays the insert position of the current transaction log.

- pg_start_backup(label text [, fast boolean])

  Description: Starts to perform online backup. (An administrator, replication role, or O&M administrator must enable **operation_mode**.) Label strings

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

starting with **gs_roach** are reserved and can be used only by the internal backup tool GaussRoach.

Return type: text

Note: pg_start_backup receives a user-defined backup label (usually the name of the position where the backup dump file is stored). This function writes a backup label file to the data directory of the database cluster and then returns the starting position of backed up transaction logs in text mode. This function must be used together with pg_stop_backup(). If this function is called independently, backup_label remains. During WAL replay, the checkpoint is read based on backup_label. If the WAL file corresponding to the checkpoint has been reclaimed, the database cannot be started.

```
gaussdb=# SELECT pg_start_backup('label_goes_here',true);
 pg_start_backup
-----------------
 0/3000020
(1 row)
```

- pg_stop_backup()

  Description: Completes online backup. **operation_mode** must be enabled by an administrator role, replication role, or OPRADMIN.

  Return type: text

  Note: pg_stop_backup deletes the label file created by pg_start_backup and creates a backup history file in the transaction log archive area. The history file includes the label given to pg_start_backup, the start and end transaction log locations for the backup, and the start and end time of the backup. The return value is the backup's ending transaction log location. After the end position is calculated, the insert position of the current transaction log automatically goes ahead to the next transaction log file. The ended transaction log file can be immediately archived so that backup is complete.

- pg_switch_xlog()

  Description: Switches to a new transaction log file An administrator or O&M administrator must enable **operation_mode**.

  Return type: text

  Note: pg_switch_xlog moves to the next transaction log file so that the current log file can be archived (if continuous archive is used). The return value is the ending transaction log location + 1 within the just-completed transaction log file. If there has been no transaction log activity since the last transaction log switchover, pg_switch_xlog does not move but returns the start location of the transaction log file currently in use.

- pg_xlogfile_name(location text)

  Description: Converts the position string in a transaction log to a file name.

  Return type: text

  Note: pg_xlogfile_name extracts only the transaction log file name. If the given transaction log position is the transaction log file border, a transaction log file name will be returned for both the two functions. This is usually the desired behavior for managing transaction log archiving, since the preceding file is the last one that currently needs to be archived.

- pg_xlogfile_name_offset(location text)

  Description: Converts the position string in a transaction log to a file name and returns the byte offset in the file.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: text, integer

Note: pg_xlogfile_name_offset can extract transaction log file names and byte offsets from the returned results of the preceding functions. Example:

```
gaussdb=# SELECT * FROM pg_xlogfile_name_offset(pg_stop_backup());
NOTICE:  pg_stop_backup cleanup done, waiting for required WAL segments to be archived
NOTICE:  pg_stop_backup complete, all required WAL segments have been archived
        file_name         | file_offset
--------------------------+-------------
000000010000000000000003 |         272
(1 row)
```

- pg_xlog_location_diff(location text, location text)

  Description: Calculates the difference in bytes between two transaction log locations.

  Return type: numeric

- pg_cbm_start_tracked_location()

  Description: Queries the start LSN parsed by CBM.

  Return type: text

- pg_cbm_tracked_location()

  Description: Queries the LSN parsed by CBM.

  Return type: text

- pg_cbm_get_merged_file(startLSNArg text, endLSNArg text)

  Description: Combines CBM files within the specified LSN range into one and returns the name of the combined file.

  Return type: text

  Note: Only SYSADMIN or OPRADMIN can obtain the CBM combination file.

- pg_cbm_get_changed_block(startLSNArg text, endLSNArg text)

  Description: Merges CBM files within the specified LSN range into a table and returns records in this table.

  Return type: records

  Note: The table columns returned by pg_cbm_get_changed_block include the start LSN, end LSN, tablespace OID, database OID, table relfilenode, table fork number, whether the table is a system catalog, whether the table is dropped, whether the table is created, whether the table is truncated, number of pages in the truncated table, number of modified pages, and list of modified page numbers.

- gs_cbm_get_changed_block(startLSNArg text, endLSNArg text)

  Description: Merges CBM files within the specified LSN range into a table and returns records in this table.

  Return type: records

  Note: The table columns returned by gs_cbm_get_changed_block include the LSNs where the merge starts and ends, tablespace OID, database OID, relfilenode of the table, fork number of the table, whether the table is a system catalog, whether the table is dropped, whether the table is created, whether the table is truncated, number of pages after the table is truncated, number of modified pages, list of modified page numbers, and XID of the relevant transaction (only for the operations of creating and dropping databases or tablespaces).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_cbm_recycle_file(targetLSNArg text)

  Description: Deletes the CBM files that are no longer used and returns the first LSN after the deletion.

  Return type: text

- pg_cbm_force_track(targetLSNArg text,timeOut int)

  Description: Forcibly executes the CBM trace to the specified Xlog position and returns the Xlog position of the actual trace end point.

  Return type: text

- pg_enable_delay_ddl_recycle()

  Description: Enables DDL delay and returns the Xlog position of the enabling point. An administrator or O&M administrator must enable **operation_mode**.

  Return type: text

- pg_disable_delay_ddl_recycle(barrierLSNArg text, isForce bool)

  Description: Disables DDL delay and returns the Xlog range where DDL delay takes effect. An administrator or O&M administrator must enable **operation_mode**.

  Return type: records

- pg_enable_delay_xlog_recycle()

  Description: Enables the Xlog recycling delay function for CN recovery. An administrator or O&M administrator must enable **operation_mode**.

  Return type: void

- pg_disable_delay_xlog_recycle()

  Description: Disables the Xlog recycling delay function for CN recovery. An administrator or O&M administrator must enable **operation_mode**.

  Return type: void

- pg_cbm_rotate_file(rotate_lsn text)

  Description: Forcibly switches the file after the CBM parses **rotate_lsn**. This function is called during the build process.

  Return type: void

- gs_roach_stop_backup(backupid text)

  Description: Stops a backup started by the internal backup tool GaussRoach. It is similar to the **pg_stop_backup system** function but is more lightweight.

  Return type: text. The content is the insertion position of the current log.

  Note: To call this function, users must possess the SYSADMIN or OPRADMIN permission. In the case of OPRADMIN, **operation_mode** must be enabled.

- gs_roach_enable_delay_ddl_recycle(backupid name)

  Description: Enables DDL delay and returns the log position of the enabling point. It is similar to the pg_enable_delay_ddl_recycle system function but is more lightweight. In addition, different **backupid** values can be used to concurrently enable DDL statements with delay. **backupid** values that start with "gs_roach" are reserved for the internal backup tool, GaussRoach. To use them, you need to set **application_name** to '**gs_roach**'.

  Return type: text. The content is the log location of the start point.

- gs_roach_disable_delay_ddl_recycle(backupid text)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Disables DDL delay and returns the log range where DDL delay takes effect. It is similar to the pg_enable_delay_ddl_recycle system function but is more lightweight. In addition, different **backupid** values can be used to concurrently disable DDL statements with delay. **backupid** values that start with "gs_roach" are reserved for the internal backup tool, GaussRoach. To use them, you need to set **application_name** to **'gs_roach'**.

Return type: records. The content is the range of logs for which DDL delay takes effect.

- gs_roach_switch_xlog(request_ckpt bool)

  Description: Switches the currently used log segment file and triggers a full checkpoint if **request_ckpt** is set to **true**.

  Return type: text. The content is the location of the segment log.

  Note: To call this function, users must possess the SYSADMIN or OPRADMIN permission. In the case of OPRADMIN, **operation_mode** must be enabled.

- gs_block_dw_io(timeout int, identifier text)

  Description: Blocks doublewrite page flushing.

  Parameters:

  – timeout

    Block duration.

    Value range: [0,3600], in seconds. The value **0** indicates that the block duration is 0s.

  – identifier

    ID of the operation.

    Value range: a string, supporting only uppercase letters, lowercase letters, digits, and underscores (_).

  Return type: Boolean

  Note: To call this function, users must possess the SYSADMIN or OPRADMIN permission. In the case of OPRADMIN, **operation_mode** must be enabled.

- gs_is_dw_io_blocked()

  Description: Checks whether disk flushing on the current doublewrite page is blocked. If disk flushing is blocked, **true** is returned.

  Return type: Boolean

  Note: To call this function, users must possess the SYSADMIN or OPRADMIN permission. In the case of OPRADMIN, **operation_mode** must be enabled.

- gs_pitr_advance_last_updated_barrier()

  Description: In PITR mode, forcibly updates the global maximum archived recovery point uploaded to OBS/NAS last time to the current point. No input parameter is required.

  Return type: text

  Note: To call this function, users must possess the SYSADMIN or OPRADMIN permission. In the case of OPRADMIN, **operation_mode** must be enabled. This parameter is valid only on the first normal CN in a distributed system. The return value is the latest local maximum archived recovery point.

- gs_pitr_clean_local_barrier_files('delete_timestamp')

  Description: Clears locally cached barrier record files.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: The **delete_timestamp** parameter is of the text type. It is a Linux timestamp and contains 10 characters.

Return type: text

Note: To call this function, users must possess the SYSADMIN or OPRADMIN permission. In the case of OPRADMIN, **operation_mode** must be enabled. The returned result is the start timestamp of the earliest barrier file on the localhost after the deletion.

- gs_get_barrier_lsn(barrier_name text)

  Description: Obtains the LSN corresponding to the barrier created using a backup.

  Return type: text

  Note: Currently, only **gs_roach_full** and **gs_roach_inc** are supported. To call this function, the user must have the SYSADMIN or OPRADMIN permission. An O&M administrator must enable **operation_mode**.

- gs_gbr_relation_associated_filenode(schemaName name, tableName name)

  Description: Returns the relfilenode of all indexes, sequences, partitions, TOASTs, and TOAST indexes related to the input table.

  Return type: records

  Note: The table columns returned by gs_gbr_relation_associated_filenode include the file type relkind, namespace where the file is located, relation name corresponding to the file, OID of the database where the file is located, OID of the tablespace where the file is located, and relfilenode of the file.

- gs_pitr_stop_archive_manually(archive_slot_name cstring)

  Description: Stops an ongoing archiving task on a single CN/DN (valid only for remote media). The function returns whether the archiving task is stopped successfully.

  Return type: text

  Note: To call this function, the user must have the SYSADMIN or OPRADMIN permission. An O&M administrator must enable **operation_mode**. This is an internal function and cannot be called by users.

- pg_create_physical_replication_slot_extern(slotname text, dummy_standby bool, extra_content text, need_recycle_xlog bool)

  Description: Creates an OBS or NAS archive slot or backup slot. For an archive slot, **slotname** indicates the name of the archive slot or recovery slot. The primary and standby slots must use the same slot name. **dummy_standby** is a reserved parameter. **extra_content** contains some information about the archive slot. For an OBS archive slot, the format is **OBS;obs_server_ip;obs_bucket_name;obs_ak;obs_sk;archive_path;is_recovery;is_vote_replicate**, in which **OBS** indicates the archive media of the archive slot, **obs_server_ip** indicates the IP address of OBS, **obs_bucket_name** indicates the bucket name, **obs_ak** indicates the AK of OBS, **obs_sk** indicates the SK of OBS, **archive_path** indicates the archive path, and **is_recovery** specifies whether the slot is an archive slot or a recovery slot (**0**: archive slot; **1**: recovery slot). **is_vote_replicate** specifies whether the voting copy is archived first. The value **0** indicates that the synchronous standby node is archived first, and the value **1** indicates that the voting copy is archived first. This column is reserved in the current version and is not adapted yet. For a NAS archive slot, the format is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NAS;archive_path;is_recovery;is_vote_replicate. Compared with the OBS archive slot, the NAS archive slot does not have the OBS configuration information, while the meanings of other fields are the same.

If the media is not OBS or NAS, the OBS archive slot is used by default. The **extra_content** format is **obs_server_ip;obs_bucket_name;obs_ak;obs_sk;archive_path;is_recovery;is_vote_replicate**.

need_recycle_xlog specifies whether to recycle old archive logs when creating an archive slot. The value **true** indicates that old archive logs are recycled, and the value **false** indicates that old archive logs are not recycled.

For a backup slot, **slotname** starts with **gs_roach_** and is usually created by the backup process. **dummy_standby** is a reserved parameter. The value of **extra_content** can be **NULL** or a valid LSN character string. **need_recycle_xlog** does not affect the function of the backup slot. If the value of **extra_content** is **NULL**, the checkpoint redo point is used as the initial position of the backup slot. If the value of **extra_content** is a valid LSN character string and the LSN is smaller than the current checkpoint redo point, the LSN is used as the initial position of the backup slot. If the character string is an invalid LSN, the creation fails.

Return type: records, including slotname and xlog_position.

Note: Users who call this function must have the SYSADMIN permission or the REPLICATION permission, or inherit permissions of the built-in role gs_role_replication. Currently, multiple archive slots cannot be created.

Example:

Create an OBS archive slot.

In the following command, *obs_server_ip*, *obs_bucket_name*, *obs_ak*, and *obs_sk* indicate the OBS IP address, bucket name, AK, and SK, respectively. **gaussdb_uuid/dn1** indicates the log archive path in the OBS bucket. Replace it with the actual path as required.

```
gaussdb=# SELECT * FROM pg_create_physical_replication_slot_extern('uuid', false,
'OBS;obs_server_ip;obs_bucket_name;obs_ak;obs_sk;gaussdb_uuid/dn1;0;0', false);
 slotname | xlog_position
----------+---------------
 uuid     |
(1 row)
```

Create a NAS archive slot.

In the following command, **/data/nas/media/gaussdb_uuid/dn1** indicates the log archive path on the NAS. Replace it with the actual path as required.

```
gaussdb=# SELECT * FROM pg_create_physical_replication_slot_extern('uuid', false, 'NAS;/data/nas/
media/gaussdb_uuid/dn1;0;0', false);
 slotname | xlog_position
----------+---------------
 uuid     |
```

Create a backup slot.

Before using the pg_create_physical_replication_slot_extern function to create a backup slot, run the **set application_name='gs_roach'** command. This function is used for backup-related O&M operations. You are advised not to use it directly.

```
gaussdb=# SELECT * FROM pg_catalog.pg_create_physical_replication_slot_extern('gs_roach_full',
false, NULL, false);
   slotname    | xlog_position
---------------+---------------
 gs_roach_full | 0/3F883530
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_set_obs_delete_location(delete_location text)

  Description: Sets the location where OBS archive logs can be deleted. The value of **delete_location** is an LSN. The logs before this location have been replayed and flushed to disks and can be deleted on OBS.

  Return type: xlog_file_name text, indicating the file name of the logs that can be deleted. The value of this parameter is returned regardless of whether OBS is deleted successfully.

  Note: To call this function, users must possess the SYSADMIN or OPRADMIN permission. In the case of OPRADMIN, **operation_mode** must be enabled.

  ```
  gaussdb=# SELECT gs_set_obs_delete_location('0/54000000');
   gs_set_obs_delete_location
  ----------------------------
   000000010000000000000054_00
  (1 row)
  ```

- gs_set_obs_delete_location_with_slotname(cstring, cstring)

  Description: Sets the location where OBS archive logs in a specified archived slot can be deleted. The first parameter indicates the LSN. The logs before this location have been replayed and flushed to disks and can be deleted on OBS. The second parameter indicates the name of the archive slot.

  Return type: xlog_file_name text, indicating the file name of the logs that can be deleted. The value of this parameter is returned regardless of whether OBS is deleted successfully.

  Note: To call this function, users must possess the SYSADMIN or OPRADMIN permission. In the case of OPRADMIN, **operation_mode** must be enabled.

- gs_get_global_barrier_status()

  Description: gs_get_global_barrier_status is used to query the latest global barrier archived in OBS.

  Return type: text

  global_barrier_id: globally latest barrier ID.

  global_achive_barrier_id: globally latest archived barrier ID.

  Note: To call this function, users must possess the SYSADMIN or OPRADMIN permission. In the case of OPRADMIN, **operation_mode** must be enabled.

- gs_get_global_barriers_status()

  Description: gs_get_global_barriers_status is used to query the latest global barrier archived in OBS.

  Return type: text

  slot_name: slot name.

  global_barrier_id: globally latest barrier ID.

  global_achive_barrier_id: globally latest archived barrier ID.

  Note: To call this function, users must possess the SYSADMIN or OPRADMIN permission. In the case of OPRADMIN, **operation_mode** must be enabled.

- gs_set_archive_standby_name_on_primary(text)

  Description: gs_set_archive_standby_name_on_primary is used to specify the standby node for archiving. After the setting, the specified standby node performs the archive operation. In DCF mode, the standby node cannot be specified. The input is the name of a standby node, for example, **dn_6002**.

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The value **true** indicates that the setting is successful, while the value **false** indicates that the setting fails.

Restoration Control Functions

Restoration control functions provide information about the status of standby nodes. These functions may be executed both during restoration and in normal running.

- pg_is_in_recovery()

  Description: Returns **true** if restoration is still in progress.

  Return type: Boolean

- pg_last_xlog_receive_location()

  Description: Obtains the last transaction log location received and synchronized to disk by streaming replication. While streaming replication is in progress, this will increase monotonically. If restoration has been completed, then this value will remain static at the value of the last WAL record received and synchronized to disk during restoration. If streaming replication is disabled or if it has not yet started, the function returns a null value.

  Return type: text

- pg_last_xlog_replay_location()

  Description: Obtains last transaction log location replayed during restoration. If restoration is still in progress, this will increase monotonically. If restoration has been completed, then this value will remain static at the value of the last WAL record received during that restoration. When the server has been started normally without restoration, the function returns a null value.

  Return type: text

- pg_last_xact_replay_timestamp()

  Description: Obtains the timestamp of last transaction replayed during restoration. This is the time to commit a transaction or abort a WAL record on the primary node. If no transactions have been replayed during restoration, this function will return a null value. If restoration is still in progress, this will increase monotonically. If restoration has been completed, then this value will remain static at the value of the last WAL record received during that restoration. If the server normally starts without manual intervention, this function will return a null value.

  Return type: timestamp with time zone

Restoration control functions control restoration threads. These functions may be executed only during restoration.

- pg_is_xlog_replay_paused()

  Description: Returns **true** if restoration is paused.

  Return type: Boolean

- pg_xlog_replay_pause()

  Description: Pauses restoration immediately.

  Return type: void

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_xlog_replay_resume()

  Description: Restarts restoration if it was paused.

  Return type: void

- gs_get_active_archiving_standby()

  Description: Queries information about archive standby nodes in the same shard. The standby node name, archive location, and number of archive logs are returned.

  Return type: text, int

- gs_pitr_get_warning_for_xlog_force_recycle()

  Description: Checks whether logs are recycled because a large number of logs are stacked in the archive slot after archiving is enabled.

  Return type: Boolean

- gs_pitr_clean_history_global_barriers(stop_barrier_timestamp cstring)

  Description: Clears all barrier records generated before the specified time. The earliest barrier record is returned. The input parameter is of the cstring type and is a Linux timestamp. You need to perform this operation as an administrator or O&M administrator.

  Return type: text

- gs_pitr_archive_slot_force_advance(stop_barrier_timestamp cstring)

  Description: Forcibly updates the archive slot and clears unnecessary barrier records. The new archive slot location is returned. The input parameter is of the cstring type and is a Linux timestamp. You need to perform this operation as an administrator or O&M administrator.

  Return type: text

- gs_recent_barrier_buffer_info(start_time text, end_time text)

  Description: Queries barrier information based on the time range entered by the user to obtain time_stamp, CSN, LSN, and standard_time.

  Return type: records

  Note: To call this function, you must have the SYSADMIN or OPRADMIN permission. The input parameters **start_time** and **end_time** are in the format of year-month-day time, where the time is in the clock format. The maximum query time span is one day. If the time span exceeds the limit, the end time is automatically converted to the limit boundary based on the query start time. Example:

```
gaussdb=# SELECT * FROM gs_recent_barrier_buffer_info('2024-01-15 23:27:50', '2024-01-15
23:28:00');
 timestamp |       lsn        |   csn   |    standard_time
-----------+------------------+---------+---------------------
 1705332470 | 00000000/15FFBBA0 | 41020421 | 2024-01-15 23:27:50
 1705332471 | 00000000/15FFBDF0 | 41020422 | 2024-01-15 23:27:51
 1705332472 | 00000000/15FFC058 | 41020423 | 2024-01-15 23:27:52
 1705332472 | 00000000/15FFC0F8 | 41020424 | 2024-01-15 23:27:52
 1705332473 | 00000000/15FFC348 | 41020425 | 2024-01-15 23:27:53
 1705332474 | 00000000/15FFC598 | 41020426 | 2024-01-15 23:27:54
 1705332475 | 00000000/15FFC638 | 41020427 | 2024-01-15 23:27:55
 1705332476 | 00000000/15FFC888 | 41020428 | 2024-01-15 23:27:56
 1705332476 | 00000000/15FFDC80 | 41020433 | 2024-01-15 23:27:56
 1705332477 | 00000000/15FFDD20 | 41020434 | 2024-01-15 23:27:57
 1705332478 | 00000000/15FFDF70 | 41020435 | 2024-01-15 23:27:58
 1705332479 | 00000000/15FFE1D8 | 41020436 | 2024-01-15 23:27:59
 1705332480 | 00000000/15FFE278 | 41020437 | 2024-01-15 23:28:00
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 1705332480 | 00000000/15FFE4C8 | 41020438 | 2024-01-15 23:28:00
(14 rows)
```

- gs_show_obs_media_files(slot_name cstring, src cstring, offset int32, limit int32)

  Description: Queries the OBS file list based on the archive slot (**slot_name**) and OBS directory address (**src**) entered by the user.

  Return type: records

  Note: To call this function, you must have the SYSADMIN or OPRADMIN permission. **offset** indicates the query result offset, and **limit** indicates the maximum number of output lines. All files in **src** are queried. Example:

```
gaussdb=# SELECT gs_show_obs_media_files('ssh','cn_5001/pg_xlog',1, 5);
                                gs_show_obs_media_files
-----------------------------------------------------------------------------------------------
 (wstdist_ssh/archive/cn_5001/pg_xlog/
000000010000000000000010_01_01_00000000_00000000_00000003)
 (wstdist_ssh/archive/cn_5001/pg_xlog/
000000010000000000000010_02_01_00000000_00000000_00000003)
 (wstdist_ssh/archive/cn_5001/pg_xlog/
000000010000000000000010_03_01_00000000_00000000_00000003)
 (wstdist_ssh/archive/cn_5001/pg_xlog/
000000010000000000000011_00_01_00000000_00000000_00000003)
 (wstdist_ssh/archive/cn_5001/pg_xlog/
000000010000000000000011_01_01_00000000_00000000_00000003)
(5 rows)
```

- gs_upload_obs_media_file(slot_name cstring, src cstring, dest cstring, is_forced bool)

  Description: Uploads OBS files based on the archive slot (**slot_name**), source address (**src**), OBS address (**dest**), and whether to forcibly upload files (**is_forced**).

  Return type: void

  Note: To call this function, you must have the SYSADMIN or OPRADMIN permission. The original file directory must be the directory specified by *$GAUSSLOG*. Example:

```
gaussdb=# SELECT * FROM gs_upload_obs_media_file('ssh', '/data/gauss/log/stwang/test/
000000010000000000000019_02_01_00000000_00000000_00000003', 'cn_5001/pg_xlog/
000000010000000000000019_02_01_00000000_00000000_00000003', true);
 gs_upload_obs_media_file
--------------------------

(1 row)
```

- gs_download_obs_media_file(slot_name cstring, src cstring, dest cstring)

  Description: Downloads OBS files based on the archive slot (**slot_name**), download source address (**src**), and local destination address (**dest**).

  Return type: void

  Note: To call this function, you must have the SYSADMIN or OPRADMIN permission. The download directory must be the directory specified by *$GAUSSLOG*. Example:

```
gaussdb=# SELECT * FROM gs_download_obs_media_file('ssh','cn_5001/pg_xlog/
000000010000000000000019_02_01_00000000_00000000_00000003','/data/gauss/log/stwang/test');
 gs_download_obs_media_file
----------------------------

(1 row)
```

While restoration is paused, no further database changes are applied. In hot standby mode, all new queries will see the same consistent snapshot of the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

database, and no further query conflicts will be generated until restoration is resumed.

If streaming replication is disabled, the paused state may continue indefinitely without problem. While streaming replication is in progress, WAL records will continue to be received, which will eventually fill available disk space. This progress depends on the duration of the pause, the rate of WAL generation, and available disk space.

1.6.27.5 Dual-Cluster DR Control Functions

- gs_streaming_dr_in_switchover()

  Description: Truncates services during a planned switchover in the primary cluster in streaming replication-based remote DR solutions.

  Return type: Boolean, indicating whether the service truncation is successful and whether the switchover process can be performed normally.

- gs_set_dr_dest_name()

  Description: Configures the IDs of all log receiver clusters in the log sender cluster. The input parameter is a character string consisting of the IDs of all log receiver clusters directly connected to the log sender cluster. The IDs are separated by commas (,). If the setting is successful, **t** is returned. Otherwise, **f** is returned.

  Input parameter type: text. If the input parameter is **0**, the instance ID of the log receiver cluster is not configured. This parameter is automatically set during the DR process. You are advised not to change the value.

  NOTICE

  The instance ID of the log receiver cluster cannot be **invalid_cluster_uuid** and can contain a maximum of 36 characters.

  Return type: Boolean

  Example

  ```
  gaussdb=# SELECT * FROM gs_set_dr_dest_name('cluster1,cluster2');
   gs_set_dr_dest_name
  ---------------------
   t
  (1 row)
  ```

- gs_get_dr_dest_name()

  Description: Queries all instance IDs of the log receiver cluster configured using **gs_set_dr_dest_name**.

  Return type: text. If the return value is **0**, the configured instance ID of the log receiver cluster is empty.

  Example

  ```
  gaussdb=# SELECT * FROM gs_get_dr_dest_name();
   gs_get_dr_dest_name
  ---------------------
   cluster1,cluster2
  (1 row)
  ```

- gs_set_hadr_link_info()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Sets the DR link information. The input parameter is a text in JSON format. If the setting is successful, **"success"** is returned. If the setting fails, **"failed"** is returned. This parameter is automatically set during the DR process. You are advised not to change the value.

Return type: text.

Table 1-127 gs_set_hadr_link_info parameters

Parameter	Type	Description
meta_version	string	Version number of the DR link in JSON format.
version	string	Version number of the DR link information.
replication_info	string	Stores DR link information, which is a JSON array containing information about multiple links.
dest_name	string	ID of the standby cluster instance for DR.
dest_type	string	Types of DR links. Value type: enumerated value Value range: **physical_standby_cluster**, **logical_standby_cluster**, and **gaussstor** Default value: **physical_standby_cluster**
mode	string	Log synchronization mode of DR links. Value type: enumerated value Value range: • **sync**: The link waits for the standby node at the log receiving end to persist redo logs to disks to ensure that no data is lost. This affects the host performance. • **semisync**: The link is preferentially synchronized. If the link is disconnected or the RPO is abnormal, the link is degraded to asynchronous. • **async**: The asynchronous transmission is performed, preferentially ensuring host performance. Default value: **async**
net_timeout	string	Timeout threshold in semi-synchronous mode. Value range: [0,*max*] Unit: second Default value: **0**
rpo_threshold	int	RPO threshold in semi-synchronous mode. Value range: [0,*max*] Unit: second Default value: **0**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
compression	string	Specifies whether to enable compression. Value type: enumerated value Value range: **enable** and **disable** Default value: **enable**
compression_algo_setting	string	Compression algorithm. Currently, this parameter cannot be modified and the default value **"lz4,1"** is used.
replication_rate_limit	int/ string	Speed limit of the DR link. The unit can be **K**, **M**, or **G** for example, **1024K**, **1M**, **10M**, and **2048G**. Unit: byte Value range: **-1**, **0**, and 1 MB to 2 TB ● Integer: **-1**, **0**, and 1048576 to 2199023255552 ● String: **-1**, **0**, and 1048576 to 2199023255552 Default value: **-1**
rate_burst	int	Maximum size of a data packet that can be sent at a time. The value is a multiple of **replication_rate_limit**. Value range: [1,5] Default value: **2**
conn_info	string	Connection string and access permission information. For details, see the following example.

Example

```
gaussdb=# SELECT gs_set_hadr_link_info('{"meta_version":"1","version":"3","replication_info":
[{"dest_name":"clusterb","dest_type":"physical_standby_cluster","mode":"async","net_timeout":0,"rpo_t
hreshold":0,"compression":"enable","compression_algo_setting":"lz4,1","replication_rate_limit":-1,"rate_
burst":2,"conn_info":{"shards":{"replconninfo":{"shard_1":{"localinfo":{"dn_6001":
{"h":"127.0.0.1","p":40001,"hbp":40005},"dn_6002":{"h":"127.0.0.1","p":40001,"hbp":40005},"dn_6003":
{"h":"127.0.0.1","p":40001,"hbp":40005}},"remoteinfo":[{"h":"127.0.0.1","p":40001,"hbp":40005},
{"h":"127.0.0.1","p":40001,"hbp":40005},{"h":"127.0.0.1","p":40001,"hbp":40005}]},"shard_2":
{"localinfo":{"dn_6006":{"h":"127.0.0.1","p":40021,"hbp":40025},"dn_6004":
{"h":"127.0.0.1","p":40021,"hbp":40025},"dn_6005":
{"h":"127.0.0.1","p":40021,"hbp":40025}},"remoteinfo":[{"h":"127.0.0.1","p":40021,"hbp":40025},
{"h":"127.0.0.1","p":40021,"hbp":40025},{"h":"127.0.0.1","p":40021,"hbp":40025}]},"shard_3":
{"localinfo":{"dn_6008":{"h":"127.0.0.1","p":40041,"hbp":40045},"dn_6009":
{"h":"127.0.0.1","p":40041,"hbp":40045},"dn_6007":
{"h":"127.0.0.1","p":40041,"hbp":40045}},"remoteinfo":[{"h":"127.0.0.1","p":40041,"hbp":40045},
{"h":"127.0.0.1","p":40041,"hbp":40045},{"h":"127.0.0.1","p":40041,"hbp":40045}]}},"hba":
[{"type":"host","database":"all","user":"rdsAdmin","method":"trust"},
{"type":"host","database":"replication","user":"stream_user","method":"sha256"}]},"cn":{"replconninfo":
{"cn_5001":{"h":"127.0.0.1","p":9081,"hbp":9085,"remoteinfo":
[{"h":"127.0.0.1","p":9081,"hbp":9085}]},"cn_5002":{"h":"127.0.0.1","p":9081,"hbp":9085,"remoteinfo":
[{"h":"127.0.0.1","p":9081,"hbp":9085}]},"cn_5003":{"h":"127.0.0.1","p":9081,"hbp":9085,"remoteinfo":
[{"h":"127.0.0.1","p":9081,"hbp":9085}]}},"hba":
[{"type":"host","database":"all","user":"rdsAdmin","method":"trust"},
{"type":"host","database":"replication","user":"stream_user","method":"sha256"}]}}}]} ');
 gs_set_hadr_link_info
-----------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 success
(1 row)
```

- gs_get_hadr_link_info()

  Description: Displays information about all DR links.

  Return type: record

  Example

```
gaussdb=# SELECT * FROM gs_get_hadr_link_info();
-[ RECORD 1 ]------------
+-------------------------------------------------------------------------------------------------------------------------------------------------
--------
dest_name               | clusterb
dest_type               | physical_standby_cluster
mode                    | semisync
net_timeout             | 60
rpo_threshold           | 10
compression             | enable
compression_algo_setting | lz4,1
replication_rate_limit   | 0
rate_burst              | 2
replconninfo            | localhost=127.0.0.1 localport=20001 localheartbeatport=20005
remotehost=127.0.0.1 remoteport=20025 remoteheartbeatport=20029
                        | localhost=127.0.0.1 localport=20001 localheartbeatport=20005
remotehost=127.0.0.1 remoteport=20033 remoteheartbeatport=20037
                        | localhost=127.0.0.1 localport=20001 localheartbeatport=20005
remotehost=127.0.0.1 remoteport=20041 remoteheartbeatport=20045
                        |
hba                     | local all all trust
                        | local all all trust
                        | local all all trust
                        |
```

- gs_get_hadr_link_version()

  Description: Obtains the version number of the DR link information.

  Return type: int8

  Example

```
gaussdb=# SELECT * FROM gs_get_hadr_link_version();
 link_info_version
-------------------
               100
(1 row)
```

- gs_get_hadr_sync_status()

  Description: Displays the log synchronization status of all DR links.

  Return type: record

Table 1-128 gs_get_hadr_sync_status parameters

Category	Parameter	Type	Description
Output parameter	dest_name	text	ID of the cluster that receives logs over the DR link.
Output parameter	sync_state	text	Actual synchronization status of the current DR link.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	current_rpo	int	RPO time of the current DR link, in seconds.
Output parameter	sender_sent_location	text	Location where the DR link log sender sends logs.
Output parameter	sender_write_location	text	Location where the DR link log sender writes logs to the file system.
Output parameter	sender_flush_location	text	Location of the flush disk of the DR link log sender.
Output parameter	sender_replay_location	text	Location of the replay log of the DR link log sender.
Output parameter	receiver_received_location	text	Location of the received log of the DR link log receiver.
Output parameter	receiver_flush_location	text	Location of the log flush disk of the DR link log receiver.
Output parameter	last_disconnect_time	timestamptz	Latest DR link disconnection time.
Output parameter	last_upgrade_to_sync_time	timestamptz	Time when the semi-synchronous mode is upgraded to the synchronous mode for the last time.
Output parameter	last_downgrade_to_async_time	timestamptz	Time when the semi-synchronous mode is downgraded to the asynchronous mode for the last time.
Output parameter	mode	text	Value of the log synchronization mode of the DR link.
Output parameter	net_timeout	text	Timeout threshold of the semi-synchronous mode of the DR link.
Output parameter	rpo_threshold	text	RPO threshold of the semi-synchronous mode of the DR link.

Example

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT * FROM gs_get_hadr_sync_status();
-[ RECORD 1 ]----------------+------------------------------
dest_name                    | clusterb
sync_state                   | Sync
current_rpo                  | 0
sender_sent_location         | 0/11380B20
sender_write_location        | 0/11380B20
sender_flush_location        | 0/11380B20
sender_replay_location       | 0/11380B20
receiver_received_location   | 0/11380B20
receiver_flush_location      | 0/11380B20
last_disconnect_time         | 2026-01-15 20:35:42.18516+08
last_upgrade_to_sync_time    | 2026-01-15 21:29:55.147578+08
last_downgrade_to_async_time | 2026-01-15 20:36:42.187545+08
mode                         | semisync
net_timeout                  | 60
rpo_threshold                | 10
```

- gs_stat_get_replication_limit()

  Description: Queries the parameter values of flow control for all links of the current cluster and the average bandwidth usage in the last 1s, 5s, and 60s.

  Return type: record

  Table 1-129 gs_stat_get_replication_limit parameters

Category	Parameter	Type	Description
Output parameter	dest_name	text	Name of the target cluster of the sending link.
Output parameter	replication_rate_limit	text	Maximum bandwidth configured for the link.
Output parameter	rate_burst	text	Maximum burst traffic configured for the link, which is a multiple of the maximum bandwidth.
Output parameter	current_usage	text	Average bandwidth used by the link in the last 1s, in byte/s.
Output parameter	average_usage_5s	text	Average bandwidth used by the link in the last 5s, in byte/s.
Output parameter	average_usage_60s	text	Average bandwidth used by the link in the last 60s, in byte/s.

  Example

```
gaussdb=# SELECT * FROM gs_stat_get_replication_limit();
-[ RECORD 1 ]----------+---------
dest_name              | clusterb
replication_rate_limit | -1
rate_burst             | 2
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
current_usage          | 0
average_usage_5s       | 0
average_usage_60s      | 0
-[ RECORD 2 ]----------+---------
dest_name              | clusterc
replication_rate_limit | -1
rate_burst             | 2
current_usage          | 0
average_usage_5s       | 0
average_usage_60s      | 0
```

1.6.27.6 Dual-Cluster DR Query Functions

- gs_get_local_barrier_status()

  Description: If the streaming replication-based remote DR feature is enabled, the primary cluster and standby cluster for DR synchronize logs. The barrier log is flushed to disk in the primary cluster, and replayed in the standby cluster for DR to determine the archive log progress of the primary cluster and the log replay progress of the standby cluster for DR. **gs_get_local_barrier_status** is used to query the current log replay status of each node in the standby cluster for DR.

  Return type: text.

  barrier_id: latest barrier ID of a node in the standby cluster for DR.

  barrier_lsn: LSN of the latest barrier ID returned by a node in the standby cluster for DR.

  archive_lsn: location of archive logs obtained by a node in the standby cluster for DR. This parameter does not take effect currently.

  flush_lsn: location of logs that have been flushed to disks on a node in the standby cluster for DR.

- gs_hadr_in_recovery()

  Description: Checks whether the current node is in barrier-based log restoration if the streaming replication-based remote DR feature is enabled. If it is in restoration, **true** is returned. Only after the log restoration is complete, can the standby cluster for DR be promoted to the production cluster during the switchover process. This operation must be performed by system administrators.

  Return type: Boolean

  > **NOTE**
  >
  > This function is used only when a planned switchover is performed in the standby cluster for DR.

- gs_streaming_dr_get_switchover_barrier()

  Description: Checks whether the CN and main standby DN in the standby cluster for DR have received the switchover barrier logs and replayed the logs in the streaming replication-based remote DR solution. If it has, **true** is returned. In the standby cluster for DR, the procedure for promoting the standby database instance for DR to the production database instance in the switchover process can be started only after the switchover barrier logs of all DNs are replayed (the SYSADMIN permission is required).

  Return type: Boolean

  Note: This function is used only when a planned switchover is performed in the standby database instance in streaming DR solutions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_streaming_dr_service_truncation_check()

  Description: Checks whether the CN and primary DN in the primary cluster has sent the switchover barrier logs in the streaming replication-based remote DR solution. If it has, **true** is returned. The procedure for demoting the production database instance to the standby database instance in the switchover process can be started only after the logs are sent (restricted to system administrators).

  Return type: Boolean

  Note: This function is used only when a planned switchover is performed in the standby database instance for DR.

- gs_hadr_local_rto_and_rpo_stat()

  Description: Displays the log flow control information of the local database instance and standby database instance for streaming DR. (If this command is executed on a node that does not participate in streaming DR, for example, a standby DN or a CN, no information may be returned.)

  The return value type is record. The types and meanings of the columns are as described in **Table 1-130**.

Table 1-130 gs_hadr_local_rto_and_rpo_stat parameters

Parameter	Type	Description
hadr_sender_node_name	text	Node name, including the primary database instance and the main standby node of the standby database instance.
hadr_receiver_node_name	text	Name of the main standby node of the standby database instance.
source_ip	text	IP address of the primary DN of the primary database instance.
source_port	int	Communication port of the primary DN of the primary database instance.
dest_ip	text	IP address of the main standby DN of the standby database instance.
dest_port	int	Communication port of the main standby DN of the standby database instance.
current_rto	int	Flow control information, that is, log RTO time of the current primary and standby database instances (unit: second).
target_rto	int	Flow control information, that is, the RTO time between the target primary and standby database instances (unit: second).
current_rpo	int	Flow control information, that is, log RPO time of the current primary and standby database instances (unit: second).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
target_rpo	int	Flow control information, that is, the RPO time between the target primary and standby database instances (unit: second).
rto_sleep_time	int	RTO flow control information, that is, expected sleep time (unit: μs) required by WAL sender on the host to reach the specified RTO.
rpo_sleep_time	int	RPO flow control information, that is, the expected sleep time (unit: μs) required by xlogInsert on the host to reach the specified RPO.

- gs_hadr_remote_rto_and_rpo_stat()

  Description: Displays the log flow control information of all other shards or CN database instances and DR database instances for streaming DR. (Generally, this command is executed on CNs. If this command is executed on DNs, no information may be returned.)

  The return value type is record. The types and meanings of the columns are as described in **Table 1-131**.

  Table 1-131 gs_hadr_remote_rto_and_rpo_stat

Parameter	Type	Description
hadr_sender_node_name	text	Node name, including the primary database instance and the main standby node of the standby database instance.
hadr_receiver_node_name	text	Name of the main standby node of the standby database instance.
source_ip	text	IP address of the primary DN of the primary database instance.
source_port	int	Communication port of the primary DN of the primary database instance.
dest_ip	text	IP address of the main standby DN of the standby database instance.
dest_port	int	Communication port of the main standby DN of the standby database instance.
current_rto	int	Flow control information, that is, log RTO time of the current primary and standby database instances (unit: second).
target_rto	int	Flow control information, that is, the RTO time between the target primary and standby database instances (unit: second).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
current_rpo	int	Flow control information, that is, log RPO time of the current primary and standby database instances (unit: second).
target_rpo	int	Flow control information, that is, the RPO time between the target primary and standby database instances (unit: second).
rto_sleep_time	int	RTO flow control information, that is, expected sleep time (unit: μs) required by WAL sender on the host to reach the specified RTO.
rpo_sleep_time	int	RPO flow control information, that is, the expected sleep time (unit: μs) required by xlogInsert on the host to reach the specified RPO.

- gs_is_wal_truncate_complete()

  Description: Queries whether online log truncation is complete when the standby cluster for DR is promoted to primary. If online log truncation is complete, **true** is returned. The value is reset each time the query is complete.

  Return type: Boolean

  Remarks: This function is used only when the standby cluster for DR is promoted to primary.

- gs_hadr_local_replay_delay_stat()

  Description: Displays the barrier receiving, barrier replay, and delay time of the standby cluster for DR when the standby cluster for DR delays replay.

  Return type: record.

  Table 1-132 gs_hadr_local_replay_delay_stat return parameters

Parameter	Type	Description
node_name	text	Specifies the instance name.
receive_barrier	text	Receives logs that carry timestamps.
receive_barrier_lsn	text	Receives the LSN of the log with the timestamp.
receive_time	timestamp with time zone	Parses the received timestamp.
replay_barrier	text	Replays logs that carry timestamps.
replay_barrier_lsn	text	Replays the LSN of the log that carries the timestamp.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
replay_time	timestamp with time zone	Parses the replay timestamp.
delay_replay	int8	Specifies the replay delay.
current_remain_interval	int8	Indicates the actual delay interval.

Example:

```
gaussdb=# SELECT * FROM gs_hadr_local_replay_delay_stat();
  node_name   |             receive_barrier             | receive_barrier_lsn |        receive_time        |
replay_barrier              | replay_barrier_lsn |        replay_time         | delay_replay |
current_remain_interval
--------------+-----------------------------------------+---------------------+----------------------------
+-----------------------------------------+--------------------+----------------------------+--------------
+-------------------------
 dn_6001_6002 | csn_00000000000000023117_1772440335494 | 00000000/1225B4B0   | 2026-03-02
16:32:15.494+08 | csn_00000000000000023111_1772440331492 | 00000000/1225B320  | 2026-03-02
16:32:11.492+08 |            0 |                         4
(1 row)
```

- disaster_cluster_run_mode()

  Description: Queries whether the primary cluster or standby cluster for DR is connected during connection. When this function is queried, **0** is returned for the primary cluster and **1** is returned for the standby cluster for DR.

  Return type: int.

  Example:

  ```
  gaussdb=# SELECT pg_catalog.disaster_cluster_run_mode();
  ----------------
               0
  (1 row)
  ```

- pgxc_disaster_read_node()

  Description: Returns the cluster information related to the standby cluster for DR during connection.

  The return value type is record. The types and meanings of the columns are as described in **Table 1-133**.

  Note: An error is reported when the primary cluster is queried. This function is used only for the standby cluster for DR.

Table 1-133 pgxc_disaster_read_node

Parameter	Type	Description
node_name	name	Node name.
node_type	char	Node type. **D** represents the main standby DN, **S** represents the cascaded standby DN, and **C** represents the query node.
node_host	name	Communication IP address of a node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
node_port	int	Communication port of a node.
node_host1	name	Virtual communication IP address of a node.
node_port1	int	Virtual communication port of a node.
nodeis_active	boolean	Node status. **true** indicates normal, and **false** indicates abnormal.

Example:

```
gaussdb=# SELECT * from pg_catalog.pgxc_disaster_read_node();
  node_name   | node_type | node_host  | node_port | node_host1 | node_port1 | nodeis_active
--------------+-----------+------------+-----------+------------+------------+---------------
 dn_6001_6002 | D         | 8.92.0.126 |      4000 | 8.92.0.126 |       4000 | t
 dn_6003_6004 | D         | 8.92.0.128 |      4020 | 8.92.0.128 |       4020 | t
 dn_6005_6006 | D         | 8.92.0.127 |      4040 | 8.92.0.127 |       4040 | t
 dn_6007_6008 | D         | 8.92.0.68  |      4060 | 8.92.0.68  |       4060 | t
 dn_6001_6002 | S         | 8.92.0.127 |      4000 | 8.92.0.127 |       4000 | t
 dn_6003_6004 | S         | 8.92.0.68  |      4020 | 8.92.0.68  |       4020 | t
 dn_6005_6006 | S         | 8.92.0.126 |      4040 | 8.92.0.126 |       4040 | t
 dn_6007_6008 | S         | 8.92.0.128 |      4060 | 8.92.0.128 |       4060 | t
 cn_5001      | C         | 8.92.0.128 |      9080 | 10.0.0.128 |       9080 | t
 cn_5002      | C         | 8.92.0.127 |      9080 | 10.0.0.127 |       9080 | t
(10 rows)
```

1.6.27.7 Snapshot Synchronization Functions

Snapshot synchronization functions save the current snapshot and return its identifier.

- pg_export_snapshot()

  Description: Saves the snapshot system information of the current transaction and returns the snapshot ID to the client.

  Return type: text

  Note: **pg_export_snapshot** saves the current snapshot and returns a text string identifying the snapshot. This string must be passed to clients that want to import the snapshot. A snapshot can be imported when the **set transaction snapshot snapshot_id;** command is executed. Doing so is possible only when the transaction is set to the **SERIALIZABLE** or **REPEATABLE READ** isolation level. Currently, GaussDB does not support the two isolation levels. The output of the function cannot be used as the input of **set transaction snapshot**.

- pg_export_snapshot_and_csn()

  Description: Saves the snapshot system information of the current transaction and returns the snapshot ID and the corresponding CSN to the client. Compared with **pg_export_snapshot()**, **pg_export_snapshot()** returns a CSN, indicating the CSN of the current snapshot.

  Return type: text.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.6.27.8 Database Object Functions

Database Object Size Functions

Database object size functions calculate the actual disk space used by database objects.

- pg_column_size(any)

  Description: Specifies the number of bytes used to store a particular value (possibly compressed).

  Return type: int.

  Note: **pg_column_size** displays the space for storing an independent data value.

  ```
  gaussdb=# SELECT pg_column_size(1);
   pg_column_size
  ----------------
               4
  (1 row)
  ```

- pg_database_size(oid)

  Description: Specifies the disk space used by the database with a specified OID.

  Return type: bigint

- pg_database_size(name)

  Description: Specifies the disk space used by the database with a specified name.

  Return type: bigint

  Note: **pg_database_size** receives the OID or name of a database and returns the disk space used by the corresponding object.

  Example:

  ```
  gaussdb=# CREATE DATABASE testdb;
  gaussdb=# SELECT pg_database_size('testdb');
   pg_database_size
  ------------------
          51590112
  (1 row)
  gaussdb=# DROP DATABASE testdb;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

The query time of the pg_database_size(oid/name) function is linearly related to the number of object files in the database corresponding to the OID/name. To evaluate the time required, perform the following steps:

1. Since the time required is related to the software and hardware environments, the test result of a certain database (the number of object files in the selected database should not be too large) is used as the benchmark. Calculate the time consumed by a single object file (**database_size_t**) based on the query time and the number of object files in the database.
2. Obtain the OID of the database to be queried using the command **SELECT oid, * FROM pg_database WHERE datname = 'database';**. In the **base** and **pg_tblspc** tablespace directories, count the total number of files (**database_obj_num**) in the directory corresponding to the OID. If the directory contains subdirectories, you need to enter it to view the number of object files. You can run **ls | wc -l** to query the number of files in the directory.
3. Calculate the estimated total time consumption based on **database_size_t** (time consumption of a single file) and **database_obj_num** (number of all object files in the database) in the benchmark test result. The total time consumption is equal to **database_size_t** multiplied by **database_obj_num**. If the estimated latency is too long, consider adjusting the execution timeout threshold to avoid unnecessary alarms.

- get_db_source_datasize()

  Description: Estimates the total size of non-compressed data in the current database.

  Return type: bigint

  Note: ANALYZE must be performed before this function is called.

  Example:

```
gaussdb=# analyze;
ANALYZE
gaussdb=# SELECT get_db_source_datasize();
 get_db_source_datasize
------------------------
            35384925667
(1 row)
```

- pg_relation_size(relation regclass, fork text)

  Description: Specifies the disk space used by the specified bifurcating tree ('main', 'fsm', or 'vm') of a certain table or index.

  Return type: bigint

- pg_relation_size(relation regclass)

  Description: Is an abbreviation of **pg_relation_size(..., 'main')**.

  Return type: bigint

  Note: **pg_relation_size** receives the OID or name of a table, an index, or a compressed table, and returns the size.

- pg_partition_size(oid, oid)

  Description: Specifies the disk space used by the partition with a specified OID. The first **oid** is the OID of the table and the second **oid** is the OID of the partition.

  Return type: bigint

- pg_partition_size(text, text)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Specifies the disk space used by the partition with a specified name. The first **text** is the table name and the second **text** is the partition name.

Return type: bigint

- pg_partition_indexes_size(oid, oid)

  Description: Specifies the disk space used by the index of the partition with a specified OID. The first **oid** is the OID of the table and the second **oid** is the OID of the partition.

  Return type: bigint

- pg_partition_indexes_size(text, text)

  Description: Specifies the disk space used by the index of the partition with a specified name. The first **text** is the table name and the second **text** is the partition name.

  Return type: bigint

- pg_indexes_size(regclass)

  Description: Specifies the total disk space used by the index appended to the specified table.

  Return type: bigint

- pg_size_pretty(bigint)

  Description: Converts a size in bytes expressed as a 64-bit integer into a human-readable format with size units.

  Return type: text.

- pg_size_pretty(numeric)

  Description: Converts a size in bytes expressed as a numeric value into a human-readable format with size units.

  Return type: text.

  Note: **pg_size_pretty** formats the results of other functions into a human-readable format. KB, MB, GB, and TB can be used.

- pg_table_size(regclass)

  Description: Specifies the disk space used by the specified table, excluding indexes (but including TOAST, free space mapping, and visibility mapping).

  Return type: bigint

- pg_tablespace_size(oid)

  Description: Specifies the disk space used by the tablespace with a specified OID.

  Return type: bigint

- pg_tablespace_size(name)

  Description: Specifies the disk space used by the tablespace with a specified name.

  Return type: bigint

  Note:

  pg_tablespace_size receives the OID or name of a database and returns the disk space used by the corresponding object.

- pg_total_relation_size(regclass)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Specifies the total disk space used by the specified table, including all indexes and TOAST data.

Return type: bigint

- dist_relation_size(nspname name, relname name)

  Description: Specifies the number of relpages of ordinary tables in a specified schema on each DN.

  Return type: float8.

  Table 1-134 Parameters

Input Parameter	Type	Description
nspname	name	Schema name of a table.
relname	name	Table name.

  In the 1 CN+6 DN database environment, you can view the number of relpages on the 6 DNs.

```
gaussdb=# CREATE TABLE tab(col int);
-- The value of node_name must be the same as the node name in the current environment.
gaussdb=# SELECT * FROM dist_relation_size('public','tab');
 node_name | relpages
-----------+----------
 datanode1 |        0
 datanode2 |        0
 datanode3 |        0
 datanode4 |        0
 datanode6 |        0
 datanode5 |        0
(6 rows)

gaussdb=# DROP TABLE tab;
```

  NOTE

  - This function cannot be used during the upgrade observation period.
  - This function cannot be called on DNs.

- dist_partition_size(nspname name, relname name, subname name)

  Description: Specifies the number of relpages of a specified partition in a partitioned table in a specified schema on each DN.

  Return type: float8.

  Table 1-135 Parameters

Input Parameter	Type	Description
nspname	name	Schema name of a table.
relname	name	Table name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Input Parameter	Type	Description
subname	name	Name of the partition to be queried.

In the 1 CN+6 DN database environment, you can view the number of relpages on the 6 DNs.

```
gaussdb=# CREATE TABLE test_part_table(col int)
PARTITION BY RANGE(col)
(
    PARTITION p1 VALUES LESS THAN (10),
    PARTITION p2 VALUES LESS THAN (20),
    PARTITION p3 VALUES LESS THAN (MAXVALUE)
);
-- The value of node_name must be the same as the node name in the current environment.
gaussdb=# SELECT * FROM dist_partition_size('public','test_part_table','p1') ORDER BY node_name;
 node_name | relpages
-----------+----------
 datanode1 |        0
 datanode2 |        0
 datanode3 |        0
 datanode4 |        0
 datanode5 |        0
 datanode6 |        0
(6 rows)

gaussdb=# DROP TABLE test_part_table;
```

NOTE

- This function cannot be used during the upgrade observation period.
- This function cannot be called on DNs.

- datalength(any)

  Description: Specifies the number of bytes used by an expression of a specified data type (data management space, data compression, or data type conversion is not considered).

  Return type: int.

  Note: **datalength** is used to calculate the space of an independent data value.

  Example:

  ```
  gaussdb=# SELECT datalength(1);
  datalength
  ------------
  4
  (1 row)
  ```

  The following table lists the supported data types and calculation methods.

Data Type			Storage Space
Numeric data types	Integer types	TINYINT	1
		SMALLINT	2
		INTEGER	4
		BINARY_INTEGER	4

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Data Type			Storage Space
		BIGINT	8
	Arbitrary precision types	DECIMAL	Every four decimal digits occupy two bytes. The digits before and after the decimal point are calculated separately.
		NUMERIC	Every four decimal digits occupy two bytes. The digits before and after the decimal point are calculated separately.
		NUMBER	Every four decimal digits occupy two bytes. The digits before and after the decimal point are calculated separately.
	Sequence integer	SMALLSERIAL	2
		SERIAL	4
		BIGSERIAL	8
	Floating-point types	FLOAT4	4
		DOUBLE PRECISION	8
		FLOAT8	8
		BINARY_DOUBLE	8
		FLOAT[(p)]	Every four decimal digits occupy two bytes. The digits before and after the decimal point are calculated separately.
		DEC[(p[,s])]	Every four decimal digits occupy two bytes. The digits before and after the decimal point are calculated separately.
		INTEGER[(p[,s])]	Every four decimal digits occupy two bytes. The digits before and after the decimal point are calculated separately.
Boolean data types	Boolean type	BOOLEAN	1

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Data Type			Storage Space
Character data types	Character types	CHAR	1
		CHAR(n)	n
		CHARACTER(n)	n
		NCHAR(n)	n
		VARCHAR(n)	Actual number of bytes of a character
		CHARACTER VARYING(n)	Actual number of bytes of a character
		VARCHAR2(n)	Actual number of bytes of a character
		NVARCHAR2(n)	Actual number of bytes of a character
		TEXT	Actual number of bytes of a character
		CLOB	Actual number of bytes of a character
Time data types	Time types	DATE	8
		DATEA (supported only in ORA-compatible mode)	8
		TIME	8
		TIMETZ	12
		TIMESTAMP	8
		TIMESTAMPTZ	8
		SMALLDATETIME	8
		INTERVAL DAY TO SECOND	16
		INTERVAL	16
		RELTIME	4
		ABSTIME	4
		TINTERVAL	12
ROWID type	ROWID type	ROWID	24

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Database Object Position Functions

- pg_relation_filenode(relation regclass)

  Description: Specifies the number of filenodes in a specified relationship.

  Return type: oid

  Note: **pg_relation_filenode** receives the OID or name of a table, an index, a sequence, or a compressed table, and returns the number of **filenode** allocated to it. **filenode** is the basic component of the file name used by the relationship. For most tables, the result is the same as that of **pg_class.relfilenode**. For the specified system directory, **relfilenode** is **0** and this function must be used to obtain the correct value. If a relationship that is not stored is transmitted, such as a view, this function returns a null value.

- pg_relation_filepath(relation regclass)

  Description: Specifies the name of a file path with the specified relationship. It can be used only for non-segment-page relationships.

  Return type: text.

  Description: pg_relation_filepath is similar to pg_relation_filenode, except that pg_relation_filepath returns the whole file path name for the relationship (relative to the data directory *PGDATA* of the database cluster).

  For segment-page relationships, you are advised to use functions or views. For example:

  ```
  SELECT e.*, f.file_name
  FROM gs_seg_extents e, gs_seg_datafiles f
  WHERE e.tablespace_name = f.tablespace_name AND e.bucketnode = f.bucketnode AND e.file_id =
  f.file_id AND e.forknum = f.forknum;
  ```

- get_large_table_name(relfile_node text, threshold_size_gb int8)

  Description: Queries whether the table size (in GB) exceeds the threshold (**threshold_size_gb**) based on the table file code (**relfile_node**). If the size exceeds the threshold, the schema name and table name (in *schemaname*.*tablename* format) are returned, otherwise, **null** is returned.

  Return type: text.

- pg_filenode_relation(tablespace oid, filenode oid)

  Description: Obtains the table names corresponding to the tablespace and relfilenode.

  Return type: regclass

- pg_partition_filenode(partition_oid)

  Description: Obtains **filenode** corresponding to the OID lock of a specified partitioned table.

  Return type: oid

- pg_partition_filepath(partition_oid)

  Description: Specifies the file path name of a specified partition. It can be used only for non-segment-page relationships.

  Return type: text.

  Note: For segment-page relationships, you are advised to use functions or views. For example:

  ```
  SELECT e.*, f.file_name
  FROM gs_seg_extents e, gs_seg_datafiles f
  WHERE e.tablespace_name = f.tablespace_name AND e.bucketnode = f.bucketnode AND e.file_id =
  f.file_id AND e.forknum = f.forknum;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Recycle Bin Object Functions

- gs_is_recycle_object(classid, objid, objname)

  Description: Determines whether an object is in the recycle bin. This function is not supported in distributed mode.

  Return type: Boolean

- gs_is_recycle_obj(classid, objid, objname)

  Description: Determines whether an object is in the recycle bin. **classid** indicates the OID of the system catalog where the object is located, **objid** indicates the OID of the object, and **objname** indicates the name of the object.

  Return type: Boolean

1.6.27.9 Advisory Lock Functions

Advisory lock functions manage advisory locks.

- pg_advisory_lock(key bigint)

  Description: Obtains an exclusive session-level advisory lock.

  Return type: void

  Note: **pg_advisory_lock** locks resources defined by an application. The resources can be identified using a 64-bit or two nonoverlapped 32-bit key values. If another session locks the resources, the function blocks the resources until they can be used. The lock is exclusive. Multiple locking requests are pushed into the stack. Therefore, if the same resource is locked three times, it must be unlocked three times so that it is released to another session.

- pg_advisory_lock(key1 int, key2 int)

  Description: Obtains an exclusive session-level advisory lock.

  Return type: void

  Note: Only users with the SYSADMIN permission can add session-level exclusive advisory locks to the key-value pair (65535, 65535).

- pg_advisory_lock(lock_id int4, lock_id int4, database_name Name)

  Description: Obtains the exclusive advisory lock of a specified database by inputting the lock ID and database name.

  Return type: void

- pg_advisory_lock_shared(key bigint)

  Description: Obtains a shared session-level advisory lock.

  Return type: void

- pg_advisory_lock_shared(key1 int, key2 int)

  Description: Obtains a shared session-level advisory lock.

  Return type: void

  Note: **pg_advisory_lock_shared** works in the same way as **pg_advisory_lock**, except that **pg_advisory_lock_shared** obtains an advisory lock shared with other sessions requesting the lock, while **pg_advisory_lock** obtains an exclusive advisory lock.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_advisory_unlock(key bigint)

  Description: Releases an exclusive session-level advisory lock.

  Return type: Boolean

- pg_advisory_unlock(key1 int, key2 int)

  Description: Releases an exclusive session-level advisory lock.

  Return type: Boolean

  Note: **pg_advisory_unlock** releases the obtained exclusive advisory lock. If the release is successful, the function returns **true**. If the lock was not held, it will return **false**. In addition, an SQL warning will be reported by the server.

- pg_advisory_unlock(lock_id int4, lock_id int4, database_name Name)

  Description: Releases the exclusive advisory lock of a specified database by inputting the lock ID and database name.

  Return type: Boolean

  Note: If the release is successful, **true** is returned. If no lock is held, **false** is returned.

- pg_advisory_unlock_shared(key bigint)

  Description: Releases a shared session-level advisory lock.

  Return type: Boolean

- pg_advisory_unlock_shared(key1 int, key2 int)

  Description: Releases a shared session-level advisory lock.

  Return type: Boolean

  Note: **pg_advisory_unlock_shared** works in the same way as **pg_advisory_unlock**, except that it releases a shared session-level advisory lock.

- pg_advisory_unlock_all()

  Description: Releases all advisory locks owned by the current session.

  Return type: void

  Note: **pg_advisory_unlock_all** releases all advisory locks owned by the current session. The function is implicitly called when the session ends even if the client is abnormally disconnected.

- pg_advisory_xact_lock(key bigint)

  Description: Obtains an exclusive transaction-level advisory lock.

  Return type: void

- pg_advisory_xact_lock(key1 int, key2 int)

  Description: Obtains an exclusive transaction-level advisory lock.

  Return type: void

  Note: pg_advisory_xact_lock works in the same way as pg_advisory_lock, except the lock is automatically released at the end of the current transaction and cannot be released explicitly. Only users with the SYSADMIN permission can add transaction-level exclusive advisory locks to the key-value pair (65535, 65535).

- pg_advisory_xact_lock_shared(key bigint)

  Description: Obtains a shared transaction-level advisory lock.

  Return type: void

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_advisory_xact_lock_shared(key1 int, key2 int)

  Description: Obtains a shared transaction-level advisory lock.

  Return type: void

  Note: pg_advisory_xact_lock_shared works in the same way as pg_advisory_lock_shared, except the lock is automatically released at the end of the current transaction and cannot be released explicitly.

- pg_try_advisory_lock(key bigint)

  Description: Obtains an exclusive session-level advisory lock if available.

  Return type: Boolean

  Note: **pg_try_advisory_lock** is similar to **pg_advisory_lock**, except **pg_try_advisory_lock** does not block the resource until the resource is released. It either immediately obtains the lock and returns **true** or returns **false**, which indicates the lock cannot be performed currently.

- pg_try_advisory_lock(key1 int, key2 int)

  Description: Obtains an exclusive session-level advisory lock if available.

  Return type: Boolean

  Note: Only users with the SYSADMIN permission can add session-level exclusive advisory locks to the key-value pair (65535, 65535).

- pg_try_advisory_lock_shared(key bigint)

  Description: Obtains a shared session-level advisory lock if available.

  Return type: Boolean

- pg_try_advisory_lock_shared(key1 int, key2 int)

  Description: Obtains a shared session-level advisory lock if available.

  Return type: Boolean

  Note: **pg_try_advisory_lock_shared** works in the same way as **pg_try_advisory_lock**, except that **pg_try_advisory_lock_shared** attempts to obtain a shared lock instead of an exclusive lock.

- pg_try_advisory_xact_lock(key bigint)

  Description: Obtains an exclusive transaction-level advisory lock if available.

  Return type: Boolean

- pg_try_advisory_xact_lock(key1 int, key2 int)

  Description: Obtains an exclusive transaction-level advisory lock if available.

  Return type: Boolean

  Note: pg_try_advisory_xact_lock works in the same way as pg_try_advisory_lock, except that the lock, if acquired, is automatically released at the end of the current transaction and cannot be released explicitly. Only users with the SYSADMIN permission can add transaction-level exclusive advisory locks to the key-value pair (65535, 65535).

- pg_try_advisory_xact_lock_shared(key bigint)

  Description: Obtains a shared transaction-level advisory lock if available.

  Return type: Boolean

- pg_try_advisory_xact_lock_shared(key1 int, key2 int)

  Description: Obtains a shared transaction-level advisory lock if available.

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Note: pg_try_advisory_xact_lock_shared works in the same way as pg_try_advisory_lock_shared, except the lock, if acquired, is automatically released at the end of the current transaction and cannot be released explicitly.

- lock_cluster_ddl()

  Description: Attempts to obtain a session-level exclusive advisory lock for all active CNs in the cluster.

  Return type: Boolean

  Note: Only users with the SYSADMIN permission can call this function.

- unlock_cluster_ddl()

  Description: Attempts to add a session-level exclusive advisory lock on the CN.

  Return type: Boolean

  Note: Only users with the SYSADMIN permission can call this function.

- pg_rel_advisory_lock(schema_name cstring, relname cstring)

  Description: Obtains a session-level advisory lock on a table.

  Return type: void

  Note: pg_rel_advisory_lock is for internal use only. You are advised not to use it.

- pg_rel_advisory_unlock(schema_name cstring, relname cstring)

  Description: Releases a session-level advisory lock on a table.

  Return type: void

  Note: pg_rel_advisory_unlock is for internal use only. You are advised not to use it.

1.6.27.10 Logical Replication Functions

NOTE

When using the logical replication functions, set the GUC parameter **wal_level** to **logical**. For details, see "Feature Usage Guidelines > Logical Replication > Logical Decoding (Non-Multi-Primary) > Logical Decoding by SQL Functions" in *Developer Guide*.

- pg_create_logical_replication_slot('slot_name', 'plugin_name', 'output_order')

  Description: Creates a logical replication slot.

  Parameters:

  - slot_name

    Indicates the name of the streaming replication slot.

    Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name.

  - plugin_name

    Indicates the name of the plug-in.

    Value range: a string. The value can be:

    - General plug-ins: **mppdb_decoding** or **sql_decoding**.
    - Supporting only for parallel decoding: **parallel_binary_decoding**, **parallel_json_decoding**, or **parallel_text_decoding**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

– output_order

Indicates the output sequence of the replication slot decoding results. This parameter is optional.

Valid value: **0** or **1**. The default value is **0** on DNs and **1** on CNs.

- **0**: The replication slot decoding results are sorted by transaction COMMIT LSN. In this case, the value of **confirmed_csn** of the replication slot is **0**. This replication slot is called an LSN-based replication slot.
- **1**: The replication slot decoding results are sorted by transaction CSN. In this case, the value of **confirmed_csn** of the replication slot is not **0**. This replication slot is called a CSN-based replication slot, and the **confirmed_flush** value of such replication slot is meaningless.

NOTE

- The CSN-based replication slot is used only for distributed strong consistency decoding. The LSN-based replication slot is required for non-distributed strong consistency decoding.
- When a replication slot is created, if the value of **output_order** is a decimal, the value is rounded off to an integer.

Return types: name and text.

Example:

```
gaussdb=# SELECT * FROM pg_create_logical_replication_slot('slot_lsn','mppdb_decoding',0);
 slotname | xlog_position
----------+---------------
 slot_lsn | 0/6D08B58
(1 row)

gaussdb=# SELECT * FROM pg_create_logical_replication_slot('slot_csn','mppdb_decoding',1);
 slotname | xlog_position
----------+---------------
 slot_csn | 0/59AD800
(1 row)
```

Note: The first return value is the slot name, and the second one has different meanings in LSN-based replication slots and CSN-based replication slots. For an LSN-based replication slot, the value is **confirmed_flush** of the replication slot, indicating that transactions whose commit LSN is less than or equal to the value will not be decoded and output. For a CSN-based replication slot, the value is **confirmed_csn** of the replication slot, indicating that transactions whose CSN is less than or equal to the value will not be decoded and output. Users who call this function must have the SYSADMIN permission or the REPLICATION permission, or inherit permissions of the built-in role gs_role_replication. Currently, this function can be called only on the CN or primary DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

CAUTION

When this function is executed on a CN, if **output_order** is set to **1**, a replication slot with the same name will be created on all CNs and primary DNs. If a replication slot with the same name already exists on some DNs, a CSN-based logical replication slot will be created on other CNs and primary DNs that do not have a replication slot with the same name. In addition, an error message is displayed, indicating that the replication slot already exists. In this case, you need to delete the existing replication slot with the same name on other CNs and primary DN (if the replication slot is an LSN-based logical replication slot, you need to manually delete it on the corresponding node) and create a replication slot on the CN. If you set **output_order** to **0** when creating a replication slot on a CN, no replication slot will be created on DNs. Replication slots created on CNs are only used to identify whether related replication slots exist on DNs.

- pg_create_physical_replication_slot('slot_name', 'isDummyStandby')

  Description: Creates a physical replication slot.

  Parameters:

  – slot_name

    Indicates the name of the streaming replication slot.

    Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name. Replication slots that start with "gs_roach" are reserved for the internal backup tool, GaussRoach. To use them, you need to set **application_name** to **'gs_roach'**.

  – isDummyStandby

    Reserved parameter.

    Type: Boolean.

  Return types: name and text.

  NOTE

  - Users who call this function must have the SYSADMIN permission or the REPLICATION permission, or inherit permissions of the built-in role gs_role_replication.
  - The physical replication slot created by this function does not have **restart_lsn**. Therefore, the slot is considered invalid and will be automatically deleted when the checkpoint is performed.

- pg_drop_replication_slot('slot_name')

  Description: Deletes a streaming replication slot.

  Parameters:

  – slot_name

    Indicates the name of the streaming replication slot.

    Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name. Replication slots that start with "gs_roach" are reserved for the internal

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

backup tool, GaussRoach. To use them, you need to set **application_name** to **'gs_roach'**.

Return type: void.

NOTE

- Users who call this function must have the SYSADMIN permission or the REPLICATION permission, or inherit permissions of the built-in role gs_role_replication.
- When a database is deleted, the logical replication slots on the database are not deleted. You need to manually delete the logical replication slots on the database. Otherwise, historical versions of Xlogs and system catalogs cannot be reclaimed.
- When a logical replication slot is deleted from a CN, if the logical replication slot is an LSN-based logical replication slot, only the replication slot of the current node is deleted. Replication slots with the same name on other nodes are not affected. When a CSN-based logical replication slot with the same name exists on other nodes, no error is reported because some nodes do not have replication slots. In addition, replication slots with the same name on all nodes are successfully deleted. If no replication slot exists on any node, an error is reported.
- If an LSN-based logical replication slot remains on the current CN and a CSN-based logical replication slot with the same name remains on other nodes, deleting the replication slot on the current CN will delete only the local LSN-based logical replication slot. After the deletion is complete, perform the deletion operation again to delete the CSN-based logical replication slots with the same name on other nodes.

- pg_logical_slot_peek_changes('slot_name', 'upto_lsn', upto_nchanges, 'options_name', 'options_value')

  Description: Performs decoding but does not go to the next streaming replication slot. (The decoded result will be returned again during the next decoding.)

  Parameters:

  – slot_name

    Indicates the name of the streaming replication slot.

    Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name.

  – upto_lsn

    For the CSN-based logical replication slot, the decoding is complete until the transaction whose CSN is less than or equal to the value is decoded (a transaction whose CSN is greater than the specified CSN may be decoded). For the LSN-based replication slot, the decoding is complete until the first transaction whose COMMIT LSN is greater than or equal to the value is decoded.

    Value range: a string, for example, **'1/2AAFC60'**, **'0/A060'**, or **'3A/0'** (a hexadecimal uint64 value containing two uint32 values separated by a slash (/); if any uint32 value is **0**, **0** is displayed.) **NULL** indicates that the end position of decoding is not specified.

  – upto_nchanges

    Indicates the number of decoded records (including the **begin** and **commit** timestamps). Assume that there are three transactions, which involve 3, 5, and 7 records, respectively. If **upto_nchanges** is set to **4**, 8 records of the first two transactions will be decoded. Specifically,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

decoding is stopped when the number of decoded records surpasses the value of **upto_nchanges** after decoding in the first two transactions is finished.

Value range: non-negative integers.

NOTE

If any of the **upto_lsn** and **upto_nchanges** values is reached, decoding ends.

- **options**: Specifies optional parameters, consisting of **options_name** and **options_value**.
  - include-xids

    Specifies whether the decoded **data** column includes XIDs.

    Value range: Boolean. The default value is **true**.
    - **false**: The decoded **data** column does not contain XID information.
    - **true**: The decoded **data** column contains XID information.
  - skip-empty-xacts

    Specifies whether to ignore empty transactions during decoding.

    Value range: Boolean. The default value is **false**.
    - **false**: The empty transaction information is not ignored during decoding.
    - **true**: The empty transaction information is ignored during decoding.
  - include-timestamp

    Specifies whether the decoded information includes commit timestamps.

    Value range: Boolean. The default value is **true**.
    - **false**: The decoded information does not contain the **commit** timestamp.
    - **true**: The decoded information contains the **commit** timestamp.
  - only-local

    Specifies whether to decode only local logs.

    Value range: Boolean. The default value is **true**.
    - **false**: Non-local logs and local logs are decoded.
    - **true**: Only local logs are decoded.
  - force-binary

    Specifies whether to output the decoding result in binary format.

    Value range: Boolean. The default value is **false**.
    - **false**: The decoding result is output in text format.
    - The value cannot be set to **true** currently.
  - white-table-list

    Whitelist parameter, including the schema and table name to be decoded.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: a string that contains table names in the whitelist. Different tables are separated by commas (,). An asterisk (*) is used to fuzzily match all tables. Schema names and table names are separated by periods (.). No space character is allowed. The following is an example:

```
SELECT * FROM pg_logical_slot_peek_changes('slot1', NULL, 4096, 'white-table-list',
'public.t1,public.t2');
```

- skip-xid-list

  Specifies the blacklist of transactions (represented by their XIDs) to be skipped during decoding. Transactions cannot be skipped during the upgrade observation period.

  Value range: A character string that contains XIDs, separated by slashes (/). No whitespace characters are allowed. Valid XIDs range from 3 to 18446744073709551615 (that is, the maximum value of uint64). The following is an example:

  ```
  SELECT * FROM pg_logical_slot_peek_changes('slot1', NULL, 4096, 'skip-xid-list',
  '1234/2345');
  ```

- max-txn-in-memory

  Memory control parameter. The unit is MB. If the memory occupied by a single transaction is greater than the value of this parameter, data is flushed to disk.

  Value range: integers from 0 to 100. The default value is **0** (disabled).

- **max-reorderbuffer-in-memory**:

  Memory control parameter. The unit is GB. If the total memory (including the cache) of transactions being concatenated in the sender thread is greater than the value of this parameter, the current decoding transaction is flushed to disk.

  Value range: integers from 0 to 100. The default value is **0** (disabled).

- **desc-memory-limit**:

  A memory control parameter, in MB. When the total memory of table metadata maintained by a logical decoding task is greater than the value of this parameter, some table metadata will be evicted.

  Value range: an integer ranging from 10 to 1048576. The default value is **200**.

- include-user

  Specifies whether the BEGIN logical log of a transaction records the username of the transaction.

  Value range: Boolean. The default value is **false**.

  - **false**: The BEGIN logical log of a transaction does not record the username of the transaction.
  - **true**: The BEGIN logical log of a transaction records the username of the transaction.

- exclude-userids

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Lists the OIDs of blacklisted users.

Value range: OIDs of blacklisted users. Multiple OIDs are separated by commas (,). The system does not verify the existence of the specified OIDs.

- exclude-users

  Specifies the name of a blacklisted user.

  Value range: names of blacklisted users. Multiple names are separated by commas (,). **dynamic-resolution** specifies whether to dynamically parse and identify usernames. If the decoding is interrupted because the user does not exist and the corresponding blacklisted user does not exist at the time when logs are generated, you can set **dynamic-resolution** to **true** or delete the username from the blacklist to start decoding and continue to obtain logical logs.

- enable-ddl-decoding

  (Logical decoding control parameter) Specifies whether to enable logical decoding for DDL statements.

  Value range: Boolean. The default value is **false**.

  - **false**: Logical decoding of DDL statements is disabled.
  - **true**: Logical decoding of DDL statements is enabled.

- enable-ddl-json-format

  (Logical decoding control parameter) Specifies the DDL reverse parsing process and output format.

  Value range: Boolean. The default value is **false**.

  - **false**: The DDL statement reverse parsing result is output in text format.
  - **true**: The DDL statement reverse parsing result is output in JSON format.

- **skip-generated-columns**:

  (Logical decoding control parameter) Specifies whether to skip generating decoding results for stored generated columns. This parameter remains ineffective on old tuples (those that have undergone an UPDATE or DELETE operation), meaning that decoding results will still be generated for their stored generated columns. Currently, the generated column syntax is not supported in distributed mode. This option is not supported for CN decoding.

  Value range: Boolean. The default value is **false** or **off**.

  - **true**/**on**: The decoding result of stored generated columns is not output.
  - **false**/**off**: The decoding result of stored generated columns is output.

  Virtual generated columns are not controlled by this parameter. DML decoding results will not be generated for virtual generated columns.

- dynamic-resolution

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies whether to dynamically parse the names of blacklisted users.

Value range: Boolean. The default value is **true**.

- **false**: An error is reported and the logical decoding exits when the decoding detects that a user does not exist in blacklist **exclude-users**.
- **true**: Decoding continues when it detects that a user does not exist in blacklist **exclude-users**.

- **decode-sequence**:

  (Logical decoding control parameter) Specifies whether to generate decoding results for logs that record sequence value changes.

  Value range: Boolean. The default value is **false**.

  - The value cannot be set to **true** currently.
  - **false**: The decoding result of the change log of the sequence value is not output.

  NOTICE

  Currently, **decode-sequence** can only be set to **false**. If **decode-sequence** is set to **true**, an error is reported when decoding is started and the decoding process exits.

- **enable-decode-position**:

  (Logical decoding control parameter) Specifies whether to send decoding position logs when the logs are read during logical decoding.

  Value range: Boolean. The default value is **false**.

  - **true**: Position logs are generated.
  - **false**: Position logs are not generated.

- enable-rowno:

  (Logical decoding control parameter) Specifies whether to output the **rowno** system column for new and old tuples in INSERT/UPDATE/DELETE operations when decoding a primary key–free table that includes the **rowid** system column and has the **REPLICA IDENTITY** option set to **FULL** or any other value equivalent to **FULL**. When using this option to output the **rowno** system column, ensure that the target database contains columns with the same name and type. Otherwise, logical decoding logs may fail to be replayed. If you have any questions, contact Huawei technical support.

  Value range: Boolean

  - **true**: The **rowno** system column is output.
  - **false**: The **rowno** system column is not output.

  Default value: **false**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- This option is ineffective for the sql_decoding plug-in.
- The actual effect of the **REPLICA IDENTITY** option is equivalent to **FULL**. Currently, it refers to the **NOTHING** and **DEFAULT** options of the Ustore table. For details, see the option description in **REPLICA IDENTITY { DEFAULT | USING INDEX index_name | FULL | NOTHING }**.

NOTE

For details about other options, see "Feature Usage Guidelines > Logical Replication > Logical Decoding (Non-Multi-Primary) > Logical Decoding Options" in *Developer Guide*.

Return types: text, xid, and text.

Example:

```
gaussdb=# EXECUTE DIRECT ON (datanode1) 'SELECT * FROM
pg_logical_slot_peek_changes(''slot_lsn'',NULL,4096,''skip-empty-xacts'',''on'')';
 location  |  xid  |                                                
```

Note: In the preceding function example, **datanode1** is the name of the DN to be decoded, and **slot_lsn** and **slot_csn** are the names of the logical replication slots.

The decoding result returned by the function contains three columns, corresponding to the preceding return value types, which are the LSN (for an LSN-based replication slot) or CSN (for a CSN-based replication slot), XID, and decoded content, respectively. If the **location** column indicates the CSN, the value of the **location** column is updated only when the commit logs are decoded. Users who call this function must have the SYSADMIN permission or the REPLICATION permission, or inherit permissions of the built-in role gs_role_replication.

NOTE

On the CN, this function cannot be executed in a CSN-based replication slot (**confirmed_csn** is not **0**).

- pg_logical_slot_get_changes('slot_name', 'upto_lsn', upto_nchanges, 'options_name', 'options_value')

  Description: Performs decoding and goes to the next streaming replication slot.

  Parameter description: Same as that for **pg_logical_slot_peek_changes**. For details, see **pg_logical_slot_peek_changes('slot_name', 'upto_lsn', upto_nchanges, 'options_name', 'options_value')**.

  Note: Users who call this function must have the SYSADMIN permission or the REPLICATION permission, or inherit permissions of the built-in role gs_role_replication.

  NOTE

  On the CN, this function cannot be executed in a CSN-based replication slot (**confirmed_csn** is not **0**). When this function is called on the standby DN, the corresponding logical replication slot on the primary DN is updated synchronously. If this function is executed on the standby DN, a WAL sender of the primary DN is occupied when the replication slot number on the primary DN is updated. The logical decoding function reserves a WAL sender for each logical replication slot. Therefore, if this function is executed in normal scenarios, the logical replication slot number on the primary DN is updated normally. If this function is executed continuously in a short period of time, the primary DN fails to update the slot number and no error is reported.

- pg_logical_slot_peek_binary_changes('slot_name', 'upto_lsn', upto_nchanges, 'options_name', 'options_value')

  Description: Performs decoding in binary mode and does not go to the next streaming replication slot. (The decoded data can be obtained again during the next decoding.)

  Parameters:

  – slot_name

    Indicates the name of the streaming replication slot.

    Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name.

  – upto_lsn

    For the CSN-based logical replication slot, the decoding is complete until the transaction whose CSN is less than or equal to the value is decoded

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

(a transaction whose CSN is greater than the specified CSN may be decoded). For the LSN-based replication slot, the decoding is complete until the first transaction whose COMMIT LSN is greater than or equal to the value is decoded.

Value range: a string, for example, **'1/2AAFC60'**, **'0/A060'**, or **'3A/0'** (a hexadecimal uint64 value containing two uint32 values separated by a slash (/); if any uint32 value is **0**, **0** is displayed.) **NULL** indicates that the end position of decoding is not specified.

- upto_nchanges

  Indicates the number of decoded records (including the **begin** and **commit** timestamps). Assume that there are three transactions, which involve 3, 5, and 7 records, respectively. If **upto_nchanges** is set to **4**, 8 records of the first two transactions will be decoded. Specifically, decoding is stopped when the number of decoded records surpasses the value of **upto_nchanges** after decoding in the first two transactions is finished.

  Value range: non-negative integers.

  NOTE

  If any of the **upto_lsn** and **upto_nchanges** values is reached, decoding ends.

- **options**: Specifies optional parameters, consisting of **options_name** and **options_value**.

  - include-xids

    Specifies whether the decoded **data** column includes XIDs.

    Value range: Boolean. The default value is **true**.

    - **false**: The decoded **data** column does not contain XID information.
    - **true**: The decoded **data** column contains XID information.

  - skip-empty-xacts

    Specifies whether to ignore empty transactions during decoding.

    Value range: Boolean. The default value is **false**.

    - **false**: The empty transaction information is not ignored during decoding.
    - **true**: The empty transaction information is ignored during decoding.

  - include-timestamp

    Specifies whether the decoded information includes commit timestamps.

    Value range: Boolean. The default value is **true**.

    - **false**: The decoded information does not contain the **commit** timestamp.
    - **true**: The decoded information contains the **commit** timestamp.

  - only-local

    Specifies whether to decode only local logs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: Boolean. The default value is **true**.

- **false**: Non-local logs and local logs are decoded.
- **true**: Only local logs are decoded.

- force-binary

  This parameter is useless for the function.

  Value range: Boolean. The result is output in binary format.

- white-table-list

  Specifies the whitelist of schemas and tables to be decoded.

  Value range: a string that contains table names in the whitelist. Different tables are separated by commas (,). An asterisk (*) is used to fuzzily match all tables. Schema names and table names are separated by periods (.). No space character is allowed. Example: **select * from pg_logical_slot_peek_binary_changes('slot1', NULL, 4096, 'white-table-list', 'public.t1,public.t2');**

- skip-xid-list

  Specifies the blacklist of transactions (represented by their XIDs) to be skipped during decoding. Transactions cannot be skipped during the upgrade observation period.

  Value range: A character string that contains XIDs, separated by slashes (/). No whitespace characters are allowed. Valid XIDs range from 3 to 18446744073709551615 (that is, the maximum value of uint64). The following is an example:

  ```
  SELECT * FROM pg_logical_slot_peek_changes('slot1', NULL, 4096, 'skip-xid-list', '1234/2345');
  ```

- max-txn-in-memory

  Memory control parameter. The unit is MB. If the memory occupied by a single transaction is greater than the value of this parameter, data is flushed to disk.

  Value range: integers from 0 to 100. The default value is **0** (disabled).

- **max-reorderbuffer-in-memory**:

  Memory control parameter. The unit is GB. If the total memory (including the cache) of transactions being concatenated in the sender thread is greater than the value of this parameter, the current decoding transaction is flushed to disk.

  Value range: integers from 0 to 100. The default value is **0** (disabled).

- **desc-memory-limit**:

  A memory control parameter, in MB. When the total memory of table metadata maintained by a logical decoding task is greater than the value of this parameter, some table metadata will be evicted.

  Value range: an integer ranging from 10 to 1048576. The default value is **200**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- include-user

  Specifies whether the BEGIN logical log of a transaction records the username of the transaction.

  Value range: Boolean. The default value is **false**.

  - **false**: The BEGIN logical log of a transaction does not record the username of the transaction.
  - **true**: The BEGIN logical log of a transaction records the username of the transaction.

- exclude-userids

  Lists the OIDs of blacklisted users.

  Value range: OIDs of blacklisted users. Multiple OIDs are separated by commas (,). The system does not verify the existence of the specified OIDs.

- exclude-users

  Specifies the name of a blacklisted user.

  Value range: names of blacklisted users. Multiple names are separated by commas (,). **dynamic-resolution** specifies whether to dynamically parse and identify usernames. If the decoding is interrupted because the user does not exist and the corresponding blacklisted user does not exist at the time when logs are generated, you can set **dynamic-resolution** to **true** or delete the username from the blacklist to start decoding and continue to obtain logical logs.

- dynamic-resolution

  Specifies whether to dynamically parse the names of blacklisted users.

  Value range: Boolean. The default value is **true**.

  - **false**: An error is reported and the logical decoding exits when the decoding detects that a user does not exist in blacklist **exclude-users**.
  - **true**: Decoding continues when it detects that a user does not exist in blacklist **exclude-users**.

- **decode-sequence**:

  (Logical decoding control parameter) Specifies whether to generate decoding results for logs that record sequence value changes.

  Value range: Boolean. The default value is **false**.

  - The value cannot be set to **true** currently.
  - **false**: The decoding result of the change log of the sequence value is not output.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

Currently, **decode-sequence** can only be set to **false**. If **decode-sequence** is set to **true**, an error is reported when decoding is started and the decoding process exits.

- **enable-decode-position**:

  (Logical decoding control parameter) Specifies whether to send decoding position logs when the logs are read during logical decoding.

  Value range: Boolean. The default value is **false**.

  - **true**: Position logs are generated.
  - **false**: Position logs are not generated.

- enable-rowno:

  (Logical decoding control parameter) Specifies whether to output the **rowno** system column for new and old tuples in INSERT/UPDATE/DELETE operations when decoding a primary key–free table that includes the **rowid** system column and has the **REPLICA IDENTITY** option set to **FULL** or any other value equivalent to **FULL**. When using this option to output the **rowno** system column, ensure that the target database contains columns with the same name and type. Otherwise, logical decoding logs may fail to be replayed. If you have any questions, contact Huawei technical support.

  Value range: Boolean

  - **true**: The **rowno** system column is output.
  - **false**: The **rowno** system column is not output.

  Default value: **false**.

NOTICE

- This option is ineffective for the sql_decoding plug-in.
- The actual effect of the **REPLICA IDENTITY** option is equivalent to **FULL**. Currently, it refers to the **NOTHING** and **DEFAULT** options of the Ustore table. For details, see the option description in **REPLICA IDENTITY { DEFAULT | USING INDEX index_name | FULL | NOTHING }**.

NOTE

Some configuration options can be configured in functions but do not take effect. For details, see "Feature Usage Guidelines > Logical Replication > Logical Decoding (Non-Multi-Primary) > Logical Decoding Options" in *Developer Guide*.

Return types: text, xid, and bytea.

Note: The function returns the decoding results. Each decoding result contains three columns, corresponding to the above return types and indicating the LSN position, XID, and decoded content in binary format, respectively. Users who call this function must have the SYSADMIN permission or the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

REPLICATION permission, or inherit permissions of the built-in role gs_role_replication.

NOTE

On the CN, this function cannot be executed in a CSN-based replication slot (**confirmed_csn** is not **0**).

- pg_logical_slot_get_binary_changes('slot_name', 'upto_lsn', upto_nchanges, 'options_name', 'options_value')

  Description: Performs decoding in binary mode and does not go to the next streaming replication slot.

  Parameters: same as **pg_logical_slot_peek_binary_changes**.

  Note: Users who call this function must have the SYSADMIN permission or the REPLICATION permission, or inherit permissions of the built-in role gs_role_replication.

  NOTE

  On the CN, this function cannot be executed in a CSN-based replication slot (**confirmed_csn** is not **0**). This function cannot be executed on the standby DN.

- pg_replication_slot_advance ('slot_name', 'upto_lsn')

  Description: Directly goes to the streaming replication slot for a specified CSN (for CSN-based logical replication slots) or LSN (for LSN-based replication slots), without outputting any decoded result.

  Parameters:

  – slot_name

    Indicates the name of the streaming replication slot.

    Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name.

  – upto_lsn

    For the CSN-based logical replication slot, it indicates the target CSN before which logs are decoded. During the next decoding, only the transaction results whose CSN is greater than this value will be output. If the input CSN is smaller than the value of **confirmed_csn** recorded in the current streaming replication slot, the input CSN is used. If the input CSN is greater than the latest CSN that can be obtained, the latest CSN is directly used. If the input CSN is smaller than the CSN of the latest physical log, the logs before first CSN greater than the input CSN are decoded.

    For the LSN-based logical replication slot, it indicates the target LSN before which logs are decoded. During the next decoding, only the transaction results whose LSN is greater than this value will be output. If the input LSN is smaller than the LSN before which logs are decoded in the current streaming replication slot, an error is reported. If the input LSN is greater than the LSN of the latest physical log, the LSN of the latest physical log is directly used. If the input LSN is smaller than the LSN of the latest physical log, the logs before first LSN greater than the input LSN are decoded.

    Value range: a string, for example, **'1/2AAFC60'**, **'0/A060'**, or **'3A/0'** (a hexadecimal uint64 value containing two uint32 values separated by a

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

slash (/); if any uint32 value is **0**, **0** is displayed.) **NULL** indicates that the end position of decoding is not specified.

Return types: name and text.

NOTE

- The return value corresponds to **slot_name** and the actual position (CSN or LSN). Users who call this function must have the SYSADMIN permission or the REPLICATION permission, or inherit permissions of the built-in role gs_role_replication.
- On the CN, this function cannot be executed in a CSN-based replication slot (**confirmed_csn** is not **0**). This function can be executed on the backup replication slot on the standby DN. This function can be executed on the logical replication slot on the standby DN and the corresponding logical replication slot on the primary DN is updated synchronously. If this function is executed on the standby DN, a WAL sender of the primary DN is occupied when the replication slot number on the primary DN is updated. The logical decoding function reserves a WAL sender for each logical replication slot. Therefore, if this function is executed in normal scenarios, the logical replication slot number on the primary DN is updated normally. If this function is executed continuously in a short period of time, the primary DN fails to update the slot number and no error is reported.

- pg_get_replication_slots()

  Description: Obtains the replication slot list.

  Table 1-136 Return values

Column	Data Type	Description
slot_name	text	Name of the replication slot.
plugin	text	Name of the output plug-in corresponding to the logical replication slot.
slot_type	text	Replication slot type. • **physical**: physical replication slot. • **logical**: logical replication slot.
datoid	oid	OID of the database where the replication slot resides.
active	Boolean	Specifies whether the replication slot is activated. • **f**: not activated. • **t**: activated.
xmin	xid	XID of the earliest transaction that the database must reserve for the replication slot.
catalog_xmin	xid	XID of the earliest system catalog-involved transaction that the database must reserve for the logical replication slot.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Data Type	Description
restart_lsn	text	Physical location of the earliest Xlog required by the replication slot.
dummy_standby	Boolean	Reserved parameter.
confirmed_flush	text	Location of the received log confirmed by the client (dedicated for logical replication slots).
confirmed_csn	xid	CSN corresponding to the last transaction in the received log confirmed by the client (dedicated for logical replication slots).
dictionary_csn_min	xid	CSN of the earliest dictionary table-involved transaction that the database must reserve for the logical replication slot.
slot_dictionary_type	text	Type of the logical replication slot.
intra_cluster	Boolean	Specifies whether the replication slot needs to synchronize data between clusters. • **true**: no. • **false**: yes.
remote_stream	Boolean	Specifies whether the replication slot is decoding the remote log stream.

Example:

```
Run the following command on the DN:
gaussdb=# SELECT * FROM pg_get_replication_slots();
 slot_name |     plugin     | slot_type | datoid | active | xmin | catalog_xmin | restart_lsn |
dummy_standby | confirmed_flush | confirmed_csn | dictionary_csn_min | slot_dictionary_type |
intra_cluster | remote_stream
-----------+----------------+-----------+--------+--------+------+--------------+-------------+---------------
+-----------------+--------------+--------------------+----------------------
 dn_6002   |              | physical |      0 | t      |      |              | 0/3622B528  | f             |
|             |                  |                    |
 f         |     f
 dn_6003   |              | physical |      0 | t      |      |              | 0/3622B528  | f             |
|             |                  |                    |
 f         |     f

 slot_lsn  | mppdb_decoding | logical   | 131072 | f      |      |        66658 | 0/36252350  | f             |
0/362523D0      |              |                  |                    |
 f         |     f
 slot_test | mppdb_decoding | logical   | 131072 | f      |      |        66658 | 0/36251718  | f
|             |     10025527 |                  |                    |
 f         |     f
(4 rows)
Run the following command on the CN:
gaussdb=# SELECT * FROM pg_get_replication_slots();
 slot_name |     plugin     | slot_type | datoid | active | xmin | catalog_xmin | restart_lsn |
dummy_standby | confirmed_flush | confirmed_csn | dictionary_csn_min | slot_dictionary_type |
intra_cluster | remote_stream
-----------+----------------+-----------+--------+--------+------+--------------+-------------+---------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
+-----------------+---------------+--------------------+----------------------
 slot_test | mppdb_decoding | logical   | 139264 | f      |      |             |            | f            |
|              |                |                    |
  f          |     f
(1 row)
```

NOTICE

When a query is performed on a DN, the **confirmed_csn** query result of the LSN-based logical replication slot is empty, and the **confirmed_flush** query result of the CSN-based logical replication slot is empty. When a query is performed on the CN, **catalog_xmin**, **restart_lsn**, **confirmed_flush**, and **confirmed_csn** of the CSN-based logical replication slot are not displayed and the query result is empty.

- pg_logical_get_area_changes('start_lsn', 'upto_lsn', upto_nchanges, 'decoding_plugin', 'xlog_path', 'options_name', 'options_value')

  Description: Specifies an LSN range or an Xlog file for decoding.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

The constraints are as follows:

- The current network and hardware environment are normal.
- It is recommended that the size of a single tuple be less than or equal to 500 MB. If the size ranges from 500 MB to 1 GB, an error is reported.
- Log decoding is not supported for operations that are not recorded in Xlogs, such as data page replication.
- When making API calls, set **wal_level** to **logical**. The system can only parse the log files generated when **wal_level** is **logical**. If the Xlog file being used is not at the logical level, an error will occur, and the file will not be decoded.
- To successfully parse Xlog files, use a copy of a fully homogeneous DN and avoid performing any DDL operations or VACUUM FULL in the database so that the system can find the meta information associated with data.
- Do not read an excessive number of Xlog files at a time. When selecting range decoding instead of file decoding, it is advisable to read one Xlog file each time. Decoding an Xlog file typically requires two to three times the memory of the file size.
- Logs before VACUUM FULL cannot be decoded.
- Xlog files generated before scale-out cannot be decoded.
- When **enable-ddl** is disabled, columns of the TOAST, CLOB, and BLOB types cannot be decoded. When enable-ddl is enabled, columns of related types can be decoded.
- This mode decodes content based on the data stored in Xlog text rather than transactions. Therefore, data that is not in the Xlog cannot be decoded. If related Xlogs are missing, the missing logs cannot be decoded.
- When **enable-ddl** is disabled: If no file is specified for decoding from the decoding point, the system will check for DDL operations between the decoding start point and the latest redo value. If any DDL operation is detected, no decoding will take place. If a file is specified for decoding, the system will check for DDL operations between the start point and the last readable content of that file, as well as between the start point of Xlogs in the data directory and the latest redo value. If any DDL operation is detected, decoding for all tables will be deactivated.
- When **enable-ddl** is disabled, columns of the TOAST, CLOB, and BLOB types cannot be decoded. When enable-ddl is enabled, columns of related types can be decoded.
- When **enable-ddl** is enabled, decoding of DDL statements is supported in the range and DDL statements can be output. For details about the output DDL statement range, see "Feature Usage Guidelines > Logical Replication > Logical Decoding (Non-Multi-Primary) > Logical Decoding of DDL Statements" in *Developer Guide*.
- To parse the UPDATE and DELETE statements of a table, you need to configure the **REPLICA IDENTITY** attribute for the table. If the table does not have a primary key, set **REPLICA IDENTITY** to **FULL**. For details, see the **REPLICA IDENTITY { DEFAULT | USING INDEX index_name | FULL | NOTHING }** column in **ALTER TABLE**.
- In M-compatible mode, the decoded data value of the float4 type is output as the actual stored value, which is displayed differently from that of the SQL query.

Note: When separation-of-duties is enabled, only the initial database user can call the function. When separation-of-duties is disabled, the SYSADMIN permission is required.

Parameters:

– start_lsn

  Specifies the LSN at the start of decoding.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: character strings in the format of *xlogid*/*xrecoff*, for example, '1/2AAFC60'. **NULL** indicates that the start position of decoding logs is not limited.

– upto_lsn

Specifies the LSN at the end of decoding.

Value range: character strings in the format of *xlogid*/*xrecoff*, for example, '1/2AAFC60'. **NULL** indicates that the end position of decoding is not specified.

– upto_nchanges

Indicates the number of decoded records (including the **begin** and **commit** timestamps). Assume that there are three transactions, which involve 3, 5, and 7 records, respectively. If **upto_nchanges** is set to **4**, 8 records of the first two transactions will be decoded. Specifically, decoding is stopped when the number of decoded records surpasses the value of **upto_nchanges** after decoding in the first two transactions is finished.

Value range: non-negative integers.

NOTE

If any of the **LSN** and **upto_nchanges** values are reached, decoding ends.

– decoding_plugin

Decoding plug-in, which is a .so plug-in that specifies the output format of the decoded content.

Value range: **mppdb_decoding** and **sql_decoding**.

– xlog_path

Decoding plug-in, which specifies the Xlog absolute path of the decoding file.

Value range: **NULL** or a character string of the absolute path of the Xlog file.

– **options**: This parameter is optional and consists of multiple pairs of **options_name** and **options_value**. You can retain the default values.

  ▪ include-xids

  Specifies whether the decoded **data** column includes XIDs.

  Value range: Boolean. The default value is **true**.

  false: The decoded **data** column does not contain XID information.

  true: The decoded **data** column contains XID information.

  ▪ include-timestamp

  Specifies whether the decoded information includes commit timestamps.

  Value range: Boolean. The default value is **true**.

  false: The decoded information does not contain the **commit** timestamp.

  true: The decoded information contains the **commit** timestamp.

  ▪ only-local

  Specifies whether to decode only local logs.

  Value range: Boolean. The default value is **true**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

false: Non-local logs and local logs are decoded.

true: Only local logs are decoded.

- force-binary

  Specifies whether to output the decoding result in binary format.

  Value range: Boolean. The default value is **false**.

  false: The decoding result is output in text format.

  The value cannot be set to **true** currently.

- white-table-list

  Specifies the whitelist of schemas and tables to be decoded. Value range: a string that contains table names in the whitelist. Different tables are separated by commas (,). An asterisk (*) is used to fuzzily match all tables. Schema names and table names are separated by periods (.). No space character is allowed.

- white-database-list

  Whitelist parameter, including names of databases to be decoded. Value range: a character string that contains database names in the whitelist. Different database names are separated by commas (,). No space character is allowed. If **white-database-list** is not set, only the database Xlogs of the current connection are decoded. If **white-database-list** is set, only the database Xlogs specified in **white-database-list** are decoded. If the system database is decoded, ordinary table data in the system database can be retrieved.

- enable-ddl

  Decoding that contains DDL syntax is supported. If the DDL syntax exists between the start position and the latest log, the function is not affected. The value can be **0** or **1**.

  If this parameter is set to **1**: enable-ddl is enabled. This function supports decoding of xlog areas containing DDL syntax.

  If this parameter is set to **0**, enable-ddl is disabled by default. This function does not support decoding of xlog areas containing DDL syntax. If a DDL statement is detected, the function exits directly.

  If a column has been deleted from the table definition, the column name of the decoded data will be replaced by the UNDEFINED placeholder when **enable-ddl** is enabled.

- **data-limit**:

  Specifies the amount of output data after decoding. The unit is KB.

  Value range: an integer greater than or equal to 0. The default value is **0**, indicating that the size of the decoding result is not limited.

  The input parameter **data-limit** of this function is used together with the GUC parameter **data-limit**. The decoding stops if it exceeds either of the **data-limit** values.

  Rules:

  1) If the GUC parameter **data-limit** is set to **0**, and **data-limit** for function decoding is not set or is set to **0**, the decoding result has no upper limit.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2) If the GUC parameter **data-limit** is not set, and **data-limit** for function decoding is set to **0**, the upper limit of the decoding result is 10 GB.
3) If the GUC parameter **data-limit** is set, and **data-limit** for function decoding is not set or is set to **0**, the upper limit of the decoding result is the value of the GUC parameter **data-limit**.
4) If both the GUC parameter **data-limit** and **data-limit** of the function are set, the smaller value is used as the upper limit of the decoding result.

- xlog_dir

  The archive log folder can be decoded, and the folder name is the same as the first eight digits of the WAL file. Decoding can be performed only when **enable-ddl** is set to **1** and the input **start_lsn** and **upto_lsn** are valid intervals. Value range: full path of the folder.

  For details, see "Feature Usage Guidelines > Logical Replication > Logical Decoding (Non-Multi-Primary) > Data Restoration by Logical Decoding" in *Developer Guide*.

  ```
  gaussdb=# SELECT * FROM pg_logical_get_area_changes('1/F9FC24C8', '1/FA67B2A0', NULL, 'sql_decoding', NULL ,'enable-ddl','1','xlog_dir','/base/username/xlog_dir/00000000');
  ```

- **decode-sequence**:

  (Logical decoding control parameter) Specifies whether to generate decoding results for logs that record sequence value changes.

  Value range: Boolean. The default value is **false**.

  - The value cannot be set to **true** currently.
  - **false**: The decoding result of the change log of the sequence value is not output.

  NOTICE

  Currently, **decode-sequence** can only be set to **false**. If **decode-sequence** is set to **true**, an error is reported when decoding is started.

- enable-rowno:

  (Logical decoding control parameter) Specifies whether to output the **rowno** system column for new and old tuples in INSERT/UPDATE/DELETE operations when decoding a primary key–free table that includes the **rowid** system column and has the **REPLICA IDENTITY** option set to **FULL** or any other value equivalent to **FULL**. When using this option to output the **rowno** system column, ensure that the target database contains columns with the same name and type. Otherwise, logical decoding logs may fail to be replayed. If you have any questions, contact Huawei technical support.

  Value range: Boolean

  - **true**: The **rowno** system column is output.
  - **false**: The **rowno** system column is not output.

  Default value: **false**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- This option is ineffective for the sql_decoding plug-in.
- The actual effect of the **REPLICA IDENTITY** option is equivalent to **FULL**. Currently, it refers to the **NOTHING** and **DEFAULT** options of the Ustore table. For details, see the option description in **REPLICA IDENTITY { DEFAULT | USING INDEX index_name | FULL | NOTHING }**.

NOTE

pg_logical_get_area_changes takes effect only for the preceding parameters. It does not take effect for the other parameters such as **max-txn-in-memory**, **include-user**, **exclude-userids**, **exclude-users** and **dynamic-resolution** (for details, see pg_logical_slot_peek_changes).

Return type: text, xid, text

Example:

```
- Perform DML operations on a CN. You can query the DML operations executed on the current DN.
gaussdb=# SELECT * FROM pg_logical_get_area_changes('0/502E418', NULL, NULL, 'sql_decoding',
NULL);
 location  |  xid  |                                 data
-----------+-------+--------------------------------------------------------------------------------
 0/502E448 | 17365 | insert into public.t1 values (1, 1);
 0/502E5A0 | 17365 | COMMIT 17365 (at 2023-11-01 11:28:43.92526+08) 2010016
 0/502E5D0 | 17366 | delete from public.t1 where a = 1;insert into public.t1 values (1, 2);
 0/502E6D8 | 17366 | COMMIT 17366 (at 2023-11-01 11:28:45.889283+08) 2010017
 0/502E7B8 | 17367 | delete from public.t1 where a = 1;
 0/502E8B0 | 17367 | COMMIT 17367 (at 2023-11-01 11:28:48.301307+08) 2010018
(6 rows)
```

Note: The function returns the decoding results. Each decoding result contains three columns, corresponding to the above return types and indicating the LSN, XID, and decoded content, respectively.

- gs_get_parallel_decode_status()

  Description: Monitors the lengths of the read log queue and decoding result queue of each decoder thread to identify performance bottlenecks in parallel decoding.

  Return type: text, int, text, text, text, int64, int64, TimestampTz

  Example:

```
gaussdb=# SELECT * FROM gs_get_parallel_decode_status();
 slot_name | parallel_decode_num | read_change_queue_length  | decode_change_queue_length |
reader_lsn | working_txn_cnt | working_txn_memory |      decoded_time
-----------+---------------------+---------------------------+----------------------------+------------
+-----------------+--------------------+------------------------
 slot1     |                   2 | queue0: 1005, queue1: 320 | queue0: 63, queue1: 748    | 0/1DCE2578
|              42 |          192927504 | 2023-01-10 11:18:22+08
(1 row)
```

  Note: In the return values, **slot_name** indicates the replication slot name, **parallel_decode_num** indicates the number of parallel decoder threads in the replication slot, **read_change_queue_length** indicates the current length of the log queue read by each decoder thread, **decode_change_queue_length** indicates the current length of the decoding result queue of each decoder thread, **reader_lsn** indicates the log location read by the reader thread, **working_txn_cnt** indicates the number of transactions being concatenated in the current sender thread, **working_txn_memory** indicates the total memory (in bytes) occupied by the concatenation transactions in the sender thread,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

and **decoded_time** indicates the time of the latest WAL decoded by the replication slot.

NOTICE

The value of **decoded_time** comes from checkpoint logs and transaction commit logs, which has a certain error. If no log containing the time is decoded, "2000-01-01 08:00:00+08" (depending on the time zone set in the database) is displayed.

- gs_get_slot_decoded_wal_time(slot_name)

  Description: Queries the time of the latest WAL decoded by a replication slot.

  Parameters:

  – slot_name

    Specifies the name of the replication slot to be queried.

    Value range: a string, supporting only letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.).

  Example:

```
gaussdb=# SELECT * FROM gs_get_slot_decoded_wal_time('replication_slot');
 gs_get_slot_decoded_wal_time
------------------------------
 2023-01-10 11:25:22+08
(1 row)
```

  Note: The returned values indicate the time of the latest WAL decoded by the replication slot.

NOTICE

- The returned time is extracted from checkpoint logs and transaction commit logs, with a certain margin of error. If no log containing the time is decoded, "2000-01-01 08:00:00+08" (depending on the time zone set in the database) is displayed. When you query the time of the most recent decoded WAL at a non-existent logical replication slot, NULL will be returned. In gsql, you can use **\pset null 'null'** to customize how NULL is displayed.
- Serial decoding is not supported.

- gs_logical_parallel_decode_status('slot_name')

  Description: Obtains the decoding statistics of active replication slots for parallel logical decoding, including 29 rows of metrics.

  The descriptions of the metrics are listed in the following table.

```
Record - (stat_id int, stat_name TEXT, value TEXT)
```

Table 1-137 Metrics

Metric	Description
slot_name	Name of the logical replication slot.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Metric	Description
reader_lsn	Location of the logical logs to be decoded.
wal_read_total_time	Time required for loading the log module, in microseconds.
wal_wait_total_time	Time required for waiting for the log module, in microseconds.
parser_total_time	Time required for the reader thread to process data, in microseconds.
decoder_total_time	Total processing duration of all decoder threads, in microseconds.
sender_total_time	Time required for the sender thread to process data, in microseconds.
net_send_total_time	Time required for the network to send logical logs, in microseconds.
net_wait_total_time	Time required for the network to wait for sending logical logs, in microseconds.
net_send_total_bytes	Number of bytes of logical logs sent by the network.
transaction_count	Number of transactions.
big_transaction_count	Number of large transactions.
max_transaction_tuples	Maximum number of transaction operation tuples.
sent_transaction_count	Number of transactions sent (by the local database).
spill_disk_transaction_count	Number of flushed transactions.
spill_disk_bytes	Total number of bytes flushed to disk.
spill_disk_count	Number of disk flushing times.
input_queue_full_count	Total number of times that the input queues of all decoder threads are full.
output_queue_full_count	Total number of times that the output queues of all decoder threads are full.
dml_count	Total number of DML statements in WALs decoded by each decoder thread in the local database.
dml_filtered_count	Total number of DML statements in WALs decoded and filtered by each decoder thread in the local database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Metric	Description
toast_count	Number of modified TOAST table rows.
candidate_catalog_xmin	Indicates the **catalog_xmin** candidate point of the current logical replication slot.
candidate_xmin_lsn	Specifies the log confirmation receiving point required for the LSN-based logical replication slot to update **catalog_xmin**.
candidate_restart_valid	Specifies the log confirmation receiving point required for the LSN-based logical replication slot to update **restart_lsn**.
candidate_restart_lsn	Indicates the **restart_lsn** candidate point of the current logical replication slot.
candidate_xmin_csn	Specifies the log confirmation receiving point required for the CSN-based logical replication slot to update **catalog_xmin**.
candidate_restart_csn	Specifies the log confirmation receiving point required for the CSN-based logical replication slot to update **restart_lsn**.
candidate_csn_min	Candidate point of the current logical replication slot dictionary_csn_min. For details, see the **pg_get_replication_slot...** function.

Parameters:

- slot_name

  Indicates the name of the streaming replication slot.

  Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name.

Return types: int, text, and text.

Example:

```
gaussdb=# SELECT * FROM gs_logical_parallel_decode_status('replication_slot');
 stat_id |          stat_name           |        value
---------+------------------------------+---------------------
       1 | slot_name                    | replication_slot_lsn
       2 | reader_lsn                   | 0/1E249228
       3 | wal_read_total_time          | 4769541
       4 | wal_wait_total_time          | 4754879
       5 | parser_total_time            | 12727
       6 | decoder_total_time           | 215633
       7 | sender_total_time            | 41707
       8 | net_send_total_time          | 28435
       9 | net_wait_total_time          | 11918
      10 | net_send_total_bytes         | 931076
      11 | transaction_count            | 19
      12 | big_transaction_count        | 1
      13 | max_transaction_tuples       | 10000
      14 | sent_transaction_count       | 19
      15 | spill_disk_transaction_count | 0
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    16 | spill_disk_bytes             | 0
    17 | spill_disk_count             | 0
    18 | input_queue_full_count       | 226
    19 | output_queue_full_count      | 13
    20 | dml_count                    | 10000
    21 | dml_filtered_count           | 0
    22 | toast_count                  | 0
    23 | candidate_catalog_xmin       | 83576
    24 | candidate_xmin_lsn           | 0/1E0992A0
    25 | candidate_restart_valid      | 0/1E0992A0
    26 | candidate_restart_lsn        | 0/1E0992A0
    27 | candidate_xmin_csn           | 0
    28 | candidate_restart_csn        | 0
    29 | candidate_csn_min            | 17157
(29 rows)
```

Note: According to the definitions of statistical items, they must meet the following requirements:

wal_read_total_time >= wal_wait_total_time;

transaction_count >= big_transaction_count;

transaction_count >= sent_transaction_count;

transaction_count >= spill_disk_transaction_count;

dml_count >= dml_filtered_count;

dml_count >= toast_count;

If spill_transaction_count == 0, then spill_disk_bytes == 0;

However, frequent locking and unlocking are required, which greatly affects the performance. As a result, the preceding constraints may not be met in multi-thread cases.

transaction_count indicates the number of transactions in all databases.

sent_transaction_count indicates the number of transactions sent by the local database because transactions that are not in the local database will not be sent.

If the value of **slot_name** does not exist, the function does not report an error and the return value is empty.

- gs_logical_parallel_decode_reset_status('slot_name')

  Description: Resets indicators in **gs_logical_parallel_decode_status('slot_name')**.

  Parameters:

  – slot_name

    Indicates the name of the streaming replication slot.

    Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name.

  Return type: text.

  Example:

```
gaussdb=# SELECT * FROM gs_logical_parallel_decode_reset_status('replication_slot');
 gs_logical_parallel_decode_reset_status
-----------------------------------------
 OK
(1 row)

gaussdb=# SELECT * FROM gs_logical_parallel_decode_status('replication_slot');
 stat_id |          stat_name           |       value
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
---------+-------------------------------+----------------------
      1 | slot_name                     | replication_slot_lsn
      2 | reader_lsn                    | 0/1E2497F0
      3 | wal_read_total_time           | 0
      4 | wal_wait_total_time           | 0
      5 | parser_total_time             | 0
      6 | decoder_total_time            | 0
      7 | sender_total_time             | 0
      8 | net_send_total_time           | 0
      9 | net_wait_total_time           | 0
     10 | net_send_total_bytes          | 0
     11 | transaction_count             | 0
     12 | big_transaction_count         | 0
     13 | max_transaction_tuples        | 0
     14 | sent_transaction_count        | 0
     15 | spill_disk_transaction_count | 0
     16 | spill_disk_bytes              | 0
     17 | spill_disk_count              | 0
     18 | input_queue_full_count        | 0
     19 | output_queue_full_count       | 0
     20 | dml_count                     | 0
     21 | dml_filtered_count            | 0
     22 | toast_count                   | 0
     23 | candidate_catalog_xmin        | 83594
     24 | candidate_xmin_lsn            | 0/1E249528
     25 | candidate_restart_valid       | 0/1E249528
     26 | candidate_restart_lsn         | 0/1E249528
     27 | candidate_xmin_csn            | 0
     28 | candidate_restart_csn         | 0
     29 | candidate_csn_min             | 17176
(29 rows)
```

Note: The return value is the execution result. If the value of **slot_name** does not exist, the function does not report an error and the return value is **invalid slot name**.

Do not reset a replication slot that is being observed. The error information is as follows:

a. If **slot_name** is empty, the following error is reported: "ERROR: inputString should not be NULL".

b. If **slot_name** is not empty but does not exist, no error is reported but "invalid slot name" is displayed.

c. If **slot_name** is not empty but the replication slot corresponding to **slot_name** is being observed, no error is reported but "can't reset during observing! use gs_logical_decode_stop_observe to stop" is displayed.

- gs_logical_decode_start_observe('slot_name', window, interval)

  Description: Enables logical replication performance sampling. In a distributed environment, this function must be executed on DNs.

  Parameters:

  – slot_name

    Indicates the name of the streaming replication slot.

    Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name.

  – window

    Specifies the sampling window.

    Value range: integers from 2 to 1024. Sampled data in the latest *interval* × *window* is collected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - interval

    Specifies the performance monitoring interval, in seconds.

    Value range: 1s to 60s (time interval type). Sampled data in the latest *interval* × *window* is collected.

    Return type: text.

  Note: The return value is the execution result. If the value of **slot_name** does not exist, the function does not report an error and the return value is **invalid slot name**.

  Do not enable the observe function for a replication slot that is being observed. The error information is as follows:

  a. If **slot_name** is empty, the following error is reported: "ERROR: inputString should not be NULL".
  b. If **slot_name** is not empty but does not exist, no error is reported but "invalid slot name" is displayed.
  c. If the value of **window** is less than **2**, "window has to be >= 2" is displayed.
  d. If the value of **window** is greater than **1024**, "window has to be <= 1024" is displayed.
  e. If the value of **interval** is less than 1s, "sample interval has to be >= 1s" is displayed.
  f. If the value of **interval** is greater than 60s, "sample interval has to be <= 60s" is displayed.
  g. If **slot_name** is not empty but the observe function has been enabled for the replication slot corresponding to **slot_name**, no error is reported and the message "observe has started!" is displayed.

- gs_logical_decode_stop_observe('slot_name')

  Description: Stops logical replication performance sampling. In a distributed environment, this function must be executed on DNs.

  Parameters:

  - slot_name

    Indicates the name of the streaming replication slot.

    Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name.

  - Return type: text.

  Note: The return value is the execution result. If the value of **slot_name** does not exist, the function does not report an error and the return value is **invalid slot name**.

  Do not disable the observe function for a replication slot that is not observed. The error information is as follows:

  a. If **slot_name** is empty, the following error is reported: "ERROR: inputString should not be NULL".
  b. If **slot_name** is not empty but does not exist, no error is reported but "invalid slot name" is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

c. If **slot_name** is not empty but the observe function has been disabled for the replication slot corresponding to **slot_name**, no error is reported and the message "observe not started!" is displayed.

- gs_logical_decode_observe_data('slot_name')

  Description: Displays the original logical replication performance sampling data. In a distributed environment, this function must be executed on DNs.

  Table 1-138 Metrics

Metric	Description
slot_name	Name of the logical replication slot.
sample_time	Performance indicator sampling time, in microseconds.
reader_lsn	Location of the logical logs to be decoded.
wal_read_total_time	Time required for loading the log module, in microseconds.
wal_wait_total_time	Time required for waiting for the log module, in microseconds.
parser_total_time	Time required for the reader thread to process data, in microseconds.
decoder_total_time	Total processing duration of all decoder threads, in microseconds.
sender_total_time	Time required for the sender thread to process data, in microseconds.
net_send_total_time	Time required for the network to send logical logs, in microseconds.
net_send_total_bytes	Number of bytes of logical logs sent by the network.
transaction_count	Number of transactions.
big_transaction_count	Number of large transactions.
sent_transaction_count	Number of transactions sent by the local database.
spill_transaction_count	Number of flushed transactions.
spill_disk_bytes	Total number of bytes flushed to disk.

  Parameters:

  – slot_name

    Indicates the name of the streaming replication slot.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name.

– Return type: SETOF record

Example:

```
gaussdb=# SELECT * FROM gs_logical_decode_observe_data('replication_slot');
 slot_name |          sample_time          | reader_lsn | wal_read_total_time | wal_wait_total_time |
parser_total_time | decoder_total_time | sender_total_time | net_send_total_time |
net_send_total_bytes | transaction_count | big_
transaction_count | sent_transaction_count | spill_transaction_count | spill_disk_bytes
-----------+-------------------------------+------------+---------------------+---------------------
+------------------+-------------------+------------------+--------------------+---------------------
+------------------+-----
------------------+-----------------------+-------------------------+------------------
 repl      | 2023-01-12 20:09:40.416798+08 |   49447976 |           776846657 |           776846244
|               65 | {46,11,11,6}       |               56 |                0 |                   0 |
                0 |                     0 |                  0 |                0
 repl      | 2023-01-12 20:09:45.416849+08 |   49447976 |           776846657 |           776846244
|               65 | {46,11,11,6}       |               56 |                0 |                   0 |
                0 |                     0 |                  0 |                0
 repl      | 2023-01-12 20:09:50.417006+08 |   49447976 |           776846657 |           776846244
|               65 | {46,11,11,6}       |               56 |                0 |                   0 |
                0 |                     0 |                  0 |                0
 repl      | 2023-01-12 20:09:55.417057+08 |   49447976 |           776846657 |           776846244
|               65 | {46,11,11,6}       |               56 |                0 |                   0 |
                0 |                     0 |                  0 |                0
 repl      | 2023-01-12 20:10:00.417115+08 |   49447976 |           776846657 |           776846244
|               65 | {46,11,11,6}       |               56 |                0 |                   0 |
                0 |                     0 |                  0 |                0
 repl      | 2023-01-12 20:10:05.417165+08 |   49447976 |           776846657 |           776846244
|               65 | {46,11,11,6}       |               56 |                0 |                   0 |
                0 |                     0 |                  0 |                0
 repl      | 2023-01-12 20:10:10.417217+08 |   49447976 |           776846657 |           776846244
|               65 | {46,11,11,6}       |               56 |                0 |                   0 |
                0 |                     0 |                  0 |                0
 repl      | 2023-01-12 20:10:15.417271+08 |   49447976 |           776846657 |           776846244
|               65 | {46,11,11,6}       |               56 |                0 |                   0 |
                0 |                     0 |                  0 |                0
 repl      | 2023-01-12 20:10:20.417342+08 |   49448264 |           836085882 |           836085442
|               67 | {46,11,11,8}       |               58 |                0 |                   0 |
                0 |                     0 |                  0 |                0
 repl      | 2023-01-12 20:10:25.417433+08 |   49448264 |           836085882 |           836085442
|               67 | {46,11,11,8}       |               58 |                0 |                   0 |
                0 |                     0 |                  0 |                0
 repl      | 2023-01-12 20:10:30.417526+08 |   49448264 |           836085882 |           836085442
|               67 | {46,11,11,8}       |               58 |                0 |                   0 |
                0 |                     0 |                  0 |                0
 repl      | 2023-01-12 20:10:35.417532+08 |   49448264 |           836085882 |           836085442
|               67 | {46,11,11,8}       |               58 |                0 |                   0 |
                0 |                     0 |                  0 |                0
 repl      | 2023-01-12 20:10:40.417644+08 |   49448264 |           836085882 |           836085442
|               67 | {46,11,11,8}       |               58 |                0 |                   0 |
                0 |                     0 |                  0 |                0
 repl      | 2023-01-12 20:10:45.417763+08 |   49448264 |           836085882 |           836085442
|               67 | {46,11,11,8}       |               58 |                0 |                   0 |
                0 |                     0 |                  0 |                0
(14 rows)
```

Note: If the value of **slot_name** does not exist, the function does not report an error and the return value is an empty record.

- gs_logical_decode_observe('slot_name')

  Description: Displays logical replication performance data. In a distributed environment, this function must be executed on DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-139 Metrics

Metric	Description
slot_name	Name of the logical replication slot.
sample_time	Performance metric sampling time.
logical_decode_rate	Overall processing rate of logical decoding, in byte/s.
wal_read_rate	Processing rate in the WAL reading phase, in byte/s.
parser_rate	Processing rate in the WAL parsing phase, in byte/s.
decoder_rate	Processing rate in the decoding phase, in byte/s.
sender_rate	Processing rate in the sending phase, in byte/s.
net_send_rate	Network sending rate, in byte/s.

Parameters:

- slot_name

  Indicates the name of the streaming replication slot.

  Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name.
- Return type: SETOF record

Example:

```
gaussdb=# SELECT * FROM gs_logical_decode_observe('replication_slot');
    slot_name     |          sample_time          | logical_decode_rate | wal_read_rate | parser_rate  |
decoder_rate  |  sender_rate  | net_send_rate
------------------+-------------------------------+---------------------+---------------+--------------
+----------------+---------------+---------------
 replication_slot | 2023-01-12 20:16:50.42448+08  | 0.000               | 0.000         | 0.000        |
0.000          | 0.000         | 0.000
 replication_slot | 2023-01-12 20:16:55.424537+08 | 0.000               | 0.000         | 0.000        |
0.000          | 0.000         | 0.000
 replication_slot | 2023-01-12 20:17:00.424641+08 | 0.000               | 0.000         | 0.000        |
0.000          | 0.000         | 0.000
 replication_slot | 2023-01-12 20:17:05.424645+08 | 0.000               | 0.000         | 0.000        |
0.000          | 0.000         | 0.000
 replication_slot | 2023-01-12 20:17:10.424795+08 | 0.000               | 0.000         | 0.000        |
0.000          | 0.000         | 0.000
 replication_slot | 2023-01-12 20:17:15.424848+08 | 0.000               | 0.000         | 0.000        |
0.000          | 0.000         | 0.000
 replication_slot | 2023-01-12 20:17:20.424849+08 | 57.600              | 699029.126    | 96000000.000 |
1152000000.000 | 144000000.000 | 0.000
 replication_slot | 2023-01-12 20:17:25.424959+08 | 0.000               | 699029.126    | 96000000.000 |
1152000000.000 | 144000000.000 | 0.000
 replication_slot | 2023-01-12 20:17:30.42496+08  | 0.000               | 699029.126    | 96000000.000 |
1152000000.000 | 144000000.000 | 0.000
 replication_slot | 2023-01-12 20:17:35.425059+08 | 0.000               | 699029.126    | 96000000.000 |
1152000000.000 | 144000000.000 | 0.000
```

Note: If the value of **slot_name** does not exist, the function does not report an error and the return value is an empty record. If the denominator is 0, the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

latest collected valid data is returned. If the denominator is not 0 and the numerator is 0, **0** is returned.

Formulas:

- logical_decode_rate = (reader_lsn1 - reader_lsn2) / (sample_time1 - sample_time2)
- wal_read_rate = (reader_lsn1 - reader_lsn2) / (wal_read_total_time1 - wal_read_total_time2) - (wal_wait_total_time1 - wal_wait_total_time2)
- parser_rate = (reader_lsn1 - reader_lsn2) / (parser_total_time1 - parser_total_time2)
- decoder_rate = (reader_lsn1 - reader_lsn2) / avg(decoder_total_time1[i]-decoder_total_time2[i])
- sender_rate = (reader_lsn1 - reader_lsn2) / (sender_total_time1 - sender_total_time2) - (net_send_total_time1 - net_send_total_time2)
- net_send_rate = (net_sent_bytes1 - net_sent_bytes2) / (net_send_total_time1 - net_send_total_time2) - (net_wait_total_time1 - net_wait_total_time2)

- gs_logical_decode_observe_status('slot_name')

  Description: Queries the monitoring status of a specified logical decoding task. In a distributed environment, this function must be executed on DNs.

  Parameters:

  - slot_name

    Indicates the name of the streaming replication slot.

    Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name.

  - Return type: text.

  Example:

```
gaussdb=# SELECT * FROM gs_logical_decode_observe_status('replication_slot');
 gs_logical_decode_observe_status
----------------------------------
 START
(1 row)

gaussdb=# SELECT * FROM gs_logical_decode_observe_status('replication_slot');
 gs_logical_decode_observe_status
----------------------------------
 invalid slot name
(1 row)

gaussdb=# SELECT * FROM gs_logical_decode_stop_observe('replication_slot');
 gs_logical_decode_stop_observe
--------------------------------
 OK
(1 row)

gaussdb=# SELECT * FROM gs_logical_decode_observe_status('replication_slot');
 gs_logical_decode_observe_status
----------------------------------
 STOP
(1 row)
```

  Note: The return value is the monitoring status of the logical decoding task. If the value of **slot_name** does not exist, the function does not report an error and returns "invalid slot name".

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_get_parallel_decode_thread_info()

  Description: Executes on the DN where parallel decoding is performed and returns the thread information of parallel decoding.

  Return types: int64, text, text, and int.

  Example:

```
gaussdb=# SELECT * FROM gs_get_parallel_decode_thread_info();
    thread_id    | slot_name | thread_type | seq_number
-----------------+-----------+-------------+------------
 140335364699904 | slot1     | sender      |          1
 140335214098176 | slot1     | reader      |          1
 140335325312768 | slot1     | decoder     |          1
 140335291750144 | slot1     | decoder     |          2
 140335274968832 | slot1     | decoder     |          3
 140335258187520 | slot1     | decoder     |          4
 140335165404928 | slot2     | sender      |          1
 140335022864128 | slot2     | reader      |          1
 140335129818880 | slot2     | decoder     |          1
 140335113037568 | slot2     | decoder     |          2
(10 rows)
```

  Note: In the return values, **thread_id** indicates the thread ID, **slot_name** indicates the replication slot name, **thread_type** indicates the thread type (including the sender, reader and decoder), and **seq_number** indicates the sequence number of each thread with same type in the current replication slot. Each parallel decoding connection only has one sender and reader. Therefore, the sequence numbers of the sender and reader are both 1. The sequence numbers of the decoder are arranged from 1 to the decoding degree of parallelism (DOP) of the current replication slot.

- gs_get_logical_decode_parameter()

  Description: Queries the parameter options of active parallel logical decoding tasks.

  Table 1-140 Description

Column	Return Type	Description
slot_name	text	Replication slot name.
include_xids	bool	Specifies whether the decoding information contains xid.
include_timestamp	bool	Specifies whether decoded information contains the timestamp.
skip_empty_xacts	bool	Specifies whether to ignore empty transactions in the decoding result.
only_local	bool	Specifies whether to decode only the data generated in the current database.
include_username	bool	Specifies whether the BEGIN log of a transaction contains the username.
enable_ddl_decoding	bool	Specifies whether to enable the decoding of DDL statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Return Type	Description
enable_ddl_json_format	bool	Specifies whether to enable the JSON output format of DDL reverse parsing.
max_txn_in_memory	int64	Maximum memory occupied by transactions. (This parameter is deprecated and does not take effect.)
max_reorderbuffer_in_memory	int64	Maximum memory for the current decoding task.
decode_style	char	Decoding format.
parallel_decode_num	int64	Number of decoder threads for parallel decoding.
sending_batch	int64	Specifies whether to send messages in batches.
parallel_queue_size	int64	Specifies the length of the queue for interaction between parallel logical decoding threads.
enable_heartbeat	bool	Specifies whether to generate heartbeat logs
slice_id	int64	Specifies the shard ID of the replication table.
output_start_csn	int64	CSN position where the output starts.
output_start_lsn	int64	LSN position where the output starts.
logical_reader_bind_cpu	int64	ID of the CPU bound to the reader thread.
logical_decoder_bind_cpu_index	int64	Index of the CPU bound to the decoder thread.
auto_advance	bool	Specifies whether to automatically update.
skip_generated_columns	bool	Specifies whether to skip the result output of the generated column.
big_transaction_limit	int64	Memory usage limit for determining whether a transaction is a large transaction, in bytes.
exclUser	text	Lists the names of blacklisted users.
table_white_list	text	Whitelist table parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Return Type	Description
desc_memory_limit	int64	Total memory (in MB) for maintaining table metadata information by a logical decoding task.
max_logical_decode_memory	int64	Total memory (in GB) for parallel decoding tasks.
max_control_memory_size	int64	Upper limit (in GB) of the **max-logical-decode-memory** parameter.
max_queue_cache_num	int64	Upper limit of the **max-decode-to-sender-cache-num** parameter.
max_toast_memory_size	int64	Upper limit (in MB) of the memory used for reading the **toast** column in Xlogs.
restart_lsn	int64	A logical decoding control parameter for specifying the decoding start point.
timezone_is_utc	bool	Logical decoding control parameter, which is used to control the output of time type data with time zones.
enable_decode_position	bool	(Logical decoding control parameter) Specifies whether to send decoding position logs when the logs are read during logical decoding.
standby_connection	bool	(Configurable only for streaming decoding) Specifies whether to restrict decoding to standby nodes only.
sender_timeout	int64	Specifies the heartbeat timeout threshold (in milliseconds) between GaussDB and client. This option is valid only for streaming decoding.
change_log_max_len	int64	Specifies the maximum length of the logical log buffer, in bytes.
max_decode_to_sender_cache_num	int64	Specifies the threshold of the number of cached parallel decoding logs.
output_order	bool	Specifies whether to output decoding results based on CSNs. This option is valid only for streaming decoding.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT * FROM gs_get_logical_decode_parameter();
 slot_name |
configs
                  -----------
+-----------------------
```

NOTE

The constraints are as follows:

- The current network and hardware environment are normal.
- When making function calls, set **wal_level** to **logical**. The system can only parse the log files generated when **wal_level** is **logical**. If the Xlog file being used is not at the logical level, the parsed content will not have the expected values or types. There is no other impact. If **wal_level** is not set to **logical**, an error will be reported and the file will not be parsed.
- In this parsing mode, the system reads data from Xlog text records. So, it cannot parse data not found in Xlogs. If related Xlogs are missing, the missing logs cannot be parsed.
- If **enable_decode_parent_xid** is disabled, only the Xlogs related to the transaction are parsed from the start LSN. The parent transaction and child transactions are not parsed.
- If **enable_decode_parent_xid** is enabled, the parsing begins at the start LSN. The output contains only the child transaction details and subsequent transactions. The parent-child relationship can be established only after the child transaction is parsed. Therefore, information about the parent transaction before the child transaction and other child transactions at the same level is not output. If complete top-level transaction information is required, specify the transaction ID as the top-level transaction ID and set **start_lsn** properly.
- The name of a table cannot be obtained if its **relfilenode** has changed, for example, when the table is truncated or dropped.
- If a schema or table name has changed, you cannot get the name of its historical version.
- When a function is executed, if the output file size exceeds 100 MB, the file will be truncated and an error will be reported.
- If an XID is smaller than the minimum XID in the Xlog, no error is reported and the returned file is empty.
- If the retrieval range of **start_lsn** is too large or **start_lsn** is set to **NULL** (indicating that the traversal starts from the beginning), Xlog parsing may be interrupted because the underlying Xlogs have been recycled. In this case, you are advised to adjust the **start_lsn** range and try again.

Parameters:

– start_lsn

Specifies the start LSN for parsing.

Value range: character strings in the format of *xlogid*|*xrecoff*, for example, '1/2AAFC60'. If the value is **NULL**, the start position of the log to be parsed is not limited.

– upto_lsn

Specifies the end LSN for parsing.

Value range: character strings in the format of *xlogid*|*xrecoff*, for example, '1/2AAFC60'. If the value is **NULL**, the end position of the log to be parsed is not limited.

– xid

Transaction ID.

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> NOTE
>
> If the XID set by the user exceeds the maximum XID used by the current database system, the system displays a message indicating that the entered XID is invalid.

- **options**: specifies optional parameters, consisting of multiple pairs of **options_name** and **options_value**. You can retain the default values.
  - enable_decode_parent_xid

    Specifies whether to parse the parent and child transactions of a specified XID.

    Value type: Boolean.

    Value range:

    - **false**: The table information of a specified transaction is parsed.
    - **true**: The parent and related child transactions of a specified transaction are parsed.

    Default value: **false**.

    > NOTE
    >
    > Only Xlogs generated in versions later than 505.2.0 support the construction of parent-child transaction relationships.

Return type: text.

Example:

```
gaussdb=# SELECT * FROM gs_logical_get_transaction_table_info('0/202FC320', '0/20306EE8',
'221557', 'enable_decode_parent_xid', '1');
                                    output_filepath
-------------------------------------------------------------------------------------------
 /home/omm/install/clusters/clusters_cent/cxz/cxz_datanode/221557_202fc320_20306ee8.log
(1 row)
Parsing result:
=== DETAIL RECORDS ===
XID|TOP_XID|PARENT_XID|RELFILE_OID|START_LSN|END_LSN|CSN|TABLE_OID|SCHEMA_NAME|
TABLE_NAME|OPERATION|ADDITIONAL_INFO
9982031|9982031|0|324521|9/297251C8|9/29725260|9417216|324521|public|t_tran|INSERT|
9982032|9982031|9982031|324521|9/297254E8|9/29725578|9417216|324521|public|t_tran|INSERT|
9982034|9982031|9982032|324521|9/29725B80|9/29725C18|9417217|324521|public|t_tran|INSERT|
9982041|9982031|9982031|324521|9/29727940|9/297279D0|9417221|324521|public|t_tran|INSERT|
9982045|9982031|9982041|324521|9/2974C458|9/2974C4F0|9417224|324521|public|t_tran|INSERT|
9982047|9982031|9982041|324521|9/2974CC00|9/2974CC98|9417225|324521|public|t_tran|INSERT|
=== SUMMARY RECORDS ===
XID|TABLE_OID|SCHEMA_NAME|TABLE_NAME|IS_COMMITTED|INSERT_COUNT|UPDATE_COUNT|
DELETE_COUNT|DDL_COUNT
9982031|324521|public|t_tran|true|6|0|0|0
```

Table 1-141 Metrics

Metric	Description
XID	XID.
TOP_XID	Top-level XID.
PARENT_XID	Parent XID.
RELFILE_NODE	Table file object identifier.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Metric	Description
START_LSN	Start LSN of a transaction.
END_LSN	End LSN of a transaction.
CSN	Commit sequence number.
TABLE_OID	Table object identifier.
SCHEMA_NAME	Schema to which the table belongs.
TABLE_NAME	Table name.
OPERATION	Operation type.
ADDITIONAL_INFO	DDL information.
IS_COMMITTED	Specifies whether a transaction is committed.
INSERT_COUNT	Number of inserted records.
UPDATE_COUNT	Number of updated records.
DELETE_COUNT	Number of deleted records.
DDL_COUNT	Number of DDL statements.

- gs_get_distribute_decode_status()

  Description: Obtains the distributed decoding status details (by replication slot) on the current node. This command must be executed on CNs. If it is executed on DNs, null is returned.

  Return types: text, int, int, int64, xid, xid, text, text, and text.

```
gaussdb=# SELECT * FROM gs_get_distribute_decode_status();
 slot_name | logical_receiver_num | slice_num | walsender_thread_id | last_sent_csn |
last_confirmed_csn |       receiver_queue_length       |          connect_times          |
csn_receive_array
-----------+----------------------+-----------+---------------------+---------------+--------------------
+---------------------------------+---------------------------------
+-----------------------------------------------------
 slot1     |                    1 |         3 |     139958481843968 |       2012169 |             2010107 | queue0: 1,
queue1: 1, queue2: 0 | slice0: 2, slice1: 2, slice2: 2 | slice0: 2012244, slice1: 2012244, slice2: 2012244
(1 row)
```

Table 1-142 Metrics

Metric	Type	Description
slot_name	text	Replication slot name.
logical_receiver_num	int	Number of receiver threads started for distributed decoding.
slice_num	int	Number of shards in a cluster.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Metric	Type	Description
walsender_thread_id	int64	WAL sender thread ID.
last_sent_csn	xid	CSN that is sent recently.
last_confirmed_csn	xid	The last CSN returned from the client that has been confirmed for reception.
receiver_queue_length	text	Length of the log queue received on each DN (displayed in the character string format after concatenation).
connect_times	text	Number of times that each DN is connected (displayed in the character string format after concatenation).
csn_receive_array	text	Latest CSN obtained by each DN (displayed in the character string format after concatenation).

- gs_logical_decode_meta_cache_summary('slot_name')

  Description: Obtains statistics about the decoding metadata cache of active replication slots in parallel logical decoding, including global metrics such as the number of cached objects, number of versions, total memory usage, cache hit ratio, and system catalog scan hit ratio.

  Parameters:

  – slot_name

    Indicates the name of the streaming replication slot.

    Value type: character string.

    Value range: a string consisting of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), or periods (.). One or two periods alone cannot be used as the replication slot name.

Table 1-143 Metrics

Metric	Description
type	Type of the logical decoding metadata. The value can be **table**, **partition**, **type**, **enum**, **auth**, or **database**.
entry_num	Number of metadata entries.
version_num	Total number of metadata entry versions (each entry may have multiple versions).
total_mem_size	Total memory of the metadata cache, in bytes.
hit_count	Number of cache hits.
miss_count	Number of cache misses.
hit_ratio	Cache hit ratio.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Metric	Description
scan_count	Number of times system catalogs are scanned.
scan_hit_count	Number of times system catalogs are hit.
scan_hit_ratio	Hit ratio of system catalog scans.
extra_info	Other information.

Example:

```
gaussdb=# SELECT * FROM gs_logical_decode_meta_cache_summary('slot1');
   type    | entry_num | version_num | total_mem_size | hit_count | miss_count | hit_ratio | scan_count |
scan_hit_count | scan_hit_ratio | extra_info
-----------+-----------+-------------+----------------+-----------+------------+-----------+------------
+----------------+----------------+------------
 table     |        53 |          53 |         173056 |         0 |          0 |         0 |          0 |
0 |
 partition |         0 |           0 |              0 |         0 |          0 |         0 |          0 |
 type      |        14 |          14 |           1791 |         0 |          0 |         0 |          0 |
|
 enum      |         0 |           0 |              0 |         0 |          0 |         0 |          0 |
 auth      |         1 |           1 |            136 |         0 |          0 |         0 |          0 |
 database  |         0 |           0 |              0 |         0 |          0 |         0 |          0 |
(6 rows)
```

- gs_logical_decode_meta_cache_stats('slot_name', 'type', 'mode', 'limit_num')

  Description: Obtains statistics about top *N* objects in the decoding metadata cache of active replication slots in parallel logical decoding. The statistics can be queried by memory usage (**memory**), version chain length (**version_cnt**), number of misses (**miss_count**), or hit ratio (**hit_ratio**).

  Parameters:

  – slot_name

    Indicates the name of the streaming replication slot.

    Value type: character string.

    Value range: a string consisting of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), or periods (.). One or two periods alone cannot be used as the replication slot name.

  – type

    Type name of the metadata.

    Value range: **table_oid**, **table_rfn**, **partition**, **type**, **enum**, **auth** and **database**.

  – mode

    Query dimension.

    Value range: **memory**, **version_cnt**, **miss_count**, and **hit_ratio**.

  – limit_num

    Maximum number of output rows.

    Value type: integer.

    Value range: 1 to 1000.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-144 Metrics

Metric	Description
oid	OID of a metadata entry.
name	Name of a metadata entry.
relfilenode	**relfilenode** of a metadata entry.
version_cnt	Number of versions of a metadata entry.
invalid_version_cnt	Number of invalid versions of a metadata entry.
total_mem_size	Total memory occupied by a metadata entry, in bytes.
avg_version_size	Average memory occupied by each version of a metadata entry, in bytes.
hit_count	Number of times a metadata entry is hit.
miss_count	Number of times a metadata entry is not hit.
hit_ratio	Hit ratio of a metadata entry.
create_time	Time when a metadata entry is loaded to the memory.
evict_fail_count	Number of times a metadata entry fails to be evicted.
evict_count	Number of times a metadata entry is evicted.
latest_evict_time	Time when a metadata entry was last evicted.
warning_flags	Exception flag information of a metadata entry.
extra_info	Other information.

Example:

```
gaussdb=# SELECT * FROM gs_logical_decode_meta_cache_stats('slot1', 'table_oid', 'memory', 20);
  oid  |                 name                 | relfilenode | version_cnt | invalid_version_cnt | total_mem_size |
avg_version_size | hit_count | miss_count | hit_ratio |          create_time          | evict_fail_count |
evict_count | latest_evict_time | warning_flags | extra_info
-------+--------------------------------------+-------------+-------------+---------------------+----------------
+------------------+-----------+------------+-----------+-------------------------------+------------------
+-------------+-------------------+---------------+------------
 16609 | snap_summary_statement               |       16609 |           1 |                   0 |           6544
|            6544 |         3 |          0 |         1 | 2026-07-01 10:55:57.313943+08 |                0 |           0
|                 |           0 |
 16616 | snap_global_statement_count          |       16616 |           1 |                   0 |           3424
|            3424 |         3 |          0 |         1 | 2026-07-01 10:55:57.31703+08  |                0 |           0
|                 |           0 |
 16622 | snap_summary_statement_count         |       16622 |           1 |                   0 |           3424
|            3424 |         3 |          0 |         1 | 2026-07-01 10:55:57.319207+08 |                0 |           0
|                 |           0 |
 16758 | snap_global_stat_all_tables          |       16758 |           1 |                   0 |           2904
|            2904 |         3 |          0 |         1 | 2026-07-01 10:55:54.406154+08 |                0 |           0
|                 |           0 |
 16664 | snap_global_redo_status              |       16664 |           1 |                   0 |           2800
|            2800 |         0 |          0 |         0 | 2026-07-01 10:55:54.385334+08 |                0 |           0
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
|                 |           0 |
 16764 | snap_summary_stat_all_tables          |       16764 |           1 |                 0 |           2800
|            2800 |           3 |           0 |           1 | 2026-07-01 10:55:54.408911+08 |                 0 |           0
|                 |           0 |
(6 rows)
```

NOTICE

During function execution, you need to access the hash table that caches the decoding metadata and obtain a lock to ensure consistent statistics. During the execution, lock contention with the logical decoding thread may occur, affecting the decoding performance. You are advised not to frequently call this function. It is recommended that this function be used in scenarios such as fault locating and performance analysis.

- gs_logical_decode_meta_cache_obj_detail('slot_name', 'type', oid, 'mode', 'limit_num')

  Description: Obtains detailed statistics and version chain details about a specified object in the decoding metadata cache of active replication slots in parallel logical decoding, including the number of versions, valid range (**csnmin**/**csnmax**), memory usage, and hit ratio.

  Parameters:

  – slot_name

    Indicates the name of the streaming replication slot.

    Value type: character string.

    Value range: a string consisting of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), or periods (.). One or two periods alone cannot be used as the replication slot name.

  – type

    Type name of the metadata.

    Value range: **table**, **partition**, **type**, **enum**, **auth** and **database**.

  – oid

    OID of the metadata to be queried.

  – mode

    Output mode of the query result.

    Value range: **memory** and **file**.

  – limit_num

    Maximum number of output rows.

    Value type: integer.

    Value range: When **mode** is set to **memory**, the value ranges from 1 to 1000. When **mode** is set to **file**, the value ranges from 1 to 2147483647.

Table 1-145 Metrics

Metric	Description
oid	OID of a metadata version.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Metric	Description
relfilenode	**relfilenode** of a metadata version.
name	Name of a metadata version.
csnmin	**csnmin** information of a metadata version.
csnmax	**csnmax** information of a metadata version.
is_invalid	Specifies whether a metadata version is valid.
mem_size	Memory usage of a metadata version, in bytes.
hit_count	Number of times a metadata version is hit.
refcnt	Reference count of a metadata version.
createtime	Time when a metadata version is loaded to the memory.
extra_info	Other information. When **mode** is set to **file**, it indicates the output file path.

Example:

```
gaussdb=# SELECT * FROM gs_logical_decode_meta_cache_obj_detail('slot1', 'table', 54921, 'memory',
100);
  oid   | relfilenode | name | csnmin | csnmax | is_invalid | mem_size | hit_count | refcnt |
createtime           | extra_info
-------+-------------+------+--------+--------+------------+----------+-----------+--------
+-------------------------------+------------
 54921 |       54921 | t1   | 409737 |      0 | f          |      408 |         0 |      0 | 2026-07-01
10:55:52.27585+08 |
(1 row)
gaussdb=# SELECT * FROM gs_logical_decode_meta_cache_obj_detail('slot1', 'table', 54921, 'file', 100);
 oid | relfilenode | name | csnmin | csnmax | is_invalid | mem_size | hit_count | refcnt | createtime
|                                 extra_info
-----+-------------+------+--------+--------+------------+----------+-----------+--------+------------
+-----------------------------------------------------------------------------------------------
   0 |           0 |      |      0 |      0 | f          |        0 |         0 |      0 |            | <DataDir>/
meta_cache_slot1_1782874593.out
(1 row)
```

NOTICE

During function execution, you need to access the hash table that caches the decoding metadata and obtain a lock to ensure consistent statistics. During the execution, lock contention with the logical decoding thread may occur, affecting the decoding performance. You are advised not to frequently call this function. It is recommended that this function be used in scenarios such as fault locating and performance analysis.

- gs_get_distribute_decode_status_detail()

  Description: Obtains the distributed decoding status details (by DN) on the current node. This command must be executed on CNs. If it is executed on DNs, null is returned.

  Return types: text, int, int64, int, int, and xid.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT * FROM gs_get_distribute_decode_status_detail();
 slot_name | slice_id |     thread_id     | queue_len | connect_times | received_csn
-----------+----------+-------------------+-----------+---------------+--------------
 slot1     |        0 | 139959895848704 |         1 |             2 |      2012244
 slot1     |        1 | 139959895848704 |         1 |             2 |      2012244
 slot1     |        2 | 139959895848704 |         0 |             2 |      2012244
(3 rows)
```

Table 1-146 Metrics

Metric	Type	Description
slot_name	text	Replication slot name.
slice_id	int	Shard ID (starting from 0).
thread_id	int64	ID of the receiver thread started for distributed decoding.
queue_len	int	Length of the log queue received on the current DN.
connect_times	int	Number of times that each DN is connected.
received_csn	xid	Latest CSN obtained from the current DN.

- gs_logical_dictionary_baseline()

  Description: Baselines the data dictionary data for logical decoding. If the operation is successful, the time required is returned. If the operation fails, the failure cause is returned.

  Return type: text.

- gs_logical_dictionary_disabled()

  Description: Disables the logical decoding data dictionary function and stops decoding incremental DDL statements. If the operation is successful, "OK" is returned. If the operation fails, the failure cause is returned.

  Return type: text.

⚠ CAUTION

If the data dictionary functionality is disabled, the replication slot cannot utilize the existing data dictionary for decoding. To enable this functionality, execute the gs_logical_dictionary_baseline() function to re-establish the logical data dictionary baseline and rebuild the replication slot.

- gs_logical_decode_lock()

  Description: Specifies the lock placing API for upgrading and managing the logical decoding component. If the lock is placed, **t** is returned. If the lock fails to be placed, **f** is returned. This function is for internal use only. You are advised not to use it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter: **wait_time** specifies the lock waiting time. Value range: a non-negative number, in seconds. If no parameter is input, the lock waiting time is 0 seconds.

Return type: Boolean.

- gs_logical_decode_unlock()

  Description: Specifies the lock releasing API for upgrading and managing the logical decoding component. There is no input parameter. If the lock is released, **t** is returned. If the lock fails to be released, **f** is returned. This function is for internal use only. You are advised not to use it.

  Return type: Boolean.

- gs_add_logical_decoding_position_xlog()

  Description: After this function is executed, Xlogs of the XLOG_STANDBY_DECODE_POSITION type are generated. The logs will be read by a decoding task in the current database to feed back the decoding position. Only users with the SYSADMIN, replication, or OPRADMIN permission can call this function; otherwise, the call fails. The execution frequency of this function is calculated based on the database statistics. The frequency of calling this function by each database is controlled by node. The minimum calling interval is 1 second. This principle also applies to distributed scenarios. Each CN controls the calling frequency separately.

  Return type: text

- gs_change_replication_slot_plugin('slot_name', 'plugin_name')

  Description: Modifies the plug-in type of a logical replication slot.

  Parameters:

  – slot_name

    Indicates the name of the streaming replication slot.

    Value range: character strings, which can only consist of lowercase letters, digits, underscores (_), question marks (?), hyphens (-), and periods (.). One or two periods alone cannot be used as the replication slot name.

  – plugin_name

    Indicates the name of the plug-in.

    Value range: a string. Currently, **mppdb_decoding**, **sql_decoding**, **parallel_binary_decoding**, **parallel_json_decoding**, and **parallel_text_decoding** are supported.

    Return type: Boolean. The default value is **true**.

  Example:

  Query information about the current replication slot.

```
gaussdb=# SELECT * FROM pg_get_replication_slots();
 slot_name |          plugin          | slot_type | datoid | active | xmin | catalog_xmin | restart_lsn |
dummy_standby | confirmed_flush | confirmed_csn | dictionary_csn_min | slot_dictionary_type
-----------+--------------------------+-----------+--------+--------+------+--------------+-------------
+---------------+-----------------+---------------+--------------------+----------------------
 slot1     | parallel_json_decoding   | logical   |  12834 | f      |      |              | 0/125A15E0  | f             |
0/125E84D0      |               |              7787 | dictionary table
 slot2     | parallel_binary_decoding | logical   |  12834 | f      |      |              | 0/E9C71B0   | f             |
0/E9CA638       |               |              3860 | dictionary table
 slot3     | parallel_text_decoding   | logical   |  12834 | f      |      |              | 0/E9D96D0   | f             | 0/
EEE6508       |               |              3867 | dictionary table
 slot4     | mppdb_decoding           | logical   |  12834 | f      |      |              | 0/E9AFC38   | f             | 0/
EEE6508       |               |              3853 | dictionary table
(4 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Change the replication slot type.

```
gaussdb=# SELECT * FROM gs_change_replication_slot_plugin('slot1', 'mppdb_decoding');
 gs_change_replication_slot_plugin
-----------------------------------
 t
(1 row)
```

Query the replication slot information again.

```
gaussdb=# SELECT * FROM pg_get_replication_slots();
 slot_name |          plugin          | slot_type | datoid | active | xmin | catalog_xmin | restart_lsn |
dummy_standby | confirmed_flush | confirmed_csn | dictionary_csn_min | slot_dictionary_type
-----------+--------------------------+-----------+--------+--------+------+--------------+-------------
+---------------+-----------------+---------------+--------------------+----------------------
 slot1     | mppdb_decoding           | logical   |  12834 | f      |      |              | 0/125A15E0  | f             |
0/125E84D0      |               |              7787 | dictionary table
 slot2     | parallel_binary_decoding | logical   |  12834 | f      |      |              | 0/E9C71B0   | f             |
0/E9CA638       |               |              3860 | dictionary table
 slot3     | parallel_text_decoding   | logical   |  12834 | f      |      |              | 0/E9D96D0   | f             | 0/
EEE6508       |               |              3867 | dictionary table
 slot4     | mppdb_decoding           | logical   |  12834 | f      |      |              | 0/E9AFC38   | f             | 0/
EEE6508       |               |              3853 | dictionary table
(4 rows)
```

- gs_get_inter_cluster_version_info()

  Description: Queries the version number of primary and standby databases, the version of the logical decoding component, and the replication type (logical replication or physical replication). It is not supported in the current version.

  Return type: int, text, int, text, text

1.6.27.11 Segment-Page Storage Functions

NOTE

To create a segment-page table on a storage engine, you need to set **segment** to **on** when creating the table. Example:

```
CREATE TABLE t1(id int) WITH (segment=on, storage_type=astore);
CREATE TABLE t1(id int) WITH (segment=on, storage_type=ustore);
```

- The values of the fields related to the segment-page storage function are described as follows:

  a. **forknum**: data file fork.

     Value range: **0** (mainfork), **1** (fsmfork), and **2** (vm fork).

  b. **file id**: data file ID.

     Value range: **1** (metadata file) and **2** to **5** (data file).

  c. **blocks**: extent size.

     Value range: **1** (No.1 file), **8** (No.2 file), **128** (No.3 file), **1024** (No.4 file), and **4096** (No.5 file).

  d. **file_block_id**/**head_block_id**/**block_id**: offset page number of the physical page in the data file and other fields that indicate the page number.

     Value range: [0,4294967294].

  e. **page_ type**: page type.

     Value range:

     Meta pages: **file head**/**file_header**: file header; **spc head**/**spc_header**: space header; **map head**/**map_header**: mapping header; **map page**/

map_pages: mapping page; **reverse pointer page**/**inverse pointer page**/**ip pages**: inverse pointer page; **segment head page**/**segment head**: segment header page; **level1 page**: level-1 page; **data_pages**/**data extent**: data page; **fork head**: fork header.

Data pages: **heap**, **uheap**, **btree**, and **ubtree**.

Unknown pages: **unknown(data extent)**: all-zero segment page whose type cannot be determined; **unknown(fsm indexurq)**: fsm or indexurq page.

f. **contents**: storage content of a data file.

Value range: **permanent** (permanent), **unlogged** (no log), **temporary** (global temporary), and **temporary2** (local temporary)

- local_space_shrink(tablespacename TEXT, databasename TEXT)

  Description: Shrinks specified physical segment-page space on the current node. Only the currently connected database can be shrank.

  Return value: empty.

- gs_space_shrink(tablespace int4, database int4, extent_type int4, forknum int4)

  Description: Shrinks the physical size of the designated segment-page space, similar to **local_space_shrink**. However, it has different parameters, requiring the tablespace and database OIDs as inputs and specifying that **extent_type** must be an integer between 2 and 5. The value **1** of **extent_type** indicates segment-page metadata. Currently, the physical file that contains the metadata cannot be shrunk. This function is used only by tools. You are advised not to use it directly.

  Return value: empty.

- global_space_shrink(tablespacename TEXT, databasename TEXT)

  Description: Compresses segment-page storage space on all DNs in a cluster. This is performed on a CN.

  Note: If **global_space_shrink** locks the cluster, DDL operations cannot be performed. **local_space_shrink** does not lock the cluster.

- gs_seg_dump_page(tablespace_name name, file_id int4, bucketnode int4, file_block_id bigint, forknum int4 default 0)

  Description: Parses a specified page in segment-page mode and returns the parsed content. Only users with the SYSADMIN attribute and users with the OPRADMIN attribute in O&M mode can execute this function. This function is used to parse physical pages. The parsing result of one page is returned each time. The returned result does not contain the actual user data. This function does not require users to enter the page type. During implementation, the system attempts to determine the page type. If the page type cannot be determined, the system outputs possible parsing results.

  The following describes the parameters.

Name	Type	Description
table_space_name	NAME	Tablespace to which a segment object belongs. Value range: existing valid tablespace names.
file_id	INTEGER	Data file ID. Value range: [1,5] of the int4 type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
bucketnode	INTEGER	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
file_block_id	BIGINT	Offset page number of the physical page in the data file. The value range is [0,4294967294].
forknum	INTEGER DEFAULT 0	Fork number of a data file. The default value is **0**. Value range: • **0**: main fork. • **1**: FSM fork. • **2**: VM fork.

The following describes the return types.

Name	Type	Description
page_type	TEXT	Page type. Value range: • Service pages: **heap**, **uheap**, **btree**, **ubtree**, **gsivfflat_index**, and **gsdiskann_index** • Segment-page metapages: **bucket main head**, **file head**, **spc head**, **map head**, **map page**, **reverse pointer page**, **segment head page**, and **level1 page**. • Unknown pages: – **unknown(data extent)**: all-zero segment page whose type cannot be determined. – **unknown(fsm indexurq)**: fsm or indexurq page.
result	TEXT	Parsing result.

For example, perform the following operations after creating a segment-page ordinary table:

```
gaussdb=# SELECT * FROM gs_seg_dump_page('pg_default', 1, 1024, 4157);
     page_type     |                                result
-------------------+-----------------------------------------------------------------------------
 segment head page | Page information of block 4157/4157                                         +
                   |         pd_lsn: 0/2C90608 ,len 8 ,offset 0                                  +
                   |                                                                             +
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
|       pd_checksum: 0x8A7F, verify success,len 2, offset 8               +
|       pd_flags:                                                        +
|       pd_lower: 24, empty, len 2, offset 12                            +
|       pd_upper: 8192, old, len 2, offset 14                            +
|       pd_special: 8192, size 0, len 2, offset 16                       +
|       Page size & version: 8192, 8, len 2, offset 18                   +
|       pd_xid_base: 0, len 8, offset 24 pd_multi_base: 0, len 8, offset 32+
|       pd_prune_xid: 0, len 4 ,offset 20                                +
|                                                                        +
|       Segment head information on this page                            +
|           magic: 44414548544E454D                                      +
|           lsn is: 0/2C90540                                            +
|           nblocks: 0                                                   +
|           total_blocks: 8                                              +
|           reserved: 0                                                  +
|           Level 0 slots information on this page                       +
|               The BlockNumber of level0 slots 0 is: 4157               +
|           fork head information on this page                           +
|               4157(valid)                                              +
|               4294967295(invalid)                                      +
|               4294967295(invalid)
(1 row)
```

- gs_seg_dump_page(relid oid, bucketnode int, block_id bigint, partition bool default false, forknum int4 default 0)

  Description: Parses a specified page in segment-page mode and returns the parsed content. This function is executed on DNs. Only users with the SYSADMIN attribute and users with the OPRADMIN attribute in O&M mode can execute this function. This function is used to parse logical pages. The parsing result of one page is returned each time. The returned result does not contain the actual user data. This function does not require users to enter the page type. During implementation, the system attempts to determine the page type. If the page type cannot be determined, the system outputs possible parsing results.

  The following describes the parameters.

Name	Type	Description
relid	OID	Segment object ID. Value range: existing valid segment-page OIDs. Not meeting this requirement will result in an error.
bucketnode	INTEGER	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
block_id	BIGINT	Logical page number. The value range is [0,4294967294].

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
partition	BOOLEAN DEFAULT FALSE	Specifies whether the segment object is a partition. The default value is **false**. Value range: Boolean values, indicating whether the object corresponding to the input OID is a partition.
forknum	INTEGER DEFAULT 0	Fork number of a data file. The default value is **0**. Value range: ● **0**: main fork. ● **1**: FSM fork. ● **2**: VM fork.

The following describes the return values.

Name	Type	Description
page_type	TEXT	Page type. Value range: ● Service pages: **heap**, **uheap**, **btree**, and **ubtree**. ● Segment-page metapages: **bucket_main_head**, **file head**, **spc head**, **map head**, **map page**, **reverse pointer page**, **segment head page**, and **level1 page**. ● Unknown pages: – **unknown(data extent)**: all-zero segment page whose type cannot be determined. – **unknown(fsm indexurq)**: fsm or indexurq page.
result	TEXT	Parsing result.

For example, perform the following operations on a DN, and ensure that **relid** must be a valid OID of the segment-page object on the DN, and the object must have data so that the page information can be queried:

```
gaussdb=# SELECT * FROM gs_seg_dump_page(16788, 1024, 0);
 page_type |                                                 result
-----------
+-----------------------------------------------------------------------------------------------------------------------------
----
 heap      | Page information of block
6021/6021                                                                                                                  +
           |        pd_lsn: 0/4463418 ,len 8 ,offset 0                                                                          +
           |                                                                                                                 +
           |        pd_checksum: 0xD4CD, verify success,len 2, offset
8                                                                                                                          +
           |        pd_flags:                                                                                                +
           |        pd_lower: 44, non-empty, len 2, offset
12                                                                                                                         +
           |        pd_upper: 8160, old, len 2, offset 14                                                                       +
           |        pd_special: 8192, size 0, len 2, offset 16                                                                  +
           |        Page size & version: 8192, 6, len 2, offset 18
+
```

```
        |       pd_xid_base: 17049, len 8, offset 24 pd_multi_base: 0, len 8, offset
32                                  +
        |       pd_prune_xid: 17049, len 4 ,offset 20                                                  +
        |                                                                                              +
        |       Heap tuple information on this page
+
        |                                                                                              +
        |               Tuple #1 is normal: length 28, offset
8160                                                +
        |                       (uint64)xmin/xmax/t_cid:
17052/0/0                                             +
        |                       (uint32)t_xmin/t_xmax: 3/3(check ilm flag to indicate whether t_xmin/
t_xmax is xid or ilm time)+
        |                       ctid:(block 0/0, offset 1)                                                +
        |                       t_infomask: HEAP_XMAX_INVALID
HEAP_HAS_NO_UID                                       +
        |                       t_infomask2: Attrs Num: 1                                                    +
        |                       t_hoff: 24                                                                +
        |                       t_bits: NNNNNNNN                                                             +
        |       Summary (1 total):  1 normal, 0 unused, 0
dead                                                  +
        |                                                                                              +
        | Normal Heap Page, special space is 0
+
        |                                                                                              +
        |
(1 row)
```

- gs_seg_get_spc_location(tablespace_name NAME, bucketnode INTEGER, head_block_id BIGINT, block_id BIGINT)

  Description: Calculates the physical location given a segment and logical page number. Only an administrator can query the information.

  The following describes the parameters.

Name	Type	Description
tablespace_name	NAME	Tablespace to which a segment object belongs.
bucketnode	INTEGER	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
head_block_id	BIGINT	Page number of a segment header.
block_id	BIGINT	Logical page number.

  The following describes the return values.

Name	Type	Description
extent_id	INTEGER	Logical extent number of the logical page.
extent_size	INTEGER	Size of the logical extent where the logical page is located.
file_id	INTEGER	ID of the data file where the physical page is located. Value range: [1,5] of the int4 type.
file_block_id	BIGINT	Offset page number of the physical page in the data file.

For example, perform the following operations on the DN after creating a segment-page ordinary table in the tablespace and inserting data:

```
gaussdb=# SELECT * FROM gs_seg_get_spc_location('pg_default', 1024, 4157, 0);
 extent_id | extent_size | file_id | file_block_id
-----------+-------------+---------+---------------
         0 |           8 |       2 |          4157
(1 row)
```

- gs_seg_get_spc_location(relid OID, bucketnode INTEGER, block_id BIGINT, partition BOOLEAN DEFAULT FALSE, forknum INTEGER DEFAULT 0)

  Description: Calculates the physical location given a segment and logical page number. Only an administrator can query the information.

  The following describes the parameters.

Name	Type	Description
relid	OID	Segment object ID.
bucketnode	INTEGER	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
block_id	BIGINT	Logical page number.
partition	BOOLEAN DEFAULT FALSE	Specifies whether the segment object is a partition. The default value is **FALSE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
forknum	INTEGER DEFAULT 0	Fork of a segment object. The default value is **0**. Value range: • **0**: main fork. • **1**: FSM fork. • **2**: VM fork.

The following describes the return values.

Name	Type	Description
extent_id	INTEGER	Logical extent number of the logical page.
extent_size	INTEGER	Size of the logical extent where the logical page is located.
file_id	INTEGER	ID of the data file where the physical page is located. Value range: [1,5] of the int4 type.
file_block_id	BIGINT	Offset page number of the physical page in the data file.

For example, perform the following operations on a DN, and ensure that **relid** must be a valid OID of the segment-page object on the DN, and the object must have data:

```
gaussdb=# SELECT * FROM gs_seg_get_spc_location(24578,1024, 0);
 extent_id | extent_size | file_id | file_block_id
-----------+-------------+---------+---------------
         0 |           8 |       2 |          4157
(1 row)
```

- gs_seg_get_location(block_id BIGINT, as_extent BOOLEAN DEFAULT FALSE)

  Description: Calculates the physical location given a segment and logical page number. Only an administrator can query the information.

  The following describes the parameters.

Name	Type	Description
block_id	BIGINT	Logical page number.
as_extent	BOOLEAN DEFAULT FALSE	Specifies whether the **block_id** parameter is an extent number. The default value is **FALSE**.

The following describes the return values.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
extent_id	BIGINT	Extent number.
extent_size	INTEGER	Extent size.
extent_offset	INTEGER	Extent offset block number.
level0_slots_idx	INTEGER	Index of the extent in the level-0 slot array in the segment header.
level1_slots_idx	INTEGER	Index of the extent in the level-1 slot array in the segment header.
level1_page_offset	INTEGER	Offset of the extent on the level-1 slot page in the segment header.
segment_blocks	BIGINT	Number of pages containing this page or extended segments.
relative_fno	INTEGER	Number of the relative file that contains this page or extent.

Example:

```
gaussdb=# SELECT * FROM gs_seg_get_location(4157);
 extent_id | extent_size | extent_offset | level0_slots_idx | level1_slots_idx | level1_page_offset |
segment_blocks | relative_fno
-----------+-------------+---------------+------------------+------------------+--------------------
+----------------+--------------
        47 |         128 |            61 |               47 |                  |                  |          4158 |              3
(1 row)
```

- gs_seg_get_segment_layout()

  Description: Outputs the static segment layout. Only an administrator can query the information.

  The following describes the return values.

Name	Type	Description
version	TEXT	Segment-page version. The default value is **1.0**.
section_id	INTEGER	ID of the data section divided by a segment.
section_type	TEXT	Extent type of the segment data section. Value range: • **meta**: segment header. • **data**: data.
extent_size	INTEGER	Extent size. The unit is byte.
extent_page_count	INTEGER	Number of extent pages.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
extent_count_start	BIGINT	Start extent number.
extent_count_end	BIGINT	End extent number.
total_size	BIGINT	Size of the segment data section. The unit is byte.

Example:

```
gaussdb=# SELECT * FROM gs_seg_get_segment_layout();
 version | section_id | section_type | extent_size | extent_page_count | extent_count_start |
extent_count_end |   total_size
---------+------------+--------------+-------------+-------------------+--------------------+------------------
+----------------
 1.0     |          1 | meta         |        8192 |                 1 |                  0 |                0 |          8192
 1.0     |          2 | data         |       65536 |                 8 |                  1 |               16 |       1048576
 1.0     |          3 | data         |     1048576 |               128 |                 17 |              143 |     134217728
 1.0     |          4 | data         |     8388608 |              1024 |                144 |              255 |
1073741824
 1.0     |          5 | data         |    33554432 |              4096 |                256 |          1025255 |
34394366541824
(5 rows)
```

- gs_seg_get_datafile_layout()

  Description: Queries the static layout of data files 1 to 5. Only an administrator can query the information.

  The following describes the return values.

Name	Type	Description
version	TEXT	Segment-page version. The default value is **1.0**.
seg_storage_type	TEXT	• **segment** indicates common segment-page data. • **hashbucket** indicates hash bucket data.
file_id	INTEGER	Data file ID. Value range: [1,5] of the int4 type.
section_id	INTEGER	Data section ID of a data file.
section_type	TEXT	Type of the data file section. Value range: • **file_header** indicates the file header. • **spc_header** indicates the space header. • **map_header** indicates the mapping header. • **map_pages** indicates the mapping page. • **ip_pages(inverse pointer pages)** indicates the inverse pointer page. • **data_pages** indicates the data page.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
page_start	BIGINT	Start page number of the data section.
page_end	BIGINT	End page number of the data section.
page_count	BIGINT	Number of pages in the data section.
total_size	BIGINT	Size of the data section. The unit is byte.

Example:

```
gaussdb=# SELECT * FROM gs_seg_get_datafile_layout();
 version | seg_storage_type | file_id | section_id | section_type |  page_start  |   page_end   |
page_count  |   total_size
---------+------------------+---------+------------+--------------+--------------+--------------+-------------
+-----------------
 1.0     | segment          |       1 |          0 | file_header  |            0 |            0 |            1 |            8192
 1.0     | segment          |       1 |          1 | spc_header   |            1 |            1 |            1 |            8192
 1.0     | segment          |       1 |          2 | map_header   |            2 |            2 |            1 |            8192
 1.0     | segment          |       1 |          3 | map_pages    |            3 |           66 |           64 |          524288
 1.0     | segment          |       1 |          4 | ip_pages     |           67 |         4156 |         4090 |
33505280
 1.0     | segment          |       1 |          5 | data_pages   |         4157 |      4147260 |      4143104 |
33940307968

... (There is a large amount of data. Only part of the data is displayed.)
```

- gs_seg_get_slice_layout(file_id INTEGER, bucketnode INTEGER,slice_id INTEGER)

  Description: Outputs the static layout of a given data file fragment. Only an administrator can query the information.

  The following describes the parameters.

Name	Type	Description
file_id	INTEGER	Data file ID. Value range: [1,5] of the int4 type.
bucketnode	INTEGER	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
slice_id	INTEGER	Slice file ID.

  The following describes the return values.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
version	TEXT	Segment-page version. The default value is **1.0**.
section_id	INTEGER	Data section ID of a data file.
section_type	INTEGER	Type of the data file section. Value range: • **file_header** indicates the file header. • **spc_header** indicates the space header. • **map_header** indicates the mapping header. • **map_pages** indicates the mapping page. • **ip_pages(inverse pointer pages)** indicates the inverse pointer page. • **data_pages** indicates the data page.
page_start	BIGINT	Start page number of the data section.
page_end	BIGINT	End page number of the data section.
page_count	BIGINT	Number of pages in the data section.
total_size	BIGINT	Size of the data section. The unit is byte.

Example:

```
gaussdb=# SELECT * FROM gs_seg_get_slice_layout(1,1024,0);
 version | section_id | section_type | page_start | page_end | page_count | total_size
---------+------------+--------------+------------+----------+------------+------------
 1.0     |          0 | file_header  |          0 |        0 |          1 |       8192
 1.0     |          1 | spc_header   |          1 |        1 |          1 |       8192
 1.0     |          2 | map_header   |          2 |        2 |          1 |       8192
 1.0     |          3 | map_pages    |          3 |       66 |         64 |     524288
 1.0     |          4 | ip_pages     |         67 |     4156 |       4090 |   33505280
 1.0     |          5 | data_pages   |       4157 |   131071 |     126915 | 1039687680
(6 rows)
```

- gs_seg_get_segment(tablespace_name NAME, bucketnode INTEGER, head_block_id BIGINT)

  Description: Outputs the segment header information of the segment header page in the segment header file of the tablespace. Only an administrator can query the information.

  The following describes the parameters.

Name	Type	Description
tablespace_name	NAME	Tablespace to which a segment object belongs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
bucketnode	INTEGER	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
head_block_id	BIGINT	Page number of a segment header.

The following describes the return values.

Name	Type	Description
blocks	BIGINT	Number of logical pages of a segment.
total_blocks	BIGINT	Number of physical pages of a segment.
extents	INTEGER	Number of logical extents of a segment.
total_extents	INTEGER	Number of physical extents of a segment.
head_lsn	TEXT	Segment header LSN.
level0_slots	BIGINT[]	Level-0 slot array of segment extent mapping.
level1_slots	BIGINT[]	Level-1 slot array of segment extent mapping.
fork_head	BIGINT[]	Fork head array of a segment.

For example, perform the following operations after creating a segment-page ordinary table:

```
gaussdb=# SELECT * FROM gs_seg_get_segment('pg_default', 1024, 4157);
 blocks | total_blocks | extents | total_extents | head_lsn | level0_slots | level1_slots |   fork_head
--------+--------------+---------+---------------+----------+--------------+--------------+---------------
      9 |           16 |       2 |             2 | 62211744 | {4157,4165}  | {}           | {4157,4158,-1}
(1 row)
```

- gs_seg_get_segment(relid OID, bucketnode INTEGER, partition BOOLEAN DEFAULT FALSE, forknum INTEGER DEFAULT 0)

  Description: Outputs the corresponding segment header information based on **relid** and **bucketnode**. Only an administrator can query the information.

  The following describes the parameters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
relid	OID	Table OID.
bucketnode	INTEGER	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
partition	BOOLEAN DEFAULT FALSE	Specifies whether the segment object is a partition. The default value is **false**. Value range: Boolean values, indicating whether the object corresponding to the input OID is a partition.
forknum	INTEGER DEFAULT 0	Fork number of a data file. The default value is **0**. Value range: • **0**: main fork. • **1**: FSM fork. • **2**: VM fork.

The following describes the return values.

Name	Type	Description
blocks	BIGINT	Number of logical pages of a segment.
total_blocks	BIGINT	Number of physical pages of a segment.
extents	INTEGER	Number of logical extents of a segment.
total_extents	INTEGER	Number of physical extents of a segment.
head_lsn	TEXT	Segment header LSN.
level0_slots	BIGINT[]	Level-0 slot array of segment extent mapping.
level1_slots	BIGINT[]	Level-1 slot array of segment extent mapping.
fork_head	BIGINT[]	Fork head array of a segment.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

For example, perform the following operations on a DN, and ensure that **relid** must be the OID of the queried segment page object on the DN:

```
gaussdb=# SELECT * FROM gs_seg_get_segment(16768, 1024);
 blocks | total_blocks | extents | total_extents | head_lsn | level0_slots | level1_slots |    fork_head
--------+--------------+---------+---------------+----------+--------------+--------------+-----------------
      9 |           16 |       2 |             2 | 62211744 | {4157,4165}  | {}           | {4157,4158,4294967295}
(1 row)
```

- gs_seg_get_extents(tablespace_name NAME, bucketnode INTEGER, head_block_id BIGINT)

  Description: Outputs all extents of the segment object on the segment header page in the segment header file of the tablespace, including **segment head**, **fork head**, **level1 page** in file 1, and data extents in files 2 to 5. Only an administrator can query the information.

  The following describes the parameters.

Name	Type	Description
tablespace_name	NAME	Tablespace to which a segment object belongs.
bucketnode	INTEGER	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
head_block_id	BIGINT	Page number of a segment header.

The following describes the return values.

Name	Type	Description
extent_id	INTEGER	Logical extent number.
file_id	INTEGER	ID of the data file where the extent is located. Value range: [1,5] of the int4 type.
forknum	INTEGER	Fork of a segment object. Value range: • **0**: main fork. • **1**: FSM fork. • **2**: VM fork.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
block_id	BIGINT	Start page number in the data file where the extent is located.
blocks	INTEGER	Extent size. Value range: • **1** indicates file 1. • **8** indicates file 2. • **128** indicates file 3. • **1024** indicates file 4. • **4096** indicates file 5.
usage_type	TEXT	Usage type of an extent. Value range: • **segment head** indicates the segment header. • **fork head** indicates the fork header. • **level1 page** indicates the level-1 page. • **data extent** indicates data extents.

For example, perform the following operations after creating a segment-page ordinary table:

```
gaussdb=# SELECT * FROM gs_seg_get_extents('pg_default', 1024, 4157);
 extent_id | file_id | forknum | block_id | blocks |  usage_type
-----------+---------+---------+----------+--------+--------------
           |       1 |       0 |     4157 |      1 | segment head
         0 |       2 |       0 |     4157 |      8 | data extent
         1 |       2 |       0 |     4165 |      8 | data extent
(3 rows)
```

- gs_seg_get_extents(relid OID, bucketnode INTEGER, partition BOOLEAN DEFAULT FALSE, forknum INTEGER DEFAULT 0)

  Description: Outputs all extents of the segment object on the segment header page in the corresponding segment header file, including **segment head**, **fork head**, **level1 page** in file 1 and, data extents in files 2 to 5. Only an administrator can query the information.

  The following describes the parameters.

Name	Type	Description
relid	OID	Table OID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
bucketnode	INTEGER	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
partition	BOOLEAN DEFAULT FALSE	Specifies whether the segment object is a partition. The default value is **false**. Value range: Boolean values, indicating whether the object corresponding to the input OID is a partition.
forknum	INTEGER DEFAULT 0	Fork number of a data file. The default value is **0**. Value range: • **0**: main fork. • **1**: FSM fork. • **2**: VM fork.

The following describes the return values.

Name	Type	Description
extent_id	INTEGER	Logical extent number.
file_id	INTEGER	ID of the data file where the extent is located. Value range: [1,5] of the int4 type.
forknum	INTEGER	Fork of a segment object. Value range: • **0**: main fork. • **1**: FSM fork. • **2**: VM fork.
block_id	BIGINT	Start page number in the data file where the extent is located.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
blocks	INTEGER	Extent size. Value range: • **1** indicates file 1. • **8** indicates file 2. • **128** indicates file 3. • **1024** indicates file 4. • **4096** indicates file 5.
usage_type	TEXT	Usage type of an extent. Value range: • **segment head** indicates the segment header. • **fork head** indicates the fork header. • **level1 page** indicates the level-1 page. • **data extent** indicates data extents.

In the following example, **relid** must be a valid OID of a multipart page object:

```
gaussdb=# SELECT * FROM gs_seg_get_extents(16768, 1024);
 extent_id | file_id | forknum | block_id | blocks |  usage_type
-----------+---------+---------+----------+--------+--------------
           |       1 |       0 |     4157 |      1 | segment head
         0 |       2 |       0 |     4157 |      8 | data extent
         1 |       2 |       0 |     4165 |      8 | data extent
(3 rows)
```

- gs_seg_free_spc_remain_segment(tablespace_name NAME, head_file_id INTEGER, bucketnode INTEGER, head_block_id BIGINT)

  Description: Releases the page occupied by the segment page-based residual segment in file 1 in a specified tablespace. You can query residual segments in the **GS_SEG_SPC_REMAIN_SEGMENTS** view. Only an administrator can query the information. This query can be executed only on the primary node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- This function is an offline segment-page clearing method. The current version provides the online clearing function: gs_seg_auto_free_remain_segment().
- To ensure the consistency between the queried residual segments and residual extents and the consistency of residual clearing, this function and the **GS_SEG_SPC_REMAIN_SEGMENTS** view must be executed in the DDL- or DML-restricted state. The current version does not provide the DML/DDL restriction capability. Therefore, when using this feature, ensure that the operating environment is in the DDL-\DML-restricted state.
- Multiple residual clearing operations cannot be performed on the same database of the same node at the same time, including manually executing this function, manually performing online clearing, and performing online clearing scheduled by AUTOVACUUM.

The following describes the parameters.

Name	Type	Description
tablespace_name	NAME	Tablespace name.
head_file_id	INTEGER	ID of a data file in which a segment header of a segment-page-type residual segment is located. The value is **1**.
bucketnode	INTEGER	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
head_block_id	BIGINT	Page number of a segment header.

Return type: void.

Example:

```
gaussdb=# SELECT * FROM gs_seg_free_spc_remain_segment('pg_default', 1, 1024, 4159);
 gs_seg_free_spc_remain_segment
--------------------------------

(1 row)
```

- gs_seg_auto_free_remain_segment()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Releases the page occupied by the segment page-based residual segment in file 1 in a specified tablespace on the local node. This query can be executed only by administrators on the primary node.

NOTICE

- This function is an online segment-page clearing method.
- This function can be used only when **enable_auto_segment_remain_cleanup** is set to **on**. For details about how to enable it, see **Developer Options**.
- Multiple residual clearing operations cannot be performed on the same database of the same node at the same time, including manually executing this function, manually performing offline clearing, and performing online clearing scheduled by AUTOVACUUM.

Parameter: void.

Return type: void.

Example:

```
gaussdb=# SELECT * FROM gs_seg_auto_free_remain_segment();
 gs_seg_auto_free_remain_segment
---------------------------------

(1 row)
```

- gs_seg_free_spc_remain_extent(tablespace_name NAME, file_id INTEGER, bucketnode INTEGER, forknum INTEGER, block_id BIGINT)

  Description: Releases residual segment-page-type isolated extents in a specified tablespace. Residual isolated extents can be queried in the **GS_SEG_SPC_REMAIN_EXTENTS** view. Only an administrator can query the information. This query can be executed only on the primary node.

  NOTICE

  - This function is an offline segment-page clearing method. The current version provides the online clearing function: gs_seg_auto_free_remain_segment().
  - Multiple residual clearing operations cannot be performed on the same database of the same node at the same time, including manually executing this function, manually performing online clearing, and performing online clearing scheduled by AUTOVACUUM.

  The following describes the parameters.

Name	Type	Description
tablespace_name	NAME	Tablespace name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
file_id	INTEGER	ID of the data file where the segment-page-type residual extent is located. The value ranges from files 1 to 5.
bucketnode	INTEGER	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
forknum	INTEGER	Fork number of a data file. Value range: • **0**: main fork. • **1**: FSM fork. • **2**: VM fork.
block_id	BIGINT	Page number of a segment header.

Return type: void.

Example:

```
gaussdb=# SELECT * FROM gs_seg_free_spc_remain_extent('pg_default', 1, 1024, 0, 4159);
 gs_seg_free_spc_remain_extent
-------------------------------

(1 row)
```

- gs_seg_get_datafiles(database_name NAME)

  Description: Displays information about all data files of an instance. Only an administrator can query the information.

  The following describes the parameters.

Name	Type	Description
database_name	NAME	Database name. The default value is **current_database()**, indicating the name of the current database.

The following describes the return values.

Name	Type	Description
file_name	TEXT	Data file name, for example, **base/17467/2_fsm**.
file_id	INTEGER	Data file ID. The value ranges from files 1 to 5.
bucketnode	INTEGER	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
forknum	INTEGER	Fork type of a data file.
tablespace_name	NAME	Name of the tablespace to which a data file belongs.
contents	TEXT	Storage content of a data file. Value range: • **permanent**: permanent. • **unlogged**: no log. • **temporary**: global temporary. • **temporary2**: local temporary.
extent_size	INTEGER	Extent size of a data file.
meta_blocks	BIGINT	Number of allocated metadata pages of a data file.
data_blocks	BIGINT	Number of allocated data pages of a data file.
total_blocks	BIGINT	Total number of physical pages in a data file.
high_water_mark	BIGINT	High watermark of the number of pages used by a data file.
utilization	REAL	Percentage of used blocks to the total number of blocks, that is, (**data_blocks** + **meta_blocks**)/ **total_blocks**.

For example, perform the following operations after creating a segment-page table:

```
gaussdb=# SELECT * FROM gs_seg_get_datafiles();
  file_name   | file_id | bucketnode | forknum | tablespace_name | contents  | extent_size | meta_blocks
| data_blocks | total_blocks | high_water_mark | utilization
--------------+---------+------------+---------+-----------------+-----------+-------------+-------------
+-------------+--------------+-----------------+-------------
 base/15949/1 |       1 |       1024 |       0 | pg_default      | permanent |           1 |        4157 |           1
```

```
|       16384 |      1048702976 | 2.24208e-44
 base/15949/2 |       2 |       1024 |       0 | pg_default     | permanent |          8 |       4157 |          8
|       16384 |      1048717312 | 2.24208e-44
(2 rows)
```

- gs_seg_get_spc_extents(tablespace_name NAME, file_id INTEGER, bucketnode INTEGER, forknum INTEGER, skip_unused BOOLEAN DEFAULT TRUE)

  Description: Obtains all extent information of a specified tablespace. Only an administrator can query the information.

  The following describes the parameters.

Name	Type	Description
tablespace_name	NAME	Tablespace name.
file_id	INTEGER	Data file ID. Value range: [1,5] of the int4 type.
bucketnode	INTEGER	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
forknum	INTEGER	Data file fork. Value range: • **0**: main fork. • **1**: FSM fork. • **2**: VM fork.
skip_unused	BOOLEAN DEFAULT TRUE	Specifies whether to output only allocated extents. The default value is **TRUE**, indicating that only allocated extents are output.

  The following describes the return values.

Name	Type	Description
block_id	BIGINT	Start page number of a data extent.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
blocks	INTEGER	Data extent size. Value range: ● **1** indicates file 1. ● **8** indicates file 2. ● **128** indicates file 3. ● **1024** indicates file 4. ● **4096** indicates file 5.
contents	TEXT	Stores content. Value range: ● **permanent**: permanent. ● **unlogged**: no log. ● **temporary**: global temporary. ● **temporary2**: local temporary.
in_used	TEXT	Specifies whether a resource has been allocated. Value range: ● **Y**: allocated. ● **N**: not allocated.
mapblock_location	TEXT	Position of the extent in the map block. Format: (page_id, offset).
head_file_id	INTEGER	ID of the data file where the segment header is located.
head_block_id	BIGINT	Page number of a segment header.
usage_type	TEXT	Usage type of an extent. Value range: ● **segment head** indicates the segment header. ● **fork head** indicates the fork header. ● **level1 page** indicates the level-1 page. ● **data extent** indicates data extents.
remain_flag	TEXT	Specifies whether it is a residual extent after the SHRINK operation. Value range: ● **Y**: residual extent upon SHRINK. ● **N**: not a residual extent upon SHRINK.
special_data	INTEGER	Special data section of the inverse pointer corresponding to an extent.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
ipblock_location	TEXT	Position of the inverse pointer corresponding to an extent. Format: (block_id, offset).

Example:

```
gaussdb=# SELECT * FROM gs_seg_get_spc_extents('pg_default', 1,1024, 0);
 block_id | blocks | contents  | in_used | mapblock_location | head_file_id | head_block_id |  usage_type
| remain_flag | special_data | ipblock_location
----------+--------+-----------+---------+-------------------+--------------+---------------+--------------
+-------------+--------------+------------------
    4157 |      1 | permanent | Y       | (4157,0)          |            1 |          4157 | segment head | N
|           0 | (67,0)
    4158 |      1 | permanent | Y       | (4157,1)          |            1 |          4158 | fork head    | N
|           1 | (67,1)
(2 rows)
```

1.6.27.12 Bucket System Functions

- gs_bucket_get_data_distribution(relation REGCLASS, forknum INTEGER, including_toast Boolean)

  Description: Obtains the data distribution of a specific hash bucket table.

  The following describes the parameters.

Name	Type	Description
relation	REGCLASS	Table name.
forknum	INTEGER	Data file branch. Possible values include: • **0**: main fork. • **1**: FSM fork. • **2**: VM fork. • **-1**: sum of all other **forknum** values.
including_toast	Boolean	Specifies whether to include the TOAST table in data distribution statistics.

  Returned values

Name	Type	Description
node_name	TEXT	Database node name.
bucketnode	INT2	Bucket ID.
used_blocks	INT8	Number of used data blocks under the bucket ID.
allocated_blocks	INT8	Number of requested data blocks under the bucket ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# select * from gs_bucket_get_data_distribution('bucket_table', 0, true);

 node_name | bucketnode | used_blocks | allocated_blocks
-----------+------------+-------------+------------------
 datanode1 |          0 | 1           | 8
 datanode1 |          1 | 1           | 8
 datanode1 |          2 | 1           | 8
 datanode1 |          3 | 1           | 8
 datanode1 |          4 | 1           | 8
 datanode1 |          5 | 1           | 8
 datanode1 |          6 | 1           | 8
 datanode1 |          7 | 1           | 8
 datanode1 |          8 | 1           | 8
 datanode1 |          9 | 1           | 8
 datanode1 |         10 | 1           | 8
 datanode1 |         11 | 1           | 8
 datanode1 |         12 | 1           | 8
 datanode1 |         13 | 1           | 8
 datanode1 |         14 | 1           | 8
 datanode1 |         15 | 1           | 8
 datanode1 |         16 | 1           | 8
 datanode1 |         17 | 1           | 8
 datanode1 |         18 | 1           | 8
 datanode1 |         19 | 1           | 8
 datanode1 |         20 | 1           | 8
 datanode1 |         21 | 1           | 8
 datanode1 |         22 | 1           | 8
 datanode1 |         23 | 1           | 8
 datanode1 |         24 | 1           | 8
(Only part of the data is displayed.)
```

- gs_redis_get_plan(origin_group_id OID, target_group_id OID)

  Description: Obtains a complete migration plan. It can be called internally only during scale-out and only on CNs.

  Parameters:

Name	Type	Description
origin_group_id	OID	Source node group.
target_group_id	OID	New node group.

  Returned values

Name	Type	Description
sender_id	OID	Source node OID.
sender_name	CSTRING	Name of the source node.
receiver_id	OID	New node OID.
receiver_name	CSTRING	Name of the new node.
bucket_number	INT4	Number of buckets in **bucket_list**.
bucket_list	OIDVECTOR_EXTEND	List of buckets involved in the migration plan.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_redis_get_bucket_statistics()

  Description: Obtains the log flow transmission status. It can be called internally only during scale-out.

  Parameter: void

  Returned values

Name	Type	Description
bucket_id	OID	Bucket ID.
redis_state	INT1	Scale-out status of the bucket. The value **0** indicates that the scale-out has not started, and the value **1** indicates that the scale-out baseline data has been completed.
xlog_count	INT8	Number of Xlogs generated by the bucket on the original DN after the current database scale-out starts (that is, after gs_redis_set_distributed_db is called last time).
sndr_latest_lsn	INT8	Latest LSN generated by the bucket on the original DN after the current database scale-out starts (that is, after gs_redis_set_distributed_db is called last time).
parser_latest_lsn	INT8	Latest LSN where the bucket is parsed to by the scale-out thread of the original DN after the current database scale-out starts (that is, after gs_redis_set_distributed_db is called last time).
parser_latest_lsn_new	INT8	Latest LSN of bucketxlog where the bucket is parsed to by the scale-out thread of the original DN after the current database scale-out starts (that is, after gs_redis_set_distributed_db is called last time).
rcvr_redo_latest_lsn	INT8	Latest LSN where the bucket is replayed to by the scale-out thread of the new DN after the current database scale-out starts (that is, after gs_redis_set_distributed_db is called last time).
rcvr_redo_latest_lsn_new	INT8	Latest LSN of bucketxlog where the bucket is parsed to by the scale-out thread of the original DN after the current database scale-out starts (that is, after gs_redis_set_distributed_db is called last time).
rcvr_checkpoint	INT8	Checkpoint of the bucket after the current database scale-out starts (that is, after the last calling of gs_redis_set_distributed_db).
rcvr_redo_start_lsn	INT8	Original LSN for starting the replay of the bucket after the scale-out of the current database starts (that is, after the last calling of gs_redis_set_distributed_db).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
rcvr_redo_end_lsn	INT8	Original LSN for ending the replay of the bucket after the scale-out of the current database starts (that is, after the last calling of gs_redis_set_distributed_db).

- gs_redis_set_distributed_db(db_name CSTRING, bucketgroup SMALLINT)

  Description: Sets the database that is being redistributed using hash bucket. It can be called internally only during scale-out and only on DNs.

  Parameters:

  - **db_name**: specifies the name of the database to be redistributed.
  - **bucketgroup**: bucket group to be re-distributed. This has no effect in the current version.

  Return type: void

- gs_redis_parse_slicebucket_tranfer_file(plan_json CSTRING, type CSTRING)

  Description: Obtains the slice bucket migration plan. It can be called internally only during scale-out and only on CNs. It is not supported in the current version.

  The following describes the parameters.

Name	Type	Description
plan_json	TEXT	Slice migration file specified by the user.
type	CSTRING	Plan type, which can be **'incremental'** or **'full'**. The default value is **'full'**.

  The following describes the return types.

Name	Type	Description
idx	INT	ID of the database to be migrated.
database_name	CSTRING	Name of the database to be migrated.

- gs_redis_get_slicebucket_plan()

  Description: Obtains the slice bucket migration plan of the current database. It can be called internally only during scale-out and only on CNs. It is not supported in the current version.

  Parameter: void

  Returned values

Name	Type	Description
sender_id	OID	OID of the source node.
sender_name	CSTRING	Name of the source node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
receiver_id	OID	OID of the target node.
receiver_name	CSTRING	Name of the new node.
bucket_number	INT4	Number of buckets in **bucket_list**.
bucket_list	OIDVECTOR_EXTEND	List of buckets involved in the migration plan.
bucketgroup	OID	Group to which the bucket in the bucket list belongs. The value ranges from 0 to 4.

- gs_redis_hashbucket_update_segment_header(origin_group_id OID, target_group_id OID)

  Description: Updates the headers of all hash bucket tables in the current database. It can be called internally only during scale-out and only on CNs.

  Parameters:

Name	Type	Description
origin_group_id	OID	Source node group.
target_group_id	OID	New node group.

  Return type: Boolean

- gs_redis_local_get_segment_header(table_name REGCLASS, bucketlist OIDVECTOR_EXTEND)

  Description: Obtains the header of a segment-page table. It can be called only on DNs.

  Parameters:

Name	Type	Description
table_name	REGCLASS	Table name.
bucketlist	OIDVECTOR_EXTEND	Bucket list.

  Return type: header_info

  Example:

```
gaussdb=#  SELECT * FROM gs_redis_local_get_segment_header('mytable', '256');
 gs_redis_local_get_segment_header
-----------------------------------------------------------------------------------------------------------------------------
-----
4294967295,4294967295,4294967295,4294967295,4294967295,4294967295,4294967295,4294967295,
4294967295,4294967295,4294967295,4294967295,4294967295, ... (Only part of the data is displayed.)
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_redis_local_update_segment_header(table_name REGCLASS, header_info CSTRING)

  Description: Updates the header of a segment-page table. It can be called only on DNs.

  Parameters:

Name	Type	Description
table_name	REGCLASS	Table name.
header_info	CSTRING	Header of a segment-page table.

  Return type: void

  Example:

```
gaussdb=#  SELECT * FROM gs_redis_local_update_segment_header('mytable',
'4294967295,4294967295,4294967295,4294967295, ....'); (Only part of the data is displayed.)
 gs_redis_local_update_segment_header
---------------------------------------------
 0
(1 row)
```

- gs_redis_hashbucket_update_inverse_pointer(buckets TEXT, origin_dn_name TEXT, new_dn_name TEXT)

  Description: Updates the inverse pointers of buckets of this batch in all hash bucket tables in the current database. It can be called internally only during scale-out and only on CNs.

  Parameters:

Name	Type	Description
buckets	TEXT	List of buckets that are brought online this time.
origin_dn_name	TEXT	Source node name.
new_dn_name	TEXT	New node name.

  Return type: void

- gs_redis_local_get_inverse_pointer(buckets TEXT, origin_dn_name TEXT, new_dn_name TEXT)

  Description: Obtains the inverse pointer. It can be called internally only during scale-out and only on DNs.

  The following describes the parameters.

Name	Type	Description
buckets	TEXT	List of buckets that are brought online this time.
origin_dn_name	TEXT	Source node name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
new_dn_name	TEXT	New node name.

Return type: void

- gs_redis_local_update_inverse_pointer(table_name TEXT, header_info TEXT, bucketlist TEXT)

  Description: Records Xlogs for updating inverse pointers. It can be called internally only during scale-out and only on DNs.

  The following describes the parameters.

Name	Type	Description
table_name	REGCLASS	Table name.
header_info	TEXT	Header of a segment-page table.
bucketlist	TEXT	Bucket list.

  Return type: void

- gs_redis_local_set_hashbucket_frozenxid

  Description: Changes the **relfrozenxid64** value of the hash bucket table in the system catalog. It can be called internally only during scale-out and only on DNs.

  Parameter description: void

  Return type: void

- gs_redis_set_hashbucket_frozenxid(origin_group_id OID, target_group_id OID)

  Description: Changes the **relfrozenxid64** value of the hash bucket table in the system catalog on the new DN. It can be called internally only during scale-out and only on CNs.

  Parameters:

Name	Type	Description
origin_group_id	OID	Source node group.
target_group_id	OID	New node group.

  Return type: void

- gs_redis_set_nextxid(xid BIGINT)

  Description: Modifies the **next_xid** value of a DN. It can be called internally only during scale-out and only on DNs.

  Parameter: **xid** indicates the expected value of **next_xid**.

  Return type: void

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_redis_set_csn(csn BIGINT)

  Description: Changes the **next_csn** value of a DN. It can be called internally only during scale-out and only on DNs in GTM-free mode.

  Parameters:

  – **csn**: specifies the expected value of **next_csn**.

  Return type: void

- gs_redis_check_bucket_flush(dn_array NAME[])

  Description: Determines whether all private buffers for RTO replay are flushed. It can be called internally only during scale-out.

  Parameters:

  – **dn_array**: specifies the DN name list.

  Return type: Boolean

- gs_redis_get_flush_page_lsn(isclean bool)

  Description: Queries the page refreshing information of bucket scale-out replay.

  Permission: It can be called only in the debug version and can be called internally only during scale-out.

  Parameters:

  – **isclean**: specifies whether to clear page refreshing information. The value is of the Boolean type.

  Returned values

Name	Type	Description
node_name	TEXT	DN where the page is located.
space_id	OID	Tablespace ID.
db_id	OID	Database ID.
file_id	OID	Data file ID. The value ranges from 1 to 5.
bucket_id	OID	Bucket partition ID. The value ranges from 0 to 1023.
forknum	OID	Data file fork. The value ranges from 0 to 2.
blocknum	OID	Offset page number of the physical page in the data file.
latest_flush_lsn	UINT8	Latest LSN of the dirty page to be flushed.

- gs_redis_show_bucketxid(bucketid_list OIDVECTOR_EXTEND)

  Description: Queries the bucketxid corresponding to a specified bucket. It can be called internally only during scale-out and only on DNs.

  Parameters:

  – **bucketid_list**: specifies the bucket list.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: CString

- gs_redis_drop_bucket_files(plan_json CSTRING)

  Description: Deletes the range bucket files that have been physically migrated from the source node. It can be called internally only during scale-out and only on CNs. It is not supported in the current version.

  The following describes the parameters.

Name	Type	Description
plan_json	CSTRING	Slice migration file specified by the user.

  Return type: Boolean

- gs_redis_drop_bucket_files(origin_group_id OID, target_group_id OID)

  Description: Deletes the files that have been physically migrated from the source node. It can be called internally only during scale-out and only on CNs.

  Parameters:

Name	Type	Description
origin_group_id	OID	Source node group.
target_group_id	OID	New node group.

  Return type: Boolean

- gs_redis_local_drop_bucket_files(bucketlist CSTRING, bucketnum SMALLINT)

  Description: Deletes the corresponding bucket list. It can be called internally only during scale-out and only on DNs.

  The following describes the parameters.

Name	Type	Description
bucketlist	CSTRING	Specified bucket list.
bucketnum	SMALLINT	Number of buckets.

  Return type: Boolean

- gs_redis_local_drop_bucket_files(database_name NAME, bucketlist CSTRING, bucketnum SMALLINT)

  Description: Deletes the bucket list of the corresponding database from the range bucket table. It can be called internally only during scale-out and only on DNs. It is not supported in the current version.

  The following describes the parameters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
database_name	NAME	Specified database name.
bucketlist	CSTRING	Specified bucket list.
bucketnum	SMALLINT	Number of buckets.

Return type: Boolean

- gs_redis_slicebucket_update_segment_header()

  Description: Updates the headers of all slice bucket tables in the current database. It can be called internally only during scale-out and only on CNs. It is not supported in the current version.

  Input parameter: void

  Return type: Boolean

- gs_redis_get_undo_state(mode INTEGER)

  Description: Obtains the undo status during scale-out and redistribution. It can be called internally only during scale-out.

  The following describes the parameters.

Name	Type	Description
mode	INTEGER	Mode for determining the undo status during redistribution: Options include: ● **0**: specifies whether there are transactions to be undone before the undo monitoring point **next_xid**. ● **1**: specifies whether there are new undo transactions whose ID is smaller than the monitoring point **next_xid**. ● **2**: specifies whether there are transactions to be undone whose ID is greater than the monitoring point **next_xid**.

  The return value **state** is of the INT4 type, indicating the undo status during scale-out and redistribution. The meanings of the return values vary depending on the parameter values:

  – When **mode** is set to **0**:

    ▪ If **state** is set to **0**, there are no transactions to be undone (**xmin** > **next_xid**).

    ▪ If **state** is set to **1**, there are no transactions to be undone (**xmin** ≤ **next_xid**).

    ▪ If **state** is set to **2**, there are transactions to be undone.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - When **mode** is set to **1**:
    - If **state** is set to **0**, there are no new undo transactions before the monitoring point **next_xid**.
    - If **state** is set to **1**, there are new undo transactions before the monitoring point **next_xid**.
  - When **mode** is set to **2**:
    - If **state** is set to **0**, there are no transactions to be undone after the monitoring point **next_xid**.
    - If **state** is set to **1**, there are transactions to be undone after the monitoring point **next_xid**.

1.6.27.13 Undo System Functions

To check the storage mode used by the current rollback segment, query the **undostoragetype** field in the gs_global_config system catalog. **segpage** indicates the segment-page mode, which is a reserved parameter and is not supported currently. **page** indicates the page mode.

Example:

```
gaussdb=# SELECT * FROM gsd_global_config where name like '%undostoragetype%';
      name       |  value
-----------------+---------
 undostoragetype |  page
(1 row)
```

NOTE

Undo system functions can be executed on DNs but cannot be executed on CNs.

- gs_undo_meta(type, zoneId, location)

  Description: Specifies metadata of a module in the undo system.

  Parameters:
  - **type** (metadata type)
    - **0**: indicates metadata corresponding to an undo zone (record).
    - **1**: indicates metadata corresponding to an undo zone (transaction slot).
    - **2**: indicates metadata corresponding to an undo space (record).
    - **3**: indicates metadata corresponding to an undo space (transaction slot).
  - **zoneId** (undo zone ID)
    - **–1**: indicates metadata of all undo zones.
    - **0** to 1024 x 1024 – 1: indicates the meta information of the corresponding zone ID.
  - **location** (read location)
    - **0**: indicates that data is read from the current memory.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **1**: indicates that data is read from a physical file.

Return type: record

NOTE

This system function supports only the rollback segments in page storage mode.

Table 1-147 gs_undo_meta parameters

Category	Parameter	Type	Description
Output parameter	zoneId	oid	ID of an undo zone.
Output parameter	persistType	oid	Persistence level.
Output parameter	insert	text	Position of the next undo record to be inserted.
Output parameter	discard	text	Position of the undo record that is recycled in common mode.
Output parameter	end	text	Position of the undo record that is forcibly recycled. Values smaller than the value of this parameter indicate that the record has been recycled.
Output parameter	used	text	Undo space that has been used.
Output parameter	lsn	text	Modifies the LSN of an undo zone.
Output parameter	pid	oid	ID of a thread bound to an undo zone.

- gs_undo_translot(location, zoneId)

  Description: Specifies transaction slot information of the undo system.

  Parameters:

  - **location** (read location)
    - **0**: indicates that data is read from the current memory.
    - **1**: indicates that data is read from a physical file.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

– **zoneId** (undo zone ID)

  - **–1**: indicates metadata of all undo zones.
  - **0** to 1024 x 1024 – 1: indicates the meta information of the corresponding zone ID.

Return type: record

NOTE

This system function supports only the rollback segments in page storage mode.

Table 1-148 gs_undo_translot parameters

Category	Parameter	Type	Description
Output parameter	groupId	oid	Undo zone ID that is used.
Output parameter	xactId	text	Transaction ID.
Output parameter	startUndoPtr	text	Position where an undo record is inserted at the start of a transaction corresponding to a transaction slot.
Output parameter	endUndoPtr	text	Position where an undo record is inserted at the end of a transaction corresponding to a transaction slot.
Output parameter	lsn	text	Transaction slot pointer.
Output parameter	slot_states	oid	Transaction status. • **0** indicates that the transaction has been committed. • **1** indicates that the task is being executed. • **2** indicates that the rollback is in progress. • **3** indicates that the rollback is complete.

- gs_stat_undo([bool init])
  Description: Collects undo statistics.
  Return type: record

NOTE

When rollback segments are stored in files in page storage mode, the following format is used.

Table 1-149 gs_stat_undo parameters

Category	Parameter	Type	Description
Input parameter	init	bool	(Optional) Specifies whether to clear statistics and restart statistics collection.
Output parameter	curr_used_zone_count	int	Number of used undo zones.
Output parameter	top_used_zones	text	Information about the first three undo zones with the maximum usage. The output format is as follows: • **zoneId1**: used size of zone 1. • **zoneId2**: used size of zone 2. • **zoneId3**: used size of zone 3.
Output parameter	curr_used_undo_size	int	Total size of the undo tablespace that is being used. The unit is MB.
Output parameter	undo_threshold	int	Calculation result of the value of the GUC parameter **undo_space_limit_size** × 80%, in MB.
Output parameter	global_recycle_xid	xid	XID of the transaction recycled to the undo space. The undo records generated by the transaction whose XID is smaller than the value of XID are recycled.
Output parameter	oldest_xmin	xid	Oldest active transaction.
Output parameter	total_undo_chain_len	int8	Total length of all accessed undo chains.
Output parameter	max_undo_chain_len	int8	Maximum length of the accessed undo chain.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	create_undo_file_count	uint32	Number of created undo files.
Output parameter	discard_undo_file_count	uint32	Number of deleted undo files.
Output parameter	info	text	If the input parameter is **false**, suggestions on the **undo_space_limit_size**, **undo_limit_size_per_transaction**, and **undo_retention_time** parameters are provided. If the input parameter is **true**, init (clearing statistics) is required, and only 'The statistics have been initialized.' is displayed.

Example 1: Clear undo statistics.

```
gaussdb=# SELECT * FROM gs_stat_undo(true);
curr_used_zone_count |   top_used_zones    | curr_used_undo_size | undo_threshold | global_recycle_xid
| oldest_xmin | total_undo_chain_len | max_undo_chain_len | create_undo_file_coun
t | discard_undo_file_count |                info
----------------------+---------------------+---------------------+----------------+--------------------
+-------------+----------------------+--------------------+----------------------
--+-------------------------+-------------------------------------
                   3 | 0 : 0, 0 : 0, 0 : 0 |                   1 |         209715 |             15741 |       15741
|                  0 |                   0 |
2 |                     0 | The statistics have been initialized.
(1 row)
```

Example 2: Output undo statistics.

```
gaussdb=# SELECT * FROM gs_stat_undo(false);
curr_used_zone_count |   top_used_zones    | curr_used_undo_size | undo_threshold | global_recycle_xid
| oldest_xmin | total_undo_chain_len | max_undo_chain_len | create_undo_file_coun
t | discard_undo_file_count |
            info

----------------------+---------------------+---------------------+----------------+--------------------
+-------------+----------------------+--------------------+----------------------
--+-------------------------
+-----------------------------------------------------------------------------------------------------------------------------------
-------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------
-
                   3 | 0 : 0, 0 : 0, 0 : 0 |                   1 |         209715 |             16253 |       16253
|                  0 |                   0 |
2 |                     0 | Based on the statistic info from last initialization, undo_space_limit_size is
recommended to set >= 120953 blocks, undo_limit_size_per_transaction is
recommended to set <= 327150 blocks, undo_retention_time is recommended to set <= 259200
seconds. Since last initialization, max undo space used by a single transaction is 32715 blocks.
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_undo_record(undoptr)

  Description: Parses undo records.

  Parameter: **undoptr** (undo record pointer).

  Return type: record

  NOTE

  This system function supports only the rollback segments in page storage mode.

- gs_undo_dump_parsepage_mv(relpath text, blkno bigint, reltype text, rmem boolean)

  Description: Parses the page header information of the disk page in the Ustore table, header information of each tuple, flag bit information, and all historical undo version information that can be queried.

  Return type: text.

  Note: Only SYSADMIN or OPRADMIN can execute this function.

  NOTE

  Currently, this API supports only Ustore tables.

  Table 1-150 gs_undo_dump_parsepage_mv parameters

Category	Parameter	Type	Description
Input parameter	relpath	text	Relative path of the Ustore table data file, in the format of *Tablespace name*|*Database OID*|*relfilenode*. For example, **base/16603/16384**. You can run the **pg_relation_filepath('tablename')** command to query the relative path.
Input parameter	blkno	int8	• **–1**: All block pages are parsed. • **0**–*MaxBlocNumber*: A specified block page is parsed.
Input parameter	reltype	text	Table type. Currently, only the Ustore table is supported. The value is **uheap**.
Input parameter	rmem	Boolean	• **false** • **true** Currently, the value can only be **false**, indicating that the corresponding page is parsed from the disk file.
Output parameter	output	text	Absolute path of the parsing result file.

- gs_undo_meta_dump_zone(zone_id int, read_memory boolean)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Parses undo zone metadata in an undo module.

Return type: record

NOTE

When rollback segments are stored in files in page storage mode, the following format is used.

Table 1-151 gs_undo_meta_dump_zone parameters

Category	Parameter	Type	Description
Input parameter	zone_id	int	Undo zone ID. ● **–1**: All undo zones are queried. ● **0–1048575**: The undo zone metadata is queried based on the zone ID.
Input parameter	read_memory	Boolean	● **true**: Data is read from the current memory. ● **false**: Data is read from the physical file.
Output parameter	zone_id	oid	Undo zone ID.
Output parameter	persist_type	oid	Persistence levels: ● **0**: ordinary table ● **1**: unlogged table ● **2**: temporary table
Output parameter	insert	text	Position of the next undo record to be inserted.
Output parameter	discard	text	Position of the undo record that is recycled in common mode.
Output parameter	forcediscard	text	Position of the undo record that is forcibly recycled. Values smaller than the value of this parameter indicate that the record has been recycled.
Output parameter	lsn	text	Modifies the LSN of an undo zone.

Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT * FROM gs_undo_meta_dump_zone(-1,true);
 zone_id | persist_type | insert | discard | forcediscard |   lsn
---------+--------------+--------+---------+--------------+----------
       0 |            0 | 244577 | 244577  | 244577       | 43967224
       1 |            0 | 108    | 66      | 66           | 43967568
  349525 |            1 | 24     | 24      | 24           | 0
  349526 |            1 | 24     | 24      | 24           | 0
  699050 |            2 | 24     | 24      | 24           | 0
  699051 |            2 | 24     | 24      | 24           | 0
(6 rows)
```

- gs_undo_meta_dump_spaces(zone_id int, read_memory boolean)

  Description: Parses metadata of an undo record space and a transaction slot space in an undo module.

  Return type: record

  NOTE

  This system function supports only the rollback segments in page storage mode.

  Table 1-152 gs_undo_meta_dump_spaces parameters

Category	Parameter	Type	Description
Input parameter	zone_id	int	Undo zone ID. • **–1**: All undo zones are queried. • **0**–**1048575**: The undo zone metadata is queried based on the zone ID.
Input parameter	read_memory	Boolean	• **true**: Data is read from the current memory. • **false**: Data is read from the physical file.
Output parameter	zone_id	int	Undo zone ID.
Output parameter	undorecord_space_tail	text	End position of an undo record space.
Output parameter	undorecord_space_head	text	Start position of an undo record space.
Output parameter	undorecord_space_lsn	text	Modifies the LSN of an undo record space.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	undoslot_space_tail	text	End position of a transaction slot space.
Output parameter	undoslot_space_head	text	Start position of a transaction slot space.
Output parameter	undoreslot_space_lsn	text	Modifies the LSN of a transaction slot space.

- gs_undo_meta_dump_slot(zone_id int, read_memory boolean)
  Description: Parses transaction slot metadata in an undo module.
  Return type: record

  NOTE

  This system function supports only the rollback segments in page storage mode.

  Table 1-153 gs_undo_meta_dump_slot parameters

Category	Parameter	Type	Description
Input parameter	zone_id	int	Undo zone ID. • **–1**: All undo zones are queried. • **0**–**1048575**: The undo zone metadata is queried based on the zone ID.
Input parameter	read_memory	Boolean	• **true**: Data is read from the current memory. • **false**: Data is read from the physical file.
Output parameter	zone_id	int	Undo zone ID.
Output parameter	allocate	text	Allocation position of the undo transaction slot.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	recycle	text	Recycling position of the undo transaction slot.
Output parameter	frozen_xid	text	Frozen XID, which is used to determine the visibility.
Output parameter	global_frozen_xid	text	Minimum frozen XID in the system. Transactions whose XID is smaller than the value of this parameter are visible.
Output parameter	recycle_xid	text	Recycled XID. Transactions whose XID is smaller than the value of this parameter are recycled.
Output parameter	global_recycle_xid	text	Minimum recycled XID in the system. Transactions whose XID is smaller than the value of this parameter are recycled.

- gs_undo_translot_dump_slot(zone_id int, read_memory boolean)

  Description: Parses a transaction slot in an undo zone.

  Return type: record

  NOTE

  When rollback segments are stored in files in page storage mode, the following format is used.

Table 1-154 gs_undo_translot_dump_slot parameters

Category	Parameter	Type	Description
Input parameter	zone_id	int	Undo zone ID. • **–1**: All undo zones are queried. • **0**–**1048575**: The undo zone metadata is queried based on the zone ID.
Input parameter	read_memory	Boolean	• **true**: Data is read from the current memory. • **false**: Data is read from the physical file.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	zone_id	oid	Undo zone ID.
Output parameter	slot_xid	text	Transaction ID.
Output parameter	start_undoptr	text	Position where the undo record is inserted at the start of the transaction corresponding to a slot.
Output parameter	end_undoptr	text	Position where the undo record is inserted at the end of the transaction corresponding to a slot.
Output parameter	slot_ptr	text	Position of a transaction slot.
Output parameter	slot_states	oid	Transaction state. • **0**: committed • **1**: being executed • **2**: being rolled back • **3**: rolled back

Example:

```
gaussdb=# SELECT * FROM gs_undo_translot_dump_slot(-1,true);
 zone_id |     slot_xid     |  start_undoptr   |   end_undoptr    |     slot_ptr     | slot_states
---------+------------------+------------------+------------------+------------------+-------------
       1 | 0000000000015758 | 0000000000000042 | 000000000000006C | 0000000000000038 |           0
(1 row)
```

- gs_undo_translot_dump_xid(slot_xid xid, read_memory boolean)

  Description: Parses a transaction slot in an undo zone based on the XID.

  Return type: record

  NOTE

  When rollback segments are stored in files in page storage mode, the following format is used.

Table 1-155 gs_undo_translot_dump_xid parameters

Category	Parameter	Type	Description
Input parameter	slot_xid	xid	Transaction ID to be queried.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Input parameter	read_memory	Boolean	• **true**: Data is read from the current memory. • **false**: Data is read from the physical file.
Output parameter	zone_id	oid	Undo zone ID.
Output parameter	slot_xid	text	Transaction ID.
Output parameter	start_undoptr	text	Position where the undo record is inserted at the start of the transaction corresponding to a slot.
Output parameter	end_undoptr	text	Position where the undo record is inserted at the end of the transaction corresponding to a slot.
Output parameter	slot_ptr	text	Position of a transaction slot.
Output parameter	slot_states	oid	Transaction state. • **0**: committed • **1**: being executed • **2**: being rolled back • **3**: rolled back

Example:

```
gaussdb=# SELECT * FROM gs_undo_translot_dump_xid('15758',false);
 zone_id |     slot_xid     |  start_undoptr   |   end_undoptr    |     slot_ptr     | slot_states
---------+------------------+------------------+------------------+------------------+-------------
       1 | 0000000000015758 | 0000000000000042 | 000000000000006C | 0000000000000038 |           0
(1 row)
```

- gs_undo_dump_record(undoptr bigint)

  Description: Parses the undo record information of a specified URP.

  Return type: record

  NOTE

  When rollback segments are stored in files in page storage mode, the following format is used.

Table 1-156 gs_undo_dump_record parameters

Category	Parameter	Type	Description
Input parameter	undoptr	xid	Start position of the undo record to be parsed. The value is in decimal format.
Output parameter	undoptr	xid	Start position of the undo record to be parsed in the current undo zone.
Output parameter	xactid	xid	Transaction ID.
Output parameter	cid	text	Command ID.
Output parameter	reloid	text	Relation OID.
Output parameter	relfilenode	text	Relfinode of the file.
Output parameter	utype	text	Undo record type.
Output parameter	blkprev	text	Position of the previous undo record in the same block.
Output parameter	blockno	text	Block number.
Output parameter	uoffset	text	Undo record offset.
Output parameter	prevurp	text	Position of the previous undo record.
Output parameter	payloadlen	text	Length of the undo record data.
Output parameter	oldxactid	text	Previous transaction ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	partitionoid	text	Partition OID.
Output parameter	tablespace	text	Tablespace.
Output parameter	alreadyread_bytes	text	Length of the read undo record.
Output parameter	prev_undorec_len	text	Length of the previous undo record.
Output parameter	td_id	text	ID of the transaction directory.
Output parameter	reserved	text	Reserved flag of the source version tuple stored in an undo record.
Output parameter	flag	text	Status flag of the source version tuple stored in an undo record.
Output parameter	flag2	text	Number of source version tuple columns stored in an undo record.
Output parameter	t_hoff	text	Length of the undo record data header.

Example (If the undo record is not recycled, the input parameter undoptr can be queried after being converted to a decimal value using the output parameter **end_undoptr** of the gs_undo_translot_dump_slot function.):

```
gaussdb=# SELECT * FROM gs_undo_dump_record('0000000000000042');
 undoptr |   xactid   | cid | reloid | relfilenode | utype |  blkprev   | blockno | uoffset | prevurp |
payloadlen | oldxactid | partitionoid | tablespace | alreadyread_byt
es | prev_undorec_len | td_id | reserved | flag | flag2 | t_hoff
---------+------------+-----+--------+-------------+-------+------------+---------+---------+---------
+------------+-----------+--------------+------------+----------------
---+------------------+-------+----------+------+-------+--------
      42 | 1073807360 | 0   | 0      | 108986369   | 0     | 1024786474 | 0       | 0       | 0       |
0          | 0            | 0          | 36
   | 16390            | -1    | -1       | -1   | -1    | -1
(1 row)
```

- gs_undo_dump_xid(undo_xid xid)

  Description: Parses undo records based on the XID.

  Return type: record

NOTE

When rollback segments are stored in files in page storage mode, the following format is used.

Table 1-157 gs_undo_dump_xid parameters

Category	Parameter	Type	Description
Input parameter	undo_xid	xid	Transaction ID.
Output parameter	undoptr	xid	Start position of the undo record to be parsed in the current undo zone.
Output parameter	xactid	xid	Transaction ID.
Output parameter	cid	text	Command ID.
Output parameter	reloid	text	Relation OID.
Output parameter	relfilenode	text	Relfinode of the file.
Output parameter	utype	text	Undo record type.
Output parameter	blkprev	text	Position of the previous undo record in the same block.
Output parameter	blockno	text	Block number.
Output parameter	uoffset	text	Undo record offset.
Output parameter	prevurp	text	Position of the previous undo record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	payloadlen	text	Length of the undo record data.
Output parameter	oldxactid	text	Previous transaction ID.
Output parameter	partitionoid	text	Partition OID.
Output parameter	tablespace	text	Tablespace.
Output parameter	alreadyread_bytes	text	Length of the read undo record.
Output parameter	prev_undorec_len	text	Length of the previous undo record.
Output parameter	td_id	text	ID of the transaction directory.
Output parameter	reserved	text	Reserved flag of the source version tuple stored in an undo record.
Output parameter	flag	text	Status flag of the source version tuple stored in an undo record.
Output parameter	flag2	text	Number of source version tuple columns stored in an undo record.
Output parameter	t_hoff	text	Length of the undo record data header.

Example:

```
gaussdb=# SELECT * FROM gs_undo_dump_xid('15779');
 undoptr | xactid | cid | reloid | relfilenode | utype | blkprev | blockno | uoffset | prevurp | payloadlen |
oldxactid | partitionoid | tablespace | alreadyread_bytes | pr
ev_undorec_len | td_id | reserved | flag | flag2 | t_hoff
---------+--------+-----+--------+-------------+-------+---------+---------+---------+---------+------------
+-----------+--------------+------------+-------------------+---
---------------+-------+----------+------+-------+--------
   108 |  15779 | 0   | 16767  | 16767       | 64    | 0       | 0       | 3       | 0       | 0         | 0         |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
0          | 1663      | 40              | 42
           | -1   | -1      | -1   | -1   | -1
(1 row)
```

- gs_verify_undo_record(type, start_idx, end_idx, location)

  Description: Verifies undo records. Currently, only the disk verification mode is supported. Offline verification can be performed only when services are not running. Before the verification, you need to manually perform checkpoint flushing.

  Return type: record

  NOTE

  When rollback segments are stored in files in page storage mode, the following format is used.

Table 1-158 gs_verify_undo_record parameters

Category	Parameter	Type	Description
Input parameter	type	text	Verification type. ● **'urp'**: verifies all undo records in a specified URP range. ● **'zone'**: verifies all undo records of all zones in a specified zone range.
Input parameter	start_idx	int8	Start position. ● When **type** is set to **'urp'**, this parameter indicates the start position of undo records. ● When **type** is set to **'zone'**, this parameter indicates the start undo zone ID.
Input parameter	end_idx	int8	End position. ● When **type** is set to **'urp'**, this parameter indicates the end position of undo records. ● When **type** is set to **'zone'**, this parameter indicates the end undo zone ID.
Input parameter	location	bool	● **0**: memory verification ● **1**: disk verification Currently, this parameter can only be set to **1**.
Output parameter	zone_id	int8	Undo zone ID.

Category	Parameter	Type	Description
Output parameter	detail	text	Verification error information.

Example 1: Verify the undo record whose URP is 24.

```
gaussdb=# SELECT * FROM gs_verify_undo_record('urp', 24, 24, 1);
zone_id | detail
--------+--------
(0 rows)
```

Example 2: Verify all undo records from zone 0 to zone 2 on the disk.

```
gaussdb=# SELECT * FROM gs_verify_undo_record('zone', 0, 2, 1);
zone_id | detail
--------+--------
(0 rows)
```

NOTE

If an error is reported when this function is called, contact Huawei technical support.

- gs_verify_undo_slot(type, start_idx, end_idx, location)

  Description: Verifies undo transaction slots. Currently, only the disk verification mode is supported. Offline verification can be performed only when services are not running. Before the verification, you need to manually perform checkpoint flushing.

  Return type: record

  NOTE

  When rollback segments are stored in files in page storage mode, the following format is used.

  Table 1-159 gs_verify_undo_slot parameters

Category	Parameter	Type	Description
Input parameter	type	text	Verification type. • **'zone'**: verifies all transaction slots of all zones in a specified zone range.
Input parameter	start_idx	int8	Start undo zone ID.
Input parameter	end_idx	int8	End undo zone ID.

Category	Parameter	Type	Description
Input parameter	location	bool	• **0**: memory verification • **1**: disk verification Currently, this parameter can only be set to **1**.
Output parameter	zone_id	int8	Undo zone ID.
Output parameter	detail	text	Verification error information.

Example: Verify all transaction slot records from zone 0 to zone 2 on the disk.

```
gaussdb=# SELECT * FROM gs_verify_undo_slot('zone', 0, 2, 1);
zone_id | detail
--------+--------
(0 rows)
```

NOTE

If an error is reported when this function is called, contact Huawei technical support.

- gs_verify_undo_meta(type, start_idx, end_idx, location)

  Description: Verifies undo metadata. Currently, only the disk verification mode is supported. Offline verification can be performed only when services are not running. Before the verification, you need to manually perform checkpoint flushing.

  Return type: record

  NOTE

  This system function supports only the rollback segments in page storage mode.

Table 1-160 gs_verify_undo_meta parameters

Category	Parameter	Type	Description
Input parameter	type	text	Verification type. The value of **type** can only be **'all'**, indicating that all meta information of all zones in the specified zone range is verified.
Input parameter	start_idx	int64	Start undo zone ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Input parameter	end_idx	int64	End undo zone ID.
Input parameter	location	bool	• **0**: memory verification • **1**: disk verification Currently, this parameter can only be set to **1**.
Output parameter	zone_id	int64	Undo zone ID.
Output parameter	detail	text	Verification error information.

Example: Verify all meta information records from zone 0 to zone 2 on the disk.

```
gaussdb=# SELECT * FROM gs_verify_undo_meta('all', 0, 2, 1);
zone_id | detail
---------+---------
(0 rows)
```

NOTE

If an error is reported when this function is called, contact Huawei technical support.

- gs_async_rollback_worker_status()

  Description: Monitors the status of active asynchronous rollback threads.

  Return type: record

Table 1-161 gs_async_rollback_worker_status parameters

Category	Parameter	Type	Description
Output parameter	datid	oid	Database ID.
Output parameter	pid	int8	Thread ID.
Output parameter	sessionid	int8	Session ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	usesysid	oid	ID of the user who initiates the thread.
Output parameter	state	int	Thread status. • **0**: undefined • **1**: idle • **2**: running
Output parameter	rollback_start_time	timestamptz	Timestamp when a thread is started.
Output parameter	idx	oid	Index of an asynchronous rollback thread in the array.
Output parameter	xid	xid	XID of the transaction that is being rolled back.
Output parameter	progress	text	Rollback progress of the transaction (number of undo records that have been rolled back/total number of undo records, in percentage).

- gs_async_rollback_xact_status()

  Description: Monitors the hash table of asynchronous rollback tasks.

  Return type: record

  Table 1-162 gs_async_rollback_xact_status parameters

Category	Parameter	Type	Description
Output parameter	xid	xid	XID of the transaction that requires asynchronous rollback.
Output parameter	start_undoptr	xid	Pointer to the start undo record of the transaction.
Output parameter	end_undoptr	xid	Pointer to the end undo record of the transaction.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	dbid	uint32	ID of the database where the transaction is located.
Output parameter	slot_ptr	xid	Pointer to the transaction slot corresponding to the transaction.
Output parameter	launched	bool	Checks whether there is an active asynchronous rollback thread.

- gs_undo_recycler_status()

  Description: Monitors the status of asynchronous recycling threads.

  Return type: record

  Table 1-163 gs_undo_recycler_status parameters

Category	Parameter	Type	Description
Output parameter	datid	oid	Database ID.
Output parameter	pid	int8	Thread ID.
Output parameter	sessionid	int8	Session ID.
Output parameter	usesysid	oid	ID of the user who initiates the thread.
Output parameter	state	int	Thread status. • **0**: undefined • **1**: idle • **2**: running
Output parameter	backend_start	timestamptz	Timestamp when a thread is started.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	total_recycle_time	xid	Total recycling time.
Output parameter	max_recycle_time	xid	Maximum recycling time.
Output parameter	total_recycle_size	xid	Total recycled space.
Output parameter	total_recycle_count	xid	Total number of recycling times.
Output parameter	recycle_sleep_count	xid	Number of sleep times.
Output parameter	recycle_sleep_time	xid	Total sleep time.
Output parameter	max_recycle_sleep_time	xid	Maximum sleep time.
Output parameter	last_recycle_timestamp	timestamptz	Timestamp of the last successful recycling.
Output parameter	last_update_global_recycle_xid_timestamp	timestamptz	Timestamp of the last global recycling transaction.

- gs_undo_launcher_status()

  Description: Monitors the status of asynchronous rollback launcher threads.

  Return type: record

  Table 1-164 gs_undo_launcher_status parameters

Category	Parameter	Type	Description
Output parameter	datid	oid	Database ID.
Output parameter	pid	int64	Thread ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	sessionid	int64	Session ID.
Output parameter	usesysid	oid	ID of the user who initiates the thread.
Output parameter	state	int32	Thread status. • **0**: undefined • **1**: idle • **2**: running
Output parameter	backend_start	timestamptz	Timestamp when a thread is started.
Output parameter	total_async_rollback_task_count	uint64	Total number of asynchronous rollback tasks initiated after the database on the local node is started.
Output parameter	average_async_rollback_time	uint64	Average duration of an asynchronous rollback task.
Output parameter	max_async_rollback_time	uint64	Maximum duration of an asynchronous rollback task.
Output parameter	min_async_rollback_time	uint64	Minimum duration of an asynchronous rollback task.

1.6.27.14 HTAP System Functions

- gs_global_imcv_info()

  Description: Displays basic information about the IMCV table in the current database, such as the loading status of column-store data and memory or disk space usage. In a distributed system, this function can be executed only on CNs. The basic information about the IMCV tables on all DNs is returned.

  Return type: record

  NOTICE

  - This function is available only when **enable_htap** is set to **on**.
  - Only a system administrator has the permission to execute this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-165 gs_global_imcv_info parameters

Parameter	Parameter	Type	Description
Output parameter	node_name	text	Node name.
Output parameter	rel_name	name	Table name.
Output parameter	partition_name	name	Partition name.
Output parameter	subpartition_name	name	Subpartition name.
Output parameter	schema_name	name	Schema name.
Output parameter	user_name	name	Username.
Output parameter	imcvnattr	smallint	Number of loaded columns.
Output parameter	imcvkey	int2vector	Array of loaded columns.
Output parameter	priority	smallint	Loading priority. (Currently, the priority does not take effect.) ● **'1'**: PRIORITY_NONE (no priority). ● **'2'**: PRIORITY_LOW (low priority). ● **'3'**: PRIORITY_HIGH (high priority).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Parameter	Type	Description
Output parameter	rowgroup_num	integer	Expected number of row groups (that is, the number of row groups after full loading).
Output parameter	populated_rowgroup_num	integer	Number of row groups that are actually loaded. (The value should be equal to that of **rowgroup_num** after full loading.)
Output parameter	cu_num_in_mem	integer	Number of IMCUs in the memory.
Output parameter	cu_mem_size	bigint	Memory occupied by IMCU, in bytes.
Output parameter	cu_num_in_disk	integer	Number of IMCUs in the disk.
Output parameter	cu_disk_size	bigint	Disk size occupied by IMCU, in bytes.

Examples

```
htapdb=# SELECT * FROM gs_global_imcv_info();
  node_name   |         rel_name         |       partition_name        | subpartition_name | schema_name |
user_name | imcvnattr |   imcvkey   | priority | rowgroup_num | populated_rowgroup_num |
cu_num_in_mem | cu_mem_size | cu_num_in_disk | cu_disk_size
--------------+--------------------------+-----------------------------+-------------------+-------------+-----------
+-----------+-------------+----------+--------------+-----------------------+--------------+-------------
+----------------+--------------
 dn_6003_6004 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p1 |                   | public      |
***       |         6 | 1 2 3 4 5 6 |        1 |            1 |                     1 |             7 |       24228 |             0
|             0
 dn_6003_6004 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      |
***       |         6 | 1 2 3 4 5 6 |        1 |            1 |                     1 |             7 |        8048 |             0
|             0
 dn_6003_6004 | htap_flush_by_name_part1 |                             |                   | public      | ***
|         6 | 1 2 3 4 5 6 |        1 |            0 |                     0 |             0 |             0 |             0 |             0
 dn_6001_6002 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p1 |                   | public      |
***       |         6 | 1 2 3 4 5 6 |        1 |            0 |                     0 |             0 |             0 |             0
|             0
 dn_6001_6002 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      |
***       |         6 | 1 2 3 4 5 6 |        1 |            1 |                     1 |             7 |        8048 |             0
|             0
 dn_6001_6002 | htap_flush_by_name_part1 |                             |                   | public      | ***
|         6 | 1 2 3 4 5 6 |        1 |            0 |                     0 |             0 |             0 |             0 |             0
(6 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_htap_imcv_info()

  Description: Displays basic information about the IMCV table in the current database, such as the loading status of column-store data and memory or disk space usage. In a distributed system, this function returns information about the current node. To query information about all nodes, use the gs_global_imcv_info() function.

  Return type: record

  NOTICE

  - This function is available only when **enable_htap** is set to **on**.
  - Only SYSADMIN or OPRADMIN can execute this function.

  Table 1-166 gs_htap_imcv_info parameters

Parameter	Parameter	Type	Description
Output parameter	relname	name	Table name.
Output parameter	partition_name	name	Partition name.
Output parameter	subpartition_name	name	Subpartition name.
Output parameter	schema_name	name	Schema name.
Output parameter	user_name	name	Username.
Output parameter	imcvnattr	smallint	Number of loaded columns.
Output parameter	imcvkey	int2vector	Array of loaded columns.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Parameter	Type	Description
Output parameter	priority	smallint	Loading priority. (Currently, the priority does not take effect.) • **'1'**: PRIORITY_NONE (no priority). • **'2'**: PRIORITY_LOW (low priority). • **'3'**: PRIORITY_HIGH (high priority).
Output parameter	rowgroup_num	integer	Expected number of row groups (that is, the number of row groups after full loading).
Output parameter	populated_rowgroup_num	integer	Number of row groups that are actually loaded. (The value should be equal to that of **rowgroup_num** after full loading.)
Output parameter	cu_num_in_mem	integer	Number of IMCUs in the memory.
Output parameter	cu_mem_size	bigint	Memory occupied by IMCU, in bytes.
Output parameter	cu_num_in_disk	integer	Number of IMCUs in the disk.
Output parameter	cu_disk_size	bigint	Disk size occupied by IMCU, in bytes.

Example:

```
htapdb=# SELECT * FROM gs_htap_imcv_info();
         rel_name         |       partition_name        | subpartition_name | schema_name | user_name |
imcvnattr |   imcvkey   | priority | rowgroup_num | populated_rowgroup_num | cu_num_in_mem |
cu_mem_size | cu_num_in_disk | cu_disk_size
--------------------------+-----------------------------+-------------------+-------------+-----------+-----------
+-------------+----------+--------------+------------------------+---------------+-------------+----------------
+---------------
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p1 |                   | public      | ***       |         6
| 1 2 3 4 5 6 |        1 |            0 |                      0 |             0 |           0 |             0 |           0
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      | ***       |         6
| 1 2 3 4 5 6 |        1 |            0 |                      0 |             0 |           0 |             0 |           0
 htap_flush_by_name_part1 |                             |                   | public      | ***       |         6 | 1 2 3 4 5 6
|        1 |            0 |                      0 |             0 |           0 |             0 |           0
(3 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_imcv_flush(reloid oid, partoid oid, parallelism integer, rebuild_mode integer)

  Description: Immediately rebuilds a specified IMCV table or the partition in the database and displays the rebuild result. During the rebuild, the IMCV table is refreshed, old data is removed, and inactive data in the current table is synchronized. When this function is executed on a CN, the IMCV of the table on all nodes is rebuilt. When this function is executed on a DN, the IMCV of the table on the DN is rebuilt.

  Return type: record

  NOTICE

  - This function is available only when **enable_htap** is set to **on**.
  - Only SYSADMIN or OPRADMIN can execute this function.
  - This function cannot be executed in a transaction block.
  - During parallel loading or rebuilding, each worker occupies extra resources, such as memory, CPU, and I/O (flushing is enabled). Therefore, exercise caution when controlling the degree of parallelism for loading or rebuilding based on the actual hardware resources and system load.

  Table 1-167 gs_imcv_flush parameters

Category	Parameter	Type	Description
Input parameter	reloid	oid	Table OID.
Input parameter	partoid	oid	Partition OID. When all partitions of a partitioned table or a non-partitioned table are queried, the value can be **0**.
Input parameter	parallelism	integer	Degree of parallelism, that is, the number of bgworkers used by a loading or rebuilding task. The value ranges from 1 to 256. If the actual number of bgworkers in the background is insufficient, the bgworkers that can be started are used to execute the rebuilding task.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Input parameter	rebuild_mode	integer	Rebuilding mode. Value range: ● **0**: non-forcible rebuilding. Rebuilding is triggered only when column-store data is not synchronized to row-store data. ● **1**: forcible rebuilding. Rebuilding is triggered regardless of whether the column-store data is the latest.
Output parameter	total	integer	Number of row groups that are actually processed.
Output parameter	success	integer	Number of row groups that are successfully processed.
Output parameter	error	integer	Number of row groups that fail to be processed. For example, the loading or rebuilding operation is not performed due to insufficient memory.
Output parameter	skip	integer	Number of row groups that are skipped. For example, the data is the latest, and the rebuilding operation is not performed.

Example:

```
htapdb=# SELECT * FROM gs_imcv_flush(20834, 20838,1,1);
 total | success | error | skip
-------+---------+-------+------
 1     | 1       | 0     | 0
(1 row)
```

- gs_imcv_flush(relname text, partname text, parallelism integer, rebuild_mode integer)

  Description: Immediately rebuilds a specified IMCV table or the partition in the database and displays the rebuild result. During the rebuild, the IMCV table is refreshed, old data is removed, and inactive data in the current table is synchronized. When this function is executed on a CN, the IMCV of the table on all nodes is rebuilt. When this function is executed on a DN, the IMCV of the table on the DN is rebuilt.

  Return type: record

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- This function is available only when **enable_htap** is set to **on**.
- Only SYSADMIN or OPRADMIN can execute this function.
- This function cannot be executed in a transaction block.
- During parallel loading or rebuilding, each worker occupies extra resources, such as memory, CPU, and I/O (flushing is enabled). Therefore, exercise caution when controlling the degree of parallelism for loading or rebuilding based on the actual hardware resources and system load.

Table 1-168 gs_imcv_flush parameters

Category	Parameter	Type	Description
Input parameter	relname	text	Table name.
Input parameter	partname	text	Partition name. When all partitions of a partitioned table or a non-partitioned table are queried, the value can be **NULL** or ''. The default value can be used. In this case, if the table contains partitions, all partitions are processed.
Input parameter	parallelism	integer	Degree of parallelism, that is, the number of bgworkers used by a loading or rebuilding task. The value ranges from 1 to 256. If the actual number of bgworkers in the background is insufficient, the bgworkers that can be started are used to execute the rebuilding task. The default value can be used. In this case, the degree of parallelism is **1**, that is, no parallelism is performed.
Input parameter	rebuild_mode	integer	Rebuilding mode. Value range: • **0**: non-forcible rebuilding. Rebuilding is triggered only when column-store data is not synchronized to row-store data. • **1**: forcible rebuilding. Rebuilding is triggered regardless of whether column-store data is fresh. The default value can be used, indicating non-forcible rebuilding.
Output parameter	total	integer	Number of row groups that are actually processed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	success	integer	Number of row groups that are successfully processed.
Output parameter	error	integer	Number of row groups that fail to be processed. For example, the loading or rebuilding operation is not performed due to insufficient memory.
Output parameter	skip	integer	Number of row groups that are skipped. For example, the data is the latest, and the rebuilding operation is not performed.

Example:

```
-- Set partname, parallelism, and rebuild_mode to their default values.
htapdb=# SELECT * FROM gs_imcv_flush('htap_flush_by_name_part1');
 total | success | error | skip
-------+---------+-------+------
 2     | 2       | 0     | 0
(1 row)
-- Set parallelism and rebuild_mode to their default values.
htapdb=# SELECT * FROM gs_imcv_flush('htap_flush_by_name_part1',
'htap_flush_by_name_part1_p1');
 total | success | error | skip
-------+---------+-------+------
 1     | 0       | 0     | 1
(1 row)
-- Set rebuild_mode to the default value.
htapdb=# SELECT * FROM gs_imcv_flush('htap_flush_by_name_part1',
'htap_flush_by_name_part1_p1',1);
 total | success | error | skip
-------+---------+-------+------
 1     | 0       | 0     | 1
(1 row)
htapdb=# SELECT * FROM gs_imcv_flush('htap_flush_by_name_part1',
'htap_flush_by_name_part1_p1',1,1);
 total | success | error | skip
-------+---------+-------+------
 1     | 1       | 0     | 0
(1 row)
```

- gs_global_htap_tmu_data(relname text, partname text, rowgroup_id integer)

  Description: Views active transaction statistics in the real-time transaction unit TMU. This function can be executed only on a distributed CN and returns information about all DNs. If **rowgroup_id** is set to **0**, all active transaction statistics in the table are returned. If **rowgroup_id** is set to a value greater than 0, the active transaction statistics in the specified row group in the table are returned.

  Return type: record

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- This function is available only when **enable_htap** is set to **on**.
- Only a system administrator has the permission to execute this function.

Table 1-169 gs_global_htap_tmu_data parameters

Category	Parameter	Type	Description
Input parameter	relname	text	Table name. If you want to query cross-schema table information, you can set the value in the '*schema.relname*' format.
Input parameter	partname	text	Partition name. When all partitions of a partitioned table or a non-partitioned table are queried, the value can be **NULL** or **''**.
Input parameter	rowgroup_id	integer	Identifies a row group.
Output parameter	node_name	text	Node name.
Output parameter	rel_name	name	Table name.
Output parameter	partition_name	name	Name of a partition. If it is not a partition, this column is left empty.
Output parameter	subpartition_name	name	Name of a subpartition. If it is not a subpartition, this column is left empty.
Output parameter	schema_name	name	Schema name.
Output parameter	rel_oid	oid	Table OID.
Output parameter	rowgroup_id	integer	Identifies a row group.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	page_id	integer	Page ID in the row group being searched.
Output parameter	flag	smallint	The value **0** indicates a non-dirty page, and the value **1** indicates a dirty page.
Output parameter	xid	xid	ID of the transaction that operates the data.

Example:

```
htapdb=# SELECT * FROM gs_global_htap_tmu_data('htap_flush_by_name_part1',
'htap_flush_by_name_part1_p1', 0);
 node_name |          rel_name          |       partition_name        | subpartition_name | schema_name |
rel_oid | rowgroup_id | page_id | flag |  xid
-----------+----------------------------+-----------------------------+-------------------+-------------
+-------------+---------+------+--------
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p1 |                   | public      |
16389 |           1 |       0 |    0 | 203236
(1 row)
```

- gs_htap_tmu_data(relname text, partname text, rowgroup_id integer)

  Description: Views active transaction statistics in the real-time transaction unit TMU and returns active transaction statistics in the table on the current node. If **rowgroup_id** is set to **0**, all active transaction statistics in the table are returned. If **rowgroup_id** is set to a value greater than 0, the active transaction statistics in the specified row group in the table are returned.

  Return type: record

NOTICE

- This function is available only when **enable_htap** is set to **on**.
- Only SYSADMIN or OPRADMIN can execute this function.

Table 1-170 gs_htap_tmu_data parameters

Category	Parameter	Type	Description
Input parameter	relname	text	Table name. If you want to query cross-schema table information, you can set the value in the '*schema.relname*' format.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Input parameter	partname	text	Partition name. When all partitions of a partitioned table or a non-partitioned table are queried, the value can be **NULL** or ''.
Input parameter	rowgroup_id	integer	Identifies a row group.
Output parameter	rel_name	name	Table name.
Output parameter	partition_name	name	Name of a partition. If it is not a partition, this column is left empty.
Output parameter	subpartition_name	name	Name of a subpartition. If it is not a subpartition, this column is left empty.
Output parameter	schema_name	name	Schema name.
Output parameter	rel_oid	oid	Table OID.
Output parameter	rowgroup_id	integer	Identifies a row group.
Output parameter	page_id	integer	Page ID in the row group being searched.
Output parameter	flag	smallint	The value **0** indicates a non-dirty page, and the value **1** indicates a dirty page.
Output parameter	xid	xid	ID of the transaction that operates the data.

Example:

```
htapdb=# SELECT * FROM gs_htap_tmu_data('htap_flush_by_name_part1',
'htap_flush_by_name_part1_p2',0);
        rel_name         |       partition_name        | subpartition_name | schema_name | rel_oid |
rowgroup_id | page_id | flag | xid
-------------------------+-----------------------------+-------------------+-------------+---------+-------------
+---------+------+-----
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      |   16390 |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
1 |     0 |   0 |  0
(1 row)
```

- gs_global_htap_tmu_chunk_meta(relname name, partname name, rowgroup_id integer)

  Description: Views the metadata of the real-time transaction unit TMU. This function can be executed only on a distributed CN and returns information about all DNs. If **rowgroup_id** is set to **0**, the real-time transaction unit TMU data of all row groups is returned. If **rowgroup_id** is greater than 0, the real-time transaction unit TMU data of the specified row group in the table is returned. If **rowgroup_id** is set to **-1**, the real-time transaction unit TMU data of all row groups is collected and returned as a single record.

  Return type: record

NOTICE

- This function is available only when **enable_htap** is set to **on**.
- Only a system administrator has the permission to execute this function.

Table 1-171 gs_global_htap_tmu_chunk_meta parameters

Category	Parameter	Type	Description
Input parameter	relname	text	Table name. If you want to query cross-schema table information, you can set the value in the '*schema.relname*' format.
Input parameter	partname	text	Partition name. When all partitions of a partitioned table or a non-partitioned table are queried, the value can be **NULL** or ''.
Input parameter	rowgroup_id	integer	Identifies a row group.
Output parameter	node_name	text	Database node name.
Output parameter	rel_name	name	Table name.
Output parameter	partition_name	name	Name of a partition. If it is not a partition, this column is left empty.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	subpartition_name	name	Name of a subpartition. If it is not a subpartition, this column is left empty.
Output parameter	schema_name	name	Schema name.
Output parameter	rel_oid	oid	Table OID.
Output parameter	rowgroup_id	integer	Identifies a row group.
Output parameter	page_count	bigint unsigned	Number of pages in the row group being searched.
Output parameter	dirty_page_count	bigint unsigned	Number of dirty pages in the row group being searched.
Output parameter	clean_page_count	bigint unsigned	Number of non-dirty pages in the row group being searched.
Output parameter	dirty_ratio	real	Ratio of dirty pages to all pages in the row group being searched.

Example:

```
htapdb=# SELECT * FROM gs_global_htap_tmu_chunk_meta('htap_flush_by_name_part1',
'htap_flush_by_name_part1_p2', 0);
 node_name |         rel_name         |       partition_name        | subpartition_name | schema_name |
rel_oid | rowgroup_id | page_count | dirty_page_count | clean_page_count | dirty_ratio
-----------+--------------------------+-----------------------------+-------------------+-------------+---------
+-------------+------------+------------------+------------------+-------------
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      |
16390 |           1 |          1 |                0 |                1 |           0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      |
16390 |           1 |          1 |                0 |                1 |           0
(2 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_htap_tmu_chunk_meta(relname text, partname text, rowgroup_id integer)

  Description: Views the metadata of the real-time transaction unit TMU. In a distributed system, information about the current node is returned. If **rowgroup_id** is set to **0**, the real-time transaction unit TMU data of all row groups is returned. If **rowgroup_id** is greater than 0, the real-time transaction unit TMU data of the specified row group in the table is returned. If **rowgroup_id** is set to **-1**, the real-time transaction unit TMU data of all row groups is collected and returned as a single record.

  Return type: record

  NOTICE

  - This function is available only when **enable_htap** is set to **on**.
  - Only SYSADMIN or OPRADMIN can execute this function.

  Table 1-172 gs_htap_tmu_chunk_meta parameters

Category	Parameter	Type	Description
Input parameter	relname	text	Table name. If you want to query cross-schema table information, you can set the value in the '*schema.relname*' format.
Input parameter	partname	text	Partition name. When all partitions of a partitioned table or a non-partitioned table are queried, the value can be **NULL** or ''.
Input parameter	rowgroup_id	integer	Identifies a row group.
Output parameter	rel_name	name	Table name.
Output parameter	partition_name	name	Name of a partition. If it is not a partition, this column is left empty.
Output parameter	subpartition_name	name	Name of a subpartition. If it is not a subpartition, this column is left empty.
Output parameter	schema_name	name	Schema name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	rel_oid	oid	Table OID.
Output parameter	rowgroup_id	integer	Identifies a row group.
Output parameter	page_count	bigint unsigned	Number of pages in the row group being searched.
Output parameter	dirty_page_count	bigint unsigned	Number of dirty pages in the row group being searched.
Output parameter	clean_page_count	bigint unsigned	Number of non-dirty pages in the row group being searched.
Output parameter	dirty_ratio	real	Ratio of dirty pages to all pages in the row group being searched.

Example:

```
htapdb=# SELECT * FROM gs_htap_tmu_chunk_meta('htap_flush_by_name_part1',
'htap_flush_by_name_part1_p2', 0);
         rel_name          |       partition_name        | subpartition_name | schema_name | rel_oid |
rowgroup_id | page_count | dirty_page_count | clean_page_count | dirty_ratio
--------------------------+-----------------------------+-------------------+-------------+---------+-------------
+------------+------------------+------------------+-------------
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      |   16390 |
1 | 1          | 0                | 1                |                0
(1 row)
```

- gs_global_imcv_bgworker_status()

  Description: Displays the status of the bgworker that rebuilds the IMCV table. In a distributed system, this function can be executed only on CNs. The information about all DNs is returned.

  Return type: record

NOTICE

- This function is available only when **enable_htap** is set to **on**.
- Only a system administrator has the permission to execute this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-173 gs_global_imcv_bgworker_status parameters

Category	Parameter	Type	Description
Output parameter	node_name	text	Database node name.
Output parameter	worker_pid	bigint	Thread ID.
Output parameter	database_id	oid	Database ID.
Output parameter	relation_id	oid	OID of the IMCV table that is being processed.
Output parameter	rowgroup_id	integer	Identifies a row group.
Output parameter	worker_status	text	Specifies the working state of a thread, which can be **init** (initialized), **running** (in progress), or **ready** (idle).
Output parameter	worker_type	text	Task type. The options are as follows: • **background_invalid_worker**: No task is being executed. • **background_unknown_worker**: An unknown task is executed. • **background_rowgroup_rebuild_worker**: A row group is rebuilt. • **background_async_populate_worker**: Asynchronous loading is performed. • **background_cache_worker**: A task in the cache is executed.
Output parameter	start_time	timestamp with time zone	Timestamp when a thread is started.

Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
htapdb=# SELECT * FROM gs_global_imcv_bgworker_status();
 node_name |    worker_pid     | database_id | relation_id | rowgroup_id | worker_status |
worker_type         |           start_time
-----------+-------------------+-------------+-------------+-------------+---------------+--------------------------
+-------------------------------
 datanode1 | 140067573069568 |       16384 |           0 |           0 | ready         |
background_invalid_worker | 2025-12-02 21:10:16.282394+08
 datanode1 | 140067645421312 |       16384 |           0 |           0 | ready         |
background_invalid_worker | 2025-12-02 21:10:12.267694+08
 datanode1 | 140067608721152 |           0 |           0 |           0 | ready         |
background_invalid_worker | 2025-12-02 21:10:12.332674+08
 datanode2 | 139803070822144 |       16384 |           0 |           0 | ready         |
background_invalid_worker | 2025-12-02 21:10:16.274911+08
 datanode2 | 139803270051584 |       16384 |           0 |           0 | ready         |
background_invalid_worker | 2025-12-02 21:10:16.284832+08
 datanode2 | 139803153659648 |           0 |           0 |           0 | ready         |
background_invalid_worker | 2025-12-02 21:10:12.374876+08
(6 rows)
```

- gs_imcv_bgworker_status()

  Description: Displays the status of the bgworker that rebuilds the IMCV table. In a distributed system, the information about the current node is returned.

  Return type: record

NOTICE

- This function is available only when **enable_htap** is set to **on**.
- Only SYSADMIN or OPRADMIN can execute this function.

Table 1-174 gs_imcv_bgworker_status parameters

Category	Parameter	Type	Description
Output parameter	worker_pid	bigint	Thread ID.
Output parameter	database_id	oid	Database ID.
Output parameter	relation_id	oid	OID of the IMCV table that is being processed.
Output parameter	rowgroup_id	integer	Identifies a row group.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	worker_status	cstring	Specifies the working state of a thread, which can be **init** (initialized), **running** (in progress), or **ready** (idle).
Output parameter	worker_type	cstring	Task type. The value can be **background_invalid_worker** (no task is being executed), **background_unknown_worker** (unknown task), **background_rowgroup_rebuild_worker** (rebuilding a row group), **background_async_populate_worker** (asynchronous loading), or **background_cache_worker** (cache task).
Output parameter	start_time	timestamp with time zone	Timestamp when a thread is started.

Example:

```
htapdb=# SELECT * FROM gs_imcv_bgworker_status();
   worker_pid    | database_id | relation_id | rowgroup_id | worker_status |        worker_type
|          start_time
-----------------+-------------+-------------+-------------+---------------+---------------------------
+-------------------------------
 140067573069568 |       16384 |           0 |           0 | ready         | background_invalid_worker |
2025-12-02 21:10:16.282394+08
 140067645421312 |       16384 |           0 |           0 | ready         | background_invalid_worker |
2025-12-02 21:10:12.267694+08
 140067608721152 |           0 |           0 |           0 | ready         | background_invalid_worker |
2025-12-02 21:10:12.332674+08
(3 rows)
```

- gs_global_imcu_meta(rel_name text, partname text)

  Description: Displays the IMCU metadata in the IMCV table with the specified **rel_name** in the current database. In a distributed system, this function can be executed only on CNs. The information about all DNs is returned.

  Return type: record

  NOTICE

  - This function is available only when **enable_htap** is set to **on**.
  - Only a system administrator has the permission to execute this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-175 gs_global_imcu_meta parameters

Category	Parameter	Type	Description
Input parameter	relname	text	IMCV table name. If you want to query cross-schema table information, you can set the value in the '*schema.relname*' format.
Input parameter	partname	text	IMCV partition name. When all partitions of a partitioned table or a non-partitioned table are queried, the value can be **NULL** or ''.
Output parameter	node_name	text	Database node name.
Output parameter	rel_name	name	IMCV table name.
Output parameter	partition_name	name	IMCV partition name.
Output parameter	subpartition_name	name	IMCV subpartition name.
Output parameter	schema_name	name	Schema name.
Output parameter	rowgroup_id	integer	ID of the row group where the IMCU is located.
Output parameter	row_count	integer	Number of rows of the IMCU.
Output parameter	imcu_size	integer	IMCU size, in bytes.
Output parameter	imcu_data_size	integer	IMCU data size (excluding IMCU metadata), in bytes.
Output parameter	compressed_data_size	integer	IMCU data size after compression.
Output parameter	compressed_ratio	real	Compression ratio of the IMCU data.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	compressed_type	text	Compression algorithm of the IMCU. • **none**: There is no IMCU data and the data cannot be compressed. • **uncompressed**: Data is not compressed. • **delta_plus_rle**: The IMCU data is compressed using the delta_plus_rle compression algorithm. • **lz4_compressed**: The IMCU data is compressed using the lz4 compression algorithm. • **dict_compressed**: The IMCU data is compressed using the dict compression algorithm.
Output parameter	imcu_mode	integer	Characteristics of column data loaded by the IMCU: • **1**: common. • **2**: all empty. • **3**: same value. • **4**: no maximum or minimum value. • **8**: contains a blank line. • **16**: no sum value.
Output parameter	magic	integer	XID value of the transaction when IMCU data is generated. This value is used to check whether the IMCU data is valid.
Output parameter	attnum	smallint	attnum corresponding to the column loaded by IMCU.
Output parameter	scan_cnt	bigint	Total number of times that the IMCU is scanned.
Output parameter	hit_cnt	bigint	Number of times that the memory-state IMCU is hit.
Output parameter	data_location	smallint	Specifies that the IMCU is in the memory, in the disk, or in escape state. • **0**: memory. • **1**: disk. • **2**: escape state. Only metadata is created due to insufficient memory. When this part of data is queried, it is read from the row store.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
htapdb=# SELECT * FROM gs_global_imcu_meta('htap_flush_by_name_part1',
'htap_flush_by_name_part1_p2');
 node_name |         rel_name         |       partition_name        | subpartition_name | schema_name |
rowgroup_id | row_count | imcu_size | imcu_data_size | compressed_data_size | compressed_ratio |
compressed_type | imcu_mode | magic | attnum | scan_cnt | hit_cnt | data_location
-----------+--------------------------+-----------------------------+-------------------+-------------+-------------
+-----------+-----------+----------------+----------------------+------------------+-----------------+-----------
+-------+--------+----------+---------+---------------
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |      4016 |            16 |                 16 |               1 | uncompressed    |         1 |
0 |      0 |        0 |       0 |             0
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |         0 |             0 |                  0 |               0 | none            |         3 |     0 |
1 |        0 |       0 |            0
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |         0 |             0 |                  0 |               0 | none            |         3 |     0 |
2 |        0 |       0 |            0
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |         0 |             0 |                  0 |               0 | none            |         3 |     0 |
3 |        0 |       0 |            0
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |         0 |             0 |                  0 |               0 | none            |         3 |     0 |
4 |        0 |       0 |            0
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |      4032 |            24 |                 24 |               1 | uncompressed    |         4 |
0 |      5 |        0 |       0 |             0
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |         0 |             0 |                  0 |               0 | none            |         3 |     0 |
6 |        0 |       0 |            0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |      4016 |            16 |                 16 |               1 | uncompressed    |         1 |
0 |      0 |        0 |       0 |             0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |         0 |             0 |                  0 |               0 | none            |         3 |     0 |
1 |        0 |       0 |            0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |         0 |             0 |                  0 |               0 | none            |         3 |     0 |
2 |        0 |       0 |            0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |         0 |             0 |                  0 |               0 | none            |         3 |     0 |
3 |        0 |       0 |            0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |         0 |             0 |                  0 |               0 | none            |         3 |     0 |
4 |        0 |       0 |            0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |      4032 |            24 |                 24 |               1 | uncompressed    |         4 |
0 |      5 |        0 |       0 |             0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|           1 |         1 |         0 |             0 |                  0 |               0 | none            |         3 |     0 |
6 |        0 |       0 |            0
(14 rows)
```

Note: In this example, each row represents a IMCU (the column whose **attnum** is **0** is the **CTID** column).

- gs_imcu_meta(rel_name text, partname text)

  Description: Displays the IMCU metadata in the IMCV table with the specified **rel_name** in the current database. In a distributed system, the information about the current node is returned.

  Return type: record

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- This function is available only when **enable_htap** is set to **on**.
- Only SYSADMIN or OPRADMIN can execute this function.

Table 1-176 gs_imcu_meta parameters

Category	Parameter	Type	Description
Input parameter	relname	text	IMCV table name. If you want to query cross-schema table information, you can set the value in the '*schema.relname*' format.
Input parameter	partname	text	IMCV partition name. When all partitions of a partitioned table or a non-partitioned table are queried, the value can be **NULL** or ''.
Output parameter	rel_name	name	IMCV table name.
Output parameter	partition_name	name	IMCV partition name.
Output parameter	subpartition_name	name	IMCV subpartition name.
Output parameter	schema_name	name	Schema name.
Output parameter	rowgroup_id	integer	ID of the row group where the IMCU is located.
Output parameter	row_count	integer	Number of rows of the IMCU.
Output parameter	imcu_size	integer	IMCU size, in bytes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	imcu_data_size	integer	IMCU data size (excluding IMCU metadata), in bytes.
Output parameter	compressed_data_size	integer	IMCU data size after compression.
Output parameter	compressed_ratio	real	Compression ratio of the IMCU data.
Output parameter	compressed_type	text	Compression algorithm of the IMCU. • **none**: There is no IMCU data and the data cannot be compressed. • **uncompressed**: Data is not compressed. • **delta_plus_rle**: The IMCU data is compressed using the delta_plus_rle compression algorithm. • **lz4_compressed**: The IMCU data is compressed using the lz4 compression algorithm. • **dict_compressed**: The IMCU data is compressed using the dict compression algorithm.
Output parameter	imcu_mode	integer	Characteristics of column data loaded by the IMCU: • **1**: common. • **2**: all empty. • **3**: same value. • **4**: no maximum or minimum value. • **8**: contains a blank line. • **16**: no sum value.
Output parameter	magic	integer	XID value of the transaction when IMCU data is generated. This value is used to check whether the IMCU data is valid.
Output parameter	attnum	smallint	attnum corresponding to the column loaded by IMCU.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	scan_cnt	bigint unsigned	Total number of times that the IMCU is scanned.
Output parameter	hit_cnt	bigint unsigned	Number of times that the memory-state IMCU is hit.
Output parameter	data_location	smallint	Specifies that the IMCU is in the memory, in the disk, or in escape state. • **0**: memory. • **1**: disk. • **2**: escape state. Only metadata is created due to insufficient memory. When this part of data is queried, it is read from the row store.

Example:

```
htapdb=# SELECT * FROM gs_imcu_meta('htap_flush_by_name_part1',
'htap_flush_by_name_part1_p2');
        rel_name         |       partition_name        | subpartition_name | schema_name | rowgroup_id |
row_count | imcu_size | imcu_data_size | compressed_data_size | compressed_ratio | compressed_type |
imcu_mode | magic | attnum | scan_cnt | hit_cnt | data_location
-------------------------+-----------------------------+-------------------+-------------+-------------+-----------
+-----------+----------------+----------------------+------------------+-----------------+-----------+-------
+--------+----------+---------+--------------
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      |           1 |
1 |      4016 |            16 |                   16 |                1 | uncompressed    |         1 |     0 |      0 | 0
| 0       |            0
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      |           1 |
1 |         0 |             0 |                    0 |                0 | none            |         3 |     0 |      1 | 0
|  0
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      |           1 |
1 |         0 |             0 |                    0 |                0 | none            |         3 |     0 |      2 | 0
|  0
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      |           1 |
1 |         0 |             0 |                    0 |                0 | none            |         3 |     0 |      3 | 0
|  0
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      |           1 |
1 |         0 |             0 |                    0 |                0 | none            |         3 |     0 |      4 | 0
|  0
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      |           1 |
1 |      4032 |            24 |                   24 |                1 | uncompressed    |         4 |     0 |      5 | 0
| 0       |            0
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public      |           1 |
1 |         0 |             0 |                    0 |                0 | none            |         3 |     0 |      6 | 0
|  0
(7 rows)
```

Note: In this example, each row represents a IMCU (the column whose **attnum** is **0** is the **CTID** column).

- gs_global_htap_file_status(db_name text)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Displays the usage information about the IMCU flush files of HTAP column-store tables. In a distributed system, this function can be executed only on CNs. The information about all DNs is returned.

Return type: record

NOTICE

- This function is available only when **enable_htap** is set to **on**.
- Only a system administrator has the permission to execute this function.

Table 1-177 gs_global_htap_file_status parameters

Category	Parameter	Type	Description
Input parameter	db_name	text	Database name. If the value is '' or **NULL**, all databases are queried.
Output parameter	node_name	text	Database node name.
Output parameter	db_oid	integer	Database ID.
Output parameter	file_name	text	Data file name.
Output parameter	file_id	integer	Data file ID. In the current version, only **2** is returned.
Output parameter	extent_size	integer	Extent size of a data file.
Output parameter	allocated_blocks	bigint	Number of pages that have been allocated to the IMCU in the data file.
Output parameter	total_blocks	bigint	Total number of physical pages in a data file on the disk.
Output parameter	high_water_mark	bigint	High watermark of the number of pages used by a data file.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	utilization	real	Percentage of used blocks to the total number of blocks.
Output parameter	ext	text	Extended column. The value is **NULL** in the current version.

Example:

```
htapdb=# SELECT * FROM gs_global_htap_file_status('htapdb');
 node_name | db_oid |        file_name         | file_id | extent_size | allocated_blocks | total_blocks |
high_water_mark | utilization | ext
-----------+--------+--------------------------+---------+-------------+------------------+--------------
+-----------------+-------------+-----
 datanode1 |  16384 | global/imcu/16384/2_imcu |       2 |           4 |                0 |            0 |
0 |           1 |
 datanode2 |  16384 | global/imcu/16384/2_imcu |       2 |           4 |                0 |            0 |
0 |           1 |
(2 rows)
```

- gs_htap_file_status(db_name text)

  Description: Displays the usage information about the IMCU flush files of HTAP column-store tables. In a distributed system, the information about the current node is returned.

  Return type: record

NOTICE

- This function is available only when **enable_htap** is set to **on**.
- Only SYSADMIN or OPRADMIN can execute this function.

Table 1-178 gs_htap_file_status parameters

Category	Parameter	Type	Description
Input parameter	db_name	text	Database name. If the value is **''** or **NULL**, all databases on the current node are queried.
Output parameter	db_oid	oid	Database ID.
Output parameter	file_name	text	Data file name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	file_id	integer	Data file ID. In the current version, only **2** is returned.
Output parameter	extent_size	integer	Extent size of a data file.
Output parameter	allocated_blocks	bigint	Number of pages that have been allocated to the IMCU in the data file.
Output parameter	total_blocks	bigint	Total number of physical pages in a data file on the disk.
Output parameter	high_water_mark	bigint	High watermark of the number of pages used by a data file.
Output parameter	utilization	real	Percentage of used blocks to the total number of blocks.
Output parameter	ext	text	Extended column. The value is **NULL** in the current version.

Example:

```
htapdb=# SELECT * FROM gs_htap_file_status('htapdb');
 db_oid |        file_name          | file_id | extent_size | allocated_blocks | total_blocks | high_water_mark | utilization | ext
--------+---------------------------+---------+-------------+------------------+--------------+-----------------+-------------+-----
  16384 | global/imcu/16384/2_imcu |       2 |           4 |                0 |            0 |               0 |           1 |
(1 row)
```

- gs_global_htap_scan_hit_status(relname text, partname text)

  Description: Queries the IMCU hit ratio information at the table level. In a distributed system, this function can be executed only on CNs. The information about all DNs is returned. If the value of **reloid** is **0**, the hit rates of all IMCV tables in the current database are returned. If the value of **reloid** is not 0, the hit rate of the specified table is returned.

  Return type: record

NOTICE

- This function is available only when **enable_htap** is set to **on**.
- Only a system administrator has the permission to execute this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-179 gs_global_htap_scan_hit_status parameters

Category	Parameter	Type	Description
Input parameter	relname	text	Table name. If you want to query cross-schema table information, you can set the value in the '*schema.relname*' format.
Input parameter	partname	text	Partition name. When all partitions of a partitioned table or a non-partitioned table are queried, the value can be **NULL** or ''.
Output parameter	node_name	text	Database node name.
Output parameter	rel_name	name	Table name.
Output parameter	partition_name	name	Partition name.
Output parameter	subpartition_name	name	Subpartition name.
Output parameter	schema_name	name	Schema name.
Output parameter	column_number	integer	Column No.
Output parameter	column_name	name	Column name.
Output parameter	imcu_in_cache_cnt	bigint unsigned	Number of IMCUs stored in the cache.
Output parameter	imcu_ave_scan_cnt	bigint unsigned	Average number of times that each IMCU is used in the column.
Output parameter	imcu_total_scan_cnt	bigint unsigned	Total number of times that IMCUs are used in the column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	imcu_ave_hit_cnt	bigint unsigned	Average number of cache hits per IMCU in the column.
Output parameter	imcu_total_hit_cnt	bigint unsigned	Total number of cache hits of IMCUs in the column.
Output parameter	total_hit_percentage	real	Total cache hit ratio of IMCUs in this column.

Example:

```
htapdb=# SELECT * FROM gs_global_htap_scan_hit_status('htap_flush_by_name_part1',
'htap_flush_by_name_part1_p2');
 node_name |        rel_name        |       partition_name       | subpartition_name | schema_name |
column_number | column_name | imcu_in_cache_cnt | imcu_ave_scan_cnt | imcu_total_scan_cnt |
imcu_ave_hit_cnt | imcu_total_hit_cnt | total_hit_percentage
-----------+------------------------+----------------------------+-------------------+-------------
+---------------+-------------+-------------------+-------------------+---------------------+------------------
+--------------------+----------------------
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|             0 | ctid        | 1                 | 0                 | 0                 | 0                   | 0
|                  0
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|             1 | c1          | 1                 | 0                 | 0                 | 0                   | 0
|                  0
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|             2 | c2          | 1                 | 0                 | 0                 | 0                   | 0
|                  0
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|             3 | c3          | 1                 | 0                 | 0                 | 0                   | 0
|                  0
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|             4 | c4          | 1                 | 0                 | 0                 | 0                   | 0
|                  0
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|             5 | c5          | 1                 | 0                 | 0                 | 0                   | 0
|                  0
 datanode1 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|             6 | c6          | 1                 | 0                 | 0                 | 0                   | 0
|                  0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|             0 | ctid        | 1                 | 0                 | 0                 | 0                   | 0
|                  0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|             1 | c1          | 1                 | 0                 | 0                 | 0                   | 0
|                  0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|             2 | c2          | 1                 | 0                 | 0                 | 0                   | 0
|                  0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|             3 | c3          | 1                 | 0                 | 0                 | 0                   | 0
|                  0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|             4 | c4          | 1                 | 0                 | 0                 | 0                   | 0
|                  0
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                   | public
|             5 | c5          | 1                 | 0                 | 0                 | 0                   | 0
|                  0
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 datanode2 | htap_flush_by_name_part1 | htap_flush_by_name_part1_p2 |                | public
|           6 | c6          | 1               | 0               | 0               | 0              | 0
|                0
(14 rows)
```

- gs_htap_scan_hit_status(relname text, partname text)

  Description: Queries the IMCU hit rate information at the table level. In a distributed system, the information about the current node is returned. If both **relname** and **partname** are **null** or empty, the hit rates of all IMCV tables in the current node are returned. In other cases, the hit rates of the specified table or partition are returned.

  Return type: record

NOTICE

- This function is available only when **enable_htap** is set to **on**.
- Only SYSADMIN or OPRADMIN can execute this function.

Table 1-180 gs_htap_scan_hit_status parameters

Category	Parameter	Type	Description
Input parameter	relname	text	Table name. If you want to query cross-schema table information, you can set the value in the '*schema.relname*' format.
Input parameter	partname	text	Partition name. When all partitions of a partitioned table or a non-partitioned table are queried, the value can be **NULL** or **''**.
Output parameter	rel_name	name	Table name.
Output parameter	partition_name	name	Partition name.
Output parameter	subpartition_name	name	Subpartition name.
Output parameter	schema_name	name	Schema name.
Output parameter	column_number	integer	Column sequence number.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	column_name	name	Column name.
Output parameter	imcu_in_cache_cnt	bigint unsigned	Number of IMCUs stored in the cache.
Output parameter	imcu_total_cnt	bigint unsigned	Total number of IMCUs in a column.
Output parameter	imcu_ave_scan_cnt	bigint unsigned	Average number of times that each IMCU is used in the column.
Output parameter	imcu_total_scan_cnt	bigint unsigned	Total number of times that IMCUs are used in the column.
Output parameter	imcu_ave_hit_cnt	bigint unsigned	Average number of cache hits per IMCU in the column.
Output parameter	imcu_total_hit_cnt	bigint unsigned	Total number of cache hits of IMCUs in the column.
Output parameter	total_hit_percentage	real	Total cache hit rate of IMCUs in the column.

Example:

```
gaussdb=# SELECT * FROM gs_htap_scan_hit_status('htap_flush_by_name_part1',
'htap_flush_by_name_part1_p1');
        rel_name         |       partition_name        | subpartition_name | schema_name | column_number
| column_name | imcu_in_cache_cnt | imcu_total_cnt | imcu_ave_scan_cnt | imcu_total_scan_cnt |
imcu_ave_hit_cnt | imcu_total_hit_cnt | tot
al_hit_percentage
-------------------------+-----------------------------+-------------------+-------------+---------------
+-------------+-------------------+----------------+-------------------+---------------------+------------------
+--------------------+----
------------------
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p1 |                   | public      |           0 |
ctid        | 1                 | 1              | 0                 | 0                 | 0                |
              0
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p1 |                   | public      |           1 |
c1          | 1                 | 1              | 0                 | 0                 | 0                |
              0
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p1 |                   | public      |           2 |
c2          | 1                 | 1              | 0                 | 0                 | 0                |
              0
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p1 |                   | public      |           3 |
c3          | 1                 | 1              | 0                 | 0                 | 0                |
              0
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p1 |                   | public      |           4 |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
c4          | 1               | 1             | 0               | 0               | 0             | 0               |
              0
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p1 |               | public     |           5 |
c5          | 1               | 1             | 0               | 0               | 0             | 0               |
              0
 htap_flush_by_name_part1 | htap_flush_by_name_part1_p1 |               | public     |           6 |
c6          | 1               | 1             | 0               | 0               | 0             | 0               |
              0
(7 rows)
```

- gs_imcu_size_estimation(relname text, partname text, colnum_name_array text[])

  Description: Returns the estimated memory usage for creating IMCV column-store data for a specified column in a specified partition of a table. When this function is executed on a DN, the estimated value of the current node is returned. When this function is executed on a CN, the estimated value of each node is returned.

  Return type: record

  NOTICE

  - This function is available only when **enable_htap** is set to **on**.
  - Only SYSADMIN or OPRADMIN can execute this function.
  - This function cannot be executed on the standby node. The restrictions of this function are the same as those of HTAP DDL. The error information is the same as that of DDL execution.

  Table 1-181 gs_imcu_size_estimation parameters

Category	Parameter	Type	Description
Input parameter	relname	text	Table name. If you want to query cross-schema table information, you can set the value in the '*schema.relname*' format.
Input parameter	partname	text	Partition name. When all partitions of a partitioned table or a non-partitioned table are queried, the value can be **NULL** or **''**.
Input parameter	colnum_name_array	text[]	Column name array.
Output parameter	node_name	name	Node name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	estimated_size	bigint unsigned	Estimated memory usage, in bytes.

Example:

```
htapdb=# SELECT * FROM gs_imcu_size_estimation('estimate_htap', '', ARRAY['id']);
 node_name | estimated_size
-----------+----------------
 datanode1 | 28411
 datanode2 | 28089
(2 rows)
```

- gs_htap_data_maintenance(rel_oid oid, type integer)

  Description: Maintains column-store data and provides two capabilities: (1) unloads column-store data (clears the memory and disk space); (2) flushes all column-store data to disks.

  Return type: text.

NOTICE

This API directly unloads IMCV to protect the row-store TP service. As a result, an error is reported for the service that has used IMCV Scan. Therefore, you need to fully confirm the impact before using this API. This function is used only for emergency escape when the cluster cannot be shut down due to excessive usage of resources (such as IMCV memory) caused by configuration or other reasons.

NOTE

- This function is available only when **enable_htap** is set to **on**.
- Only SYSADMIN or OPRADMIN can execute this function.
- This function cannot be executed in a transaction.
- In distributed mode, this function can be executed only on DNs.

Table 1-182 gs_htap_data_maintenance parameters

Category	Parameter	Type	Description
Input parameter	rel_oid	oid	OID of a row-store table. The value **0** indicates that all tables in the database on the current DN are processed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Input parameter	type	integer	Processing type. The options are as follows: • **0**: All IMCV data of the corresponding table is unloaded, including the memory and flushed data files. Only metadata is retained. The gs_imcv system catalog still contains information. The status is **'E'** in the gs_imcv_status view. When the input parameter **rel_oid** is set to **0**, IMCV data of all tables is unloaded, flushed files are deleted, and disk space is released. After the IMCV is unloaded using this API, the IMCV status is **'E'**. In this case, HTAP functions cannot be used. You need to create an IMCV again or run the VACUUM FULL command to change the IMCV status of the base table and corresponding partition to **'F'**. • **1**: This parameter has been deprecated. • **2**: IMCV data of the corresponding table is flushed to disk.

Example:

```
htapdb=# SELECT * FROM gs_htap_data_maintenance(16824,0);
 gs_htap_data_maintenance
--------------------------
 Maintenance finish.
(1 row)
htapdb=# SELECT * FROM gs_htap_data_maintenance(16824,2);
 gs_htap_data_maintenance
--------------------------
 Maintenance finish.
(1 row)
```

1.6.27.15 Other Functions

- pgxc_pool_check()

  Description: Checks whether the connection data buffered in the pool is consistent with pgxc_node.

  Return type: Boolean.

- pgxc_pool_reload()

  Description: Updates the connection information buffered in the pool.

  Return type: Boolean.

- reload_active_coordinator()

  Description: Updates the connection information buffered in the pool for all active CNs.

  Return type: void.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pgxc_lock_for_backup()

  Description: Locks a cluster for taking backup that would be restored on the new node to be added.

  Return type: Boolean.

  Note: Only users with the SYSADMIN permission can call this function.

  NOTE

  pgxc_lock_for_backup locks a cluster before gs_dump or gs_dumpall is used to back up the cluster. After a cluster is locked, operations changing the system structure are not allowed. This function does not affect DML statements.

- pg_pool_validate(clear bool, node_name text)

  Description: Displays invalid connections in the pooler between the CN and node_name. When the value of **clear** is **true**, invalid connections are cleared.

  Return type: record.

- gs_validate_node_conn(validate_type cstring, validate_node_name cstring)

  Description: Displays or clears the connections received by the current node from a specified node.

  Return type: record.

Table 1-183 gs_validate_node_conn parameters

Category	Parameter	Type	Description
Input parameter	validate_type	cstring	Specifies an input parameter type. • **'check'**: displays the connections received by the current node from a specified node. • **'clear'**: clears the connections received by the current node from a specified node.
Input parameter	validate_node_name	cstring	Specifies a node name.
Output parameter	pid	bigint	Displays the ID of a thread where the current node receives connections from the specified node. If the thread is from a thread pool, the value is **0**.
Output parameter	sessionid	bigint	Displays the ID of a session where the current node receives connections from the specified node. If the thread is not from a thread pool, the value is **0**.
Output parameter	node_name	text	Displays the name of a node from which the current node receives connections.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pgxc_pool_connection_status()

  Description: Checks whether the pooler connection status is **normal**.

  Return type: Boolean.

- pg_nodes_memory()

  Description: Queries the memory usage of all nodes.

  Return type: record.

- table_skewness(text)

  Description: Queries the percentage of table data among all nodes. This function is not applicable to hash bucket tables.

  Parameter: name of the table to be queried (text type).

  Return type: record.

- table_skewness(text, text, text)

  Description: Queries the percentage of a specified column in the table data among all nodes.

  Parameters: name of the table to be queried, specified column name, and number of records in the specified table. The default value is **0**, indicating that all records are queried. All parameters are of the text type.

  Return type: record.

  Return value description: Node ID, number of data rows in a specified column, and percentage of the data volume of the current node to the total data volume

  Example:

  ```
  -- Return the distribution of the first five rows of data in the column a of the table t on a node.
  gaussdb=# SELECT table_skewness('t', 'a',5);
   table_skewness
  ----------------
   (1,3,60.000%)
   (2,2,40.000%)
  (2 rows)

  -- Return the distribution of all data in the column a of the table t on a node.
  gaussdb=# SELECT table_skewness('t', 'a');
   table_skewness
  ----------------
   (1,7,70.000%)
   (2,2,20.000%)
   (0,1,10.000%)
  (3 rows)
  ```

- table_skewness_with_schema(text, text)

  Description: Checks the proportion of table data on all nodes. The function is the same as that of **table_skewness(text)**.

  text indicates that the types of the schema name and table name for the table to be queried are both text.

  Return type: record.

- table_data_skewness(colrecord, type)

  Description: Queries the node where the table data is located.

  Parameters:

  – **colrecord**: column name record of the table to be queried. The value is of the record type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - **type**: hash distribution type. For details about its value, see the description of the **pclocatortype** column in the **PGXC_CLASS** system catalog.

  Return type: smallint.

  Example:

```
-- Return the node where the index column data of the 'test1' table is located.
gaussdb=# SELECT table_data_skewness(row(index), 'R') FROM test1;
 table_data_skewness
---------------------
                   4
                   3
                   1
                   2
(4 rows)
```

- table_distribution(schemaname text, tablename text)

  Description: Queries the storage space occupied by a specified table on each node.

  text indicates that the types of the schema name and table name for the table to be queried are both text.

  Return type: record.

  NOTE

  - To query the storage distribution of a specified table by using this function, you must have the SELECT permission on the table.
  - The performance of **table_distribution** is better than that of **table_skewness**. Especially, in scenarios with a large amount of data, **table_distribution** is recommended.
  - When you use **table_distribution** and want to view the space usage, you can use **dnsize** or **(sum(dnsize) over ())** to view the percentage.

- table_distribution()

  Description: Queries the storage distribution of all tables in the current database.

  Return type: record.

  NOTE

  - This function involves the query for information about all tables in the database. To execute this function, you must have the administrator rights.
  - Based on the table_distribution() function, GaussDB provides the PGXC_GET_TABLE_SKEWNESS view (see **PGXC_GET_TABLE_SKEWNESS**) as an alternative way to query data skew. You are advised to use this view when the number of tables in the database is less than 10000.

- plan_seed()

  Description: Obtains the seed value of the previous query statement (internal use).

  Return type: int.

- pg_stat_get_env()

  Description: Obtains the environment variable information of the current node. Only users with the SYSADMIN or MONADMIN permission can access the environment variable information.

  Return type: record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT pg_stat_get_env();
                                                
```

- pgxc_get_node_env()

  Description: Provides the environment variable information about all nodes in a cluster.

  Return type: record.

- pgxc_disaster_read_set(text)

  Description: Configures node information about the standby cluster for DR on ETCD. Only the standby cluster for DR is available and can be called only by the initial user.

  Return type: Boolean

- pgxc_disaster_read_init()

  Description: Initializes readable DR resources and status information. Only the standby cluster for DR is available and can be called only by the initial user.

  Return type: Boolean

- pgxc_disaster_read_clear()

  Description: Clears readable DR resources and status information. Only the standby cluster for DR is available and can be called only by the initial user.

  Return type: Boolean.

- pgxc_disaster_read_status()

  Description: Provides node information about the standby cluster for DR. This function is available only for the standby cluster for DR.

  Return type: record.

- disaster_read_set_cn_status()

  Description: Updates the CN active status information during DR read. Only the standby cluster for DR is available.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT
  g_catalog.disaster_read_set_cn_status('44000000000
  ```

```
----------------------------
 t
(1 row)
```

- gs_switch_relfilenode()

  Description: Exchanges meta information of two tables or partitions. (This is only used for the redistribution tool. An error message is displayed when the function is directly used by users).

  Return type: int.

  NOTE

  This function may cause clearing of statistics. You are advised to collect statistics again after this function is called.

- pg_catalog.plancache_clean()

  Description: Clears the global plan cache that is not used on the current node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: Boolean.

- DBE_PERF.global_plancache_clean()

  Description: Clears the global plan cache that is not used on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: Boolean

- pg_catalog.global_plancache_set_invalid(hashkey, unique_sql_id)

  Description: Manually invalidates the GPC cache plan of the current node using the input parameters **hashkey** and **unique_sql_id** of the bigint type. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: text.

  Example:

  ```
  -- Obtain hash_value from the dbe_perf.global_plancache_status view as the first input parameter and unique_sql_id as the second input parameter.
  gaussdb=# SELECT * FROM pg_catalog.global_plancache_set_invalid(3928396421,3641111755);
   global_plancache_set_invalid
  ------------------------------
   Execution succeeded.
  (1 row)
  ```

  NOTE

  - You are advised not to use this function because it increases the number of hard parsing times and affects performance.
  - The function does not support the invalidation of the plan whose **unique_sql_id** is **0**.

- pg_catalog.global_plancache_show_plan(hashkey, unique_sql_id)

  Description: Uses the input parameters **hashkey** and **unique_sql_id** of the bigint type to view the corresponding GPC cache plan. After a cluster is created, by default, you must have the MONADMIN permission to query this function. This function does not support the explain_perf_mode pretty mode.

  Return type: text.

  Example:

```
-- Obtain hash_value from the dbe_perf.global_plancache_status view as the first input parameter
and unique_sql_id as the second input parameter.
gaussdb=# SELECT * FROM pg_catalog.global_plancache_show_plan(3928396421,3641111755);
      global_plancache_show_plan
----------------------------------------
 Result  (cost=0.00..0.01 rows=1 width=0)+

(1 row)
```

NOTE

The function does not display the plan whose **unique_sql_id** is **0**.

- copy_error_log_create()

  Description: The error table required by the COPY FROM error tolerance mechanism has been changed to the system catalog created during initialization. Therefore, the current function always returns **true** for compatibility.

  Return type: Boolean.

  Table 1-184 pg_catalog.pgxc_copy_error_log information

Column	Type	Description
relname	character varying	Table name in the form of *Schema name*.*Table name*
begintime	timestamp with time zone	Time when a data format error was reported.
filename	character varying	Name of the source data file where a data format error occurs.
lineno	bigint	Number of the row where a data format error occurs in a source data file.
rawrecord	text	Raw record of a data format error in the source data file.
detail	text	Error details.
custom_id	text	Value of **copy_custom_id**, which indicates the customized ID information to be imported.

- pgxc_copy_error_func(boolean need_union, boolean only_dn)

  Description: Queries the content in the pg_catalog.gs_copy_error_log catalog on each node and returns the content. The names and data types of the columns in the return value **record** correspond to those in the pg_catalog.gs_copy_error_log catalog.

  Return type: setof record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-185 pgxc_copy_error_func parameters

Parameter	Type	Description
need_union	Boolean	• **true**: queries data in both the pg_catalog.gs_copy_error_log and pg_catalog.gs_copy_error_old catalogs and performs the union_all operation. • **false**: queries only data in the pg_catalog.gs_copy_error_log catalog.
only_dn	Boolean	• **true**: queries only data on all DNs. • **false**: queries data on all nodes.

- gs_copy_error_insert(cstring relname, cstring begintime, cstring filename, bigint lineno, cstring rawrecord, cstring detail, cstring custom_id)

  Description: Inserts data into the pg_catalog.gs_copy_error_log catalog on a DN. This function is intended for internal use during COPY operations; users should not call it directly.

  Return type: Boolean.

  Table 1-186 gs_copy_error_insert parameters

Parameter	Type	Description
relname	cstring	Table name in the form of *Schema name*.*Table name*
begintime	cstring	Time when a data format error was reported.
filename	cstring	Name of the source data file where a data format error occurs.
lineno	bigint	Number of the row where a data format error occurs in a source data file.
rawrecord	cstring	Raw record of a data format error in the source data file.
detail	cstring	Error details.
custom_id	cstring	Value of **copy_custom_id**, which indicates the customized ID information to be imported.

- gs_copy_summary_func(boolean need_union)

  Description: Queries the content in the pg_catalog.gs_copy_summary_log catalog on the current node and returns the content. Currently, this function is not supported.

  Return type: setof record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-187 gs_copy_summary_func parameters

Parameter	Type	Description
need_union	Boolean	• **true**: queries data in both the pg_catalog.gs_copy_summary_log and pg_catalog.gs_copy_summary_old catalogs and performs the union_all operation. • **false**: queries only data in the pg_catalog.gs_copy_summary_log catalog.

- pg_stat_get_data_senders()

  Description: Provides detailed information about the data-copy sender thread active at the moment.

  Return type: record.

- textlen()

  Description: Provides the method of querying the logical length of text.

  Return type: int.

- threadpool_status()

  Description: Displays the status of worker threads and sessions in the thread pool.

  Return type: record.

- get_local_active_session()

  Description: Provides sampling records of historical active sessions stored in the memory by current node. Users with the SYSADMIN or MONADMIN permission can view all historical active session records of the current node. Common users can view the historical active session records of the current session.

  Return type: record.

- dbe_perf.get_global_gs_asp(timestamp,timestamp)

  Description: Provides sampling records of the historical active sessions stored in the gs_asp system catalog of all nodes. The **event_start_time**, **current_xid**, and **top_xid** columns are added to GaussDB Kernel 505.0. After GaussDB Kernel is upgraded to GaussDB Kernel 505.0 or later, the columns cannot be queried during the upgrade observation period.

  Return type: record.

- get_wait_event_info()

  Description: Provides detailed information about the wait event.

  Return type: record.

- dbe_perf.get_datanode_active_session(text)

  Description: Provides sampling records of historical active sessions stored in the memory of primary DN, which is queried from CN. The **event_start_time**, **current_xid**, and **top_xid** columns are added to GaussDB Kernel 505.0. After GaussDB Kernel is upgraded to GaussDB Kernel 505.0 or later, the columns cannot be queried during the upgrade observation period.

  Return type: record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Note: This function queries the records in the DBE_PERF.LOCAL_ACTIVE_SESSION view (see **DBE_PERF.LOCAL_ACTIVE_SESSION**) on the target DN and matches the records with those in the DBE_PERF.LOCAL_ACTIVE_SESSION view on all CNs to obtain the query string. Therefore, a large amount of memory is occupied.

- dbe_perf.get_datanode_active_session_hist(text,timestamp,timestamp)

  Description: Provides sampling records of historical active sessions stored in the gs_asp system catalog of primary DN, which is queried from CN. The **event_start_time**, **current_xid**, and **top_xid** columns are added to GaussDB Kernel 505.0. After GaussDB Kernel is upgraded to GaussDB Kernel 505.0 or later, the columns cannot be queried during the upgrade observation period.

  Return type: record.

  Note: This function queries the gs_asp records of a specified period on the target DN. If the period is specified too long, too many records will be queried, which takes a long time.

- generate_wdr_report(begin_snap_id bigint, end_snap_id bigint, report_type cstring, report_scope cstring, node_name cstring)

  Description: Generates a system diagnosis report based on two snapshots. By default, the initial user or MONADMIN can access the report. The result can be queried only in the system database but cannot be queried in the user database.

  Return type: text.

  Table 1-188 generate_wdr_report parameters

Parameter	Description	Value Range
begin_snap_id	Snapshot ID that starts the diagnosis report period.	1~9,223,372,036,854,775,807
end_snap_id	Snapshot ID that ends the diagnosis report period. By default, the value of **end_snap_id** is greater than that of **begin_snap_id**.	1~9,223,372,036,854,775,807
report_type	Specifies the type of the generated report.	• **summary** • **detail** • **all**: Both **summary** and **detail** are included.
report_scope	Specifies the scope for a report to be generated.	• **cluster**: database-level information • **node**: node-level information

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range
node_name	• If **report_scope** is set to **node**, this parameter is set to the name of the corresponding node. You can run the **SELECT * FROM pg_node_env;** command to query the node name. • If **report_scope** is set to **cluster**, the value can be omitted, left blank, or **NULL**.	• **cluster**: omitted, empty, or NULL. • **node**: node name in GaussDB.

- generate_wdr_db_report(db_name cstring, begin_snap_id bigint, end_snap_id bigint, report_type cstring, report_scope cstring, node_name cstring)

  Description: Generates database-level system diagnosis reports based on two snapshots. You need to run the command in the system database. By default, the initial user or users with the MONADMIN permission can access the database. The result can be queried only in the system database but cannot be queried in the user database.

  Return type: record.

  Table 1-189 generate_wdr_db_report parameters

Parameter	Description	Value Range
db_name	Specifies the name of a database.	N/A
begin_snap_id	Snapshot ID that starts the diagnosis report period.	1~9,223,372,036,854,775,807
end_snap_id	Snapshot ID that ends the diagnosis report period. By default, the value of **end_snap_id** is greater than that of **begin_snap_id**.	1~9,223,372,036,854,775,807
report_type	Specifies the type of the generated report.	• **summary** • **detail** • **all**: Both **summary** and **detail** are included.
report_scope	Specifies the scope for a report to be generated.	• **cluster**: database-level information • **node**: node-level information

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range
node_name	• If **report_scope** is set to **node**, this parameter is set to the name of the corresponding node. You can run the **SELECT * FROM pg_node_env;** command to query the node name. • If **report_scope** is set to **cluster**, the value can be omitted, left blank, or **NULL**.	• **cluster**: omitted, empty, or NULL. • **node**: node name in GaussDB.

- create_wdr_snapshot()

  Description: Manually generates system diagnosis snapshots. This function requires the SYSADMIN permission and can be executed only on the CCN.

  Return type: text.

- kill_snapshot()

  Description: Kills the WDR snapshot backend thread. Users who call this function must have the SYSADMIN permission or the REPLICATION permission, or inherit permissions of the built-in role gs_role_replication.

  Return type: void.

- dbe_perf.generate_asp_report(start_time timestamp with time zone, end_time timestamp with time zone, datanode text, slot_count bigint)

  Description: Generates ASP diagnosis reports based on timestamps and node information, which can be accessed by MONADMIN. **slot_count** indicates the number of time segments for displaying metrics. **datanode** can specify a partition DN (for example, dn_6001_6002_6003) or a DN (dn_6001, dn_6002, or dn_6003). This function can be used only on the CN.

  Return type: text.

- generate_asp_report(start_time timestamp with time zone, end_time timestamp with time zone, slot_count bigint, node_name text)

  Description: Generates ASP based on timestamps, which can be accessed by MONADMIN. **slot_count** indicates the number of time segments for displaying metrics. **node_name** is used to generate the ASP of a specified node. The value can only be the name of the current CN or can be left empty. When the value is left empty, this function is used to generate the report of the current node. This function can be used only on the CN.

  Return type: text.

- dbe_perf.get_active_session_profile(start_ts timestamp with time zone, end_ts timestamp with time zone, need_final boolean,node_name text)

  Description: Queries data in the ASP memory and disk based on timestamps. **need_final** specifies whether to query blocking information. The default value is **false**. **node_name** is used to distinguish the CN, active and standby DNs, for example, cn_5001, dn_6001 or dn_6002. This parameter cannot be left empty. The **event_start_time**, **current_xid**, and **top_xid** columns are added to GaussDB Kernel 505.0. After GaussDB Kernel is upgraded to GaussDB Kernel

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

505.0 or later, the columns cannot be queried during the upgrade observation period.

Return type: record.

- capture_view_to_json(text,integer)

  Description: Saves the view result to the directory specified by GUC: **perf_directory**. If **is_crossdb** is set to **1**, the view is accessed once for all databases. If the value of **is_crossdb** is **0**, the current database is accessed only once. Only users with the SYSADMIN or MONADMIN permission can execute this function.

  Return type: int.

- reset_unique_sql(text,text,bigint)

  Description: Clears the unique SQL statements in the memory of CN/DN. (The SYSADMIN or MONADMIN permission is required.)

  Return type: Boolean

  Table 1-190 reset_unique_sql parameters

Parameter	Type	Description
scope	text	Clearance scope type. The options are as follows: • **'GLOBAL'**: All CNs/DNs are cleared. If the value is **'GLOBAL'**, this function can be executed only on CNs. • **'LOCAL'**: The current node is cleared.
clean_type	text	• **'BY_USERID'**: Unique SQL statements are cleared based on user IDs. • **'BY_CNID'**: Unique SQL statements are cleared based on CN IDs. • **'ALL'**: All data is cleared.
clean_value	int8	Specifies the value to be cleared corresponding to **clean_type**. • If **clean_type** is set to **'BY_USERID'**, the records of the user whose ID is **clean_value** are cleared. • If **clean_type** is set to **'BY_CNID'**, the records of the CN whose ID is **clean_value** are cleared. • If the second parameter is set to **ALL**, the third parameter does not take effect and can be set to any value.

- wdr_xdb_query(db_name_str text, query text)

  Description: Provides the capability of executing local cross-database queries. For example, when logging to the **testdb** database, access tables in the **test** database. Only the initial user has the permission to run this command.

  ```
  SELECT col1 from wdr_xdb_query('dbname=test','select col1 from t1') as dd(col1 int);
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record.

- pg_wlm_jump_queue(pid int)

  Description: Moves a task to the top of the queue for CN.

  Return type: Boolean.

  - **true**: success
  - **false**: failure

- gs_wlm_switch_cgroup(pid int, cgroup text)

  Description: Moves a job to another Cgroup to change the job priority.

  Return type: Boolean.

  - **true**: success
  - **false**: failure

- pv_session_memctx_detail(threadid tid, MemoryContextName text)

  Description: Records information about the memory context **MemoryContextName** of the thread **tid** into the *threadid*_**timestamp.log** file in the *$GAUSSLOG*/**gs_log/***${node_name}*/**dumpmem** directory. You can obtain the thread ID by querying the **sessid** column in the PV_SESSION_MEMORY_DETAIL view (see **PV_SESSION_MEMORY_DETAIL**). In the officially released version, only the **MemoryContextName** that is an empty string (two single quotation marks indicate that the input is an empty string) is accepted. In this case, all memory context information is recorded. Otherwise, no operation is performed. This function can be executed only by the administrator.

  Return type: Boolean.

  - **true**: success
  - **false**: failure

- pg_shared_memctx_detail(MemoryContextName text)

  Description: Records information about the memory context **MemoryContextName** into the *threadid*_**timestamp.log** file in the *$GAUSSLOG*/**gs_log/***${node_name}*/**dumpmem** directory. Calling this function in the officially released version does not involve any operation. Only the administrator can execute this function.

  Return type: Boolean.

  - **true**: success
  - **false**: failure

- local_aio_completer_stat()

  Description: Displays statistics about the AIO Completer thread in the instance.

  Return type: record.

  Table 1-191 Return values of local_aio_completer_stat

Parameter	Type	Description
node_name	text	Name of the current instance.
tid	int8	ID of the AIO Completer thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
thread_type	text	AIO Completer thread type (read or write).
aio_submitted_num	int8	Number of committed asynchronous I/O requests of the AIO Completer thread.
aio_completed_num	int8	Number of completed asynchronous I/O requests of the AIO Completer thread.
aio_incompleted_num	int8	Number of asynchronous I/O requests that are not completed of the AIO Completer thread.
slot_count_left	int8	Indicates the number of idle slots.

- local_aio_slot_usage_status()

  Description: Displays statistics about asynchronous I/O commit slots in the instance.

  Return type: record.

  Table 1-192 Return values of local_aio_slot_usage_status

Parameter	Type	Description
node_name	text	Name of the current instance.
slot_id	int4	Slot ID.
slot_type	char	Slot type. The value can be **r** (read) or **w** (write).
status	bool	Slot usage status.
buffer_id	int8	Buffer ID corresponding to the slot.
relfilenode_blocknum	text	Position of the physical page where the buffer corresponding to the slot is located.
lsn	int8	LSN corresponding to the page.
submitted_time	int8	Time when a page is committed asynchronously.
elapsed_time	int8	Waiting time of the page.

- remote_aio_completer_stat()

  Description: Displays statistics about the AIO Completer thread in other instances (used only on CNs).

  Return type: record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-193 Return values of remote_aio_completer_stat

Parameter	Type	Description
node_name	text	Name of another instance.
tid	int8	ID of the AIO Completer thread.
thread_type	text	AIO Completer thread type (read or write).
aio_submitted_num	int8	Number of committed asynchronous I/O requests of the AIO Completer thread.
aio_completed_num	int8	Number of completed asynchronous I/O requests of the AIO Completer thread.
aio_incompleted_num	int8	Number of asynchronous I/O requests that are not completed of the AIO Completer thread.
slot_count_left	int8	Indicates the number of idle slots.

- remote_aio_slot_usage_status()

  Description: Displays statistics about asynchronous I/O commit slots in other instances.

  Return type: record.

  Table 1-194 Return values of remote_aio_slot_usage_status

Parameter	Type	Description
node_name	text	Name of another instance.
slot_id	int4	Slot ID.
slot_type	char	Slot type. The value can be **r** (read) or **w** (write).
status	bool	Slot usage status.
buffer_id	int8	Buffer ID corresponding to the slot.
relfilenode_blocknum	text	Position of the physical page where the buffer corresponding to the slot is located.
lsn	int8	LSN corresponding to the page.
submitted_time	int8	Time when a page is committed asynchronously.
elapsed_time	int8	Waiting time of the page.

- gs_get_io_type()

  Description: Displays the I/O mode of the instance.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: text.

- **BIO**: The instance is running in BIO mode (ADIO is disabled).
- **DIO**: The instance is running in DIO mode (ADIO is enabled).
- **BIO->DIO (In progress)**: The current instance is switching from the BIO mode to the DIO mode.

- local_bgwriter_stat()

  Description: Displays the page flushing information of the bgwriter thread of the current instance.

  Return type: record.

  Table 1-195 Return values of local_bgwriter_stat

Parameter	Type	Description
node_name	text	Name of the current instance.
bgwr_actual_flush_total_num	int8	Total number of dirty pages flushed by the bgwriter thread since the last restart (unit: pages).
bgwr_last_flush_num	int4	Number of dirty pages flushed by the bgwriter thread since the last dirty page flushing (unit: pages).
candidate_slots	int4	Number of pages that can be used for cache page requests in the candidate queue.
get_buffer_from_list	int8	Number of cache pages obtained from the candidate queue since the last restart (unit: pages).
get_buf_clock_sweep	int8	Number of cache pages obtained from the clock sweep mechanism since the last restart (unit: pages).

- local_candidate_stat()

  Description: Displays the number of pages in the candidate buffer chain of this instance and buffer elimination information, including the normal buffer pool and segment buffer pool.

  Return type: record.

- gs_buffer_manager_info()

  Description: Prints the number of buffer page accesses and misses in the current database since the last time this function was called (or since startup if it has never been executed), as well as the read rate (that is, 1 – Hit rate), and the sleep time for each round of the elimination queue provided by the intelligent elimination algorithm.

  Total access: %lu, Total read: %lu, Read rate: %0.3f, Sleep time :%u

  Return type: text.

- local_ckpt_stat()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Displays the information about checkpoints and flushing pages of the current instance.

Return type: record.

- local_double_write_stat()

  Description: Displays the doublewrite file status of the current instance.

  Return type: record.

  Table 1-196 local_double_write_stat parameters

Parameter	Type	Description
node_name	text	Instance name
curr_dwn	int8	Sequence number of the doublewrite file
curr_start_page	int8	Start page for restoring the doublewrite file
file_trunc_num	int8	Number of times that the doublewrite file is reused
file_reset_num	int8	Number of reset times after the doublewrite file is full
total_writes	int8	Total number of I/Os of the doublewrite file
low_threshold_writes	int8	Number of I/Os for writing doublewrite files with low efficiency (the number of I/O flushing pages at a time is less than 16)
high_threshold_writes	int8	Number of I/Os for writing doublewrite files with high efficiency (the number of I/O flushing pages at a time is more than 421)
total_pages	int8	Total number of pages that are flushed to the doublewrite file area
low_threshold_pages	int8	Number of pages that are flushed with low efficiency
high_threshold_pages	int8	Number of pages that are flushed with high efficiency
file_id	int8	ID of the current doublewrite file.

- local_pagewriter_running_stat()

  Description: Displays the running status of the threads responsible for batch dirty page flushing in the instance.

  Return type: record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-197 local_pagewriter_running_stat() parameters

Category	Parameter	Type	Description
Output parameter	nodename	text	Instance name
Output parameter	tid	int8	Thread ID of the pagewriter.
Output parameter	type	text	Thread type.
Output parameter	extre_start_loc	int8	Position of the ultimate dirty page flushing.
Output parameter	extre_left_size	int4	Number of dirty pages skipped in the previous round of ultimate dirty page flushing.
Output parameter	full_ckpt	bool	Specifies whether the full dirty page flushing mode is enabled.
Output parameter	shutdown_progress	bool	Exit status.
Output parameter	buf_id_start	int4	Start buffer ID of the scanning thread range.
Output parameter	buf_id_end	int4	End buffer ID of the scanning thread.
Output parameter	standby_buf_start_id	int4	Start buffer ID of the scanning thread with read on standby.
Output parameter	standby_buf_end_id	int4	End buffer ID of the scanning thread with read on standby.
Output parameter	exception_count	int4	Number of historical thread exceptions.
Output parameter	recent_exception_time	timestamptz	Timestamp of the latest exception.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- local_pagewriter_accumulate_stat()

  Description: Displays bandwidth and lock conflict statistics for the batch dirty page flushing threads of the instance within the specified time range.

  Return type: record.

  Table 1-198 local_pagewriter_accumulate_stat() parameters

Category	Parameter	Type	Description
Input parameter	last_time	int8	Statistical period (seconds).
Output parameter	nodename	text	Instance name
Output parameter	tid	int8	Thread ID of the pagewriter.
Output parameter	type	text	Thread type.
Output parameter	bandwidth	float8	Average bandwidth (MB/s).
Output parameter	lock_conflict_count	int8	Number of pages skipped due to content lock conflicts.
Output parameter	avg_pagewriter_sleep	float8	Actual average dirty page flushing period.

- local_dirty_queue_advance_stat()

  Description: Displays the update progress of the dirty page queue of the current instance.

  Return type: record.

  Table 1-199 local_dirty_queue_advance_stat() parameters

Category	Parameter	Type	Description
Output parameter	nodename	text	Instance name

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	dirty_queue_head	int8	Position of the dirty page queue header.
Output parameter	dirty_queue_tail	int8	Position of the dirty page queue tail.
Output parameter	dirty_queue_size	int8	Size of the dirty page queue.
Output parameter	dirty_queue_full	bool	Specifies whether the dirty page queue is full.
Output parameter	head_buffer	int4	Buffer value of the dirty page queue header.
Output parameter	head_buffer_lsn	text	Page LSN of the dirty page queue header.
Output parameter	head_buffer_content_lock_id	int8	lock_unique_id of the content lock for the dirty page queue head buffer.
Output parameter	head_buffer_io_lock_id	int8	lock_unique_id of the I/O lock for the dirty page queue head buffer.
Output parameter	extre_scan_current	int8	End position of the dirty page queue that has been allocated to the thread for ultimate dirty page flushing.

- local_single_flush_dw_stat()

  Description: Displays the elimination of doublewrite files on a single page in the instance.

  Return type: record.

- local_pagewriter_stat()

  Description: Displays the page flushing information and checkpoint information of the current instance.

  Return type: record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-200 local_pagewriter_stat() parameters

Category	Parameter	Type	Description
Output parameter	node_name	text	Instance name.
Output parameter	pgwr_actual_flush_total_num	int8	Total number of dirty pages flushed from the startup to the current time.
Output parameter	pgwr_last_flush_num	int4	Number of dirty pages flushed in the previous batch.
Output parameter	remain_dirty_page_num	int8	Estimated number of remaining dirty pages.
Output parameter	queue_head_page_rec_lsn	text	**recovery_lsn** of the first dirty page in the dirty page queue of the current instance.
Output parameter	queue_rec_lsn	text	**recovery_lsn** of the dirty page queue of the current instance.
Output parameter	current_xlog_insert_lsn	text	Position where Xlogs of the current instance are written. This parameter is invalid for a standby node.
Output parameter	ckpt_redo_point	text	Checkpoint of the current instance.

- local_redo_stat()

  Description: Displays the replay status of the current standby instance.

  Return type: record.

  Note: The returned replay status includes the current replay position and the replay position of the minimum restoration point.
- local_recovery_status()

  Description: Displays log flow control information about the primary and standby nodes.

  Return type: record.
- local_rto_status()

  Description: Displays log flow control information about the primary and standby nodes.

  Return type: record.
- gs_wlm_switch_cgroup(sess_id int8, cgroup name)

  Description: Switches the Cgroup of a specified session.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record.

- comm_client_info()

  Description: Queries active client connections of a single node. For details about the returned result, see **COMM_CLIENT_INFO**.

  Return type: SETOF record

- pg_get_flush_lsn()

  Description: Returns the position of the Xlog flushed from the current node.

  Return type: text.

- pg_get_sync_flush_lsn()

  Description: Returns the position of the Xlog flushed by the majority on the current node.

  Return type: text.

- pgxc_wlm_rebuild_user_resource_pool()

  Description: Rebuilds user and resource pool cache information. Only system administrators can execute this function.

  Return type: Boolean.

- locktag_decode(locktag text)

  Description: Parses lock details from **locktag**.

  Example:

```
gaussdb=# SELECT locktag_decode('271b:0:0:0:0:6');
                locktag_decode
-----------------------------------------------
 locktype:transactionid, transactionid:10011
(1 row)
```

  Return type: text.

- disable_conn(disconn_mode text, host text, port integer)

  Description: Specifies that the CM Agent processes commands delivered by the CM Server. When a DN is selected as the primary DN, it is configured to reject connections to all DNs, forcibly connect to a DN, or connect to all DNs in polling mode. Only the initial user and system administrator can call this function.

  Return type: void.

  Table 1-201 disable_conn parameters

Parameter	Type	Description
disconn_mode	text	DN connection mode: • **'prohibit_connection'**: rejects to connect to all DNs. • **'specify_connection'**: forcibly connects to a DN. • **'polling_connection'**: connects to all DNs in polling mode.
host	text	IP address of the DN

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
port	integer	Port number of the DN

- dbe_perf.get_global_full_sql_by_timestamp(start_timestamp timestamp with time zone, end_timestamp timestamp with time zone)

  Description: Obtains full SQL information about a cluster. The result can be queried only in the system database but cannot be queried in the user database.

  Return type: record.

  Table 1-202 dbe_perf.get_global_full_sql_by_timestamp parameters

Parameter	Type	Description
start_timestamp	timestamp with time zone	Start point of the SQL start time range.
end_timestamp	timestamp with time zone	End point of the SQL start time range. An error is reported during function execution if **start_timestamp** is greater than or equal to **end_timestamp**.

- dbe_perf.get_global_slow_sql_by_timestamp(start_timestamp timestamp with time zone, end_timestamp timestamp with time zone)

  Description: Obtains cluster-level slow SQL information. The result can be queried only in the system database but cannot be queried in the user database.

  Return type: record.

  Table 1-203 dbe_perf.get_global_slow_sql_by_timestamp parameters

Parameter	Type	Description
start_timestamp	timestamp with time zone	Start point of the SQL start time range.
end_timestamp	timestamp with time zone	End point of the SQL start time range. An error is reported during function execution if **start_timestamp** is greater than or equal to **end_timestamp**.

- statement_detail_decode(detail text, format text, pretty boolean)

  Description: Parses the **details** column in a full or slow SQL statement. The result can be queried only in the system database but cannot be queried in the user database.

  Return type: text.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-204 statement_detail_decode parameters

Parameter	Type	Description
detail	text	Set of events generated by the SQL statement (unreadable).
format	text	Parsing output format. The value is **plaintext**.
pretty	Boolean	Specifies whether to display the text in pretty format when **format** is set to **plaintext**. The options are as follows: • **true**: Events are separated by "\n". • **false**: Events are separated by commas (,).

- pgxc_get_csn(tid)

  Description: Returns the transaction commit sequence number (CSN) corresponding to a given transaction ID.

  Return type: int8.

- pgxc_get_csn(tid, bucketid)

  Description: Returns the transaction commit sequence number (CSN) corresponding to the transaction ID of the specified bucket ID. The input parameter **bucketid** indicates the physical bucket ID of the hash bucket table. For an ordinary table, the value is **–1**.

  Return type: int8.

- get_global_user_transaction()

  Description: Returns transaction information about each user on all nodes.

  Return types: node_name name, usename name, commit_counter bigint, rollback_counter bigint, resp_min bigint, resp_max bigint, resp_avg bigint, resp_total bigint, bg_commit_counter bigint, bg_rollback_counter bigint, bg_resp_min bigint, bg_resp_max bigint, bg_resp_avg bigint, bg_resp_total bigint, and dbid oid.

- pg_collation_for()

  Description: Returns the sorting rule corresponding to the input parameter string.

  Parameter: any. (Explicit type conversion is required for constants.)

  Return type: text.

- pgxc_unlock_for_sp_database(databasename text)

  Description: Releases a specified database lock.

  Parameter: **databasename** (database name).

  Return type: Boolean.

  Note: Only users with the SYSADMIN permission can call this function.

- pgxc_lock_for_sp_database(databasename text)

  Description: Locks a specified database.

  Parameter: **databasename** (database name).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: Boolean.

Note: Only users with the SYSADMIN permission can call this function.

- pgxc_unlock_for_transfer(schemaname text)

  Description: Releases the lock used for data transmission (data redistribution).

  Parameter: **schemaname** (schema name).

  Return type: Boolean.

  Note: Only users with the SYSADMIN permission can call this function.

- pgxc_lock_for_transfer(schemaname text)

  Description: Locks a database for data transmission (data redistribution).

  Parameter: **schemaname** (schema name).

  Return type: Boolean.

  Note: Only users with the SYSADMIN permission can call this function.

- gs_comm_proxy_thread_status()

  Description: Collects statistics on data packets sent and received by the proxy communications library **comm_proxy** when a user-mode network is configured for the cluster. This function is not supported in the current version.

  Parameter: nan.

  Return type: record.

- gs_catalog_attribute_records()

  Description: Returns the definition of each column in a system catalog specified by OID. Only ordinary system catalogs whose OIDs are less than 10000 are supported. Indexes and TOAST tables are not supported.

  Parameter: OID of a system catalog.

  Return type: record.

- dynamic_func_control(scope text, function_name text, action text, "{params}" text[])

  Description: Dynamically enables built-in functions. Currently, only full SQL statements can be dynamically enabled.

  Return type: record.

Table 1-205 dynamic_func_control parameters

Parameter	Type	Description
scope	text	Scope where the function is to be dynamically enabled. Currently, only **'GLOBAL'** and **'LOCAL'** are supported.
function_name	text	Function name. Currently, only **'STMT'** and **'GSTRACE'** are supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
action	text	When **function_name** is set to **'STMT'**, **action** can only be set to **TRACK**, **UNTRACK**, **LIST**, or **CLEAN**. All users can use the **LIST** option. Other options can be used only when the SYSADMIN or MONADMIN permission is granted. • **TRACK**: records the full SQL information of normalized SQL statements. • **UNTRACK**: cancels the recording of full SQL information of normalized SQL statements. • **LIST**: lists normalized SQL information that is recorded in the current track. • **CLEAN**: cleans normalized SQL information that is recorded in the current track. When **function_name** is set to **'GSTRACE'**, **action** can only be set to **TRACE_SQL**, **TRACE_FUNC**, **UNTRACE_SQL**, **UNTRACE_FUNC**, **UNTRACE_ALL_SQL**, **UNTRACE_ALL_FUNC**, **LIST_TRACE_SQL**, **LIST_TRACE_FUNC**, **SHOW_TRACED_FUNC**, **SHOW_MANGLED_FUNC**, or **SHOW_SUBFUNC**. • **TRACE_SQL**: GSTrace registers the specified SQL statement to be traced. • **TRACE_FUNC**: GSTrace registers the specified function to be traced. • **UNTRACE_SQL**: GSTrace unregisters the specified SQL statement to be traced. • **UNTRACE_FUNC**: GSTrace unregisters the specified function to be traced. • **UNTRACE_ALL_SQL**: GSTrace cancels all SQL statements. • **UNTRACE_ALL_FUNC**: GSTrace cancels all functions. • **LIST_TRACE_SQL**: GSTrace views the configured SQL statements to be traced. • **LIST_TRACE_FUNC**: GSTrace views the configured functions to be traced. • **SHOW_TRACED_FUNC**: GSTrace views the status of the functions being traced. • **SHOW_MANGLED_FUNC**: GSTrace views the mangled name of a specified function. • **SHOW_SUBFUNC**: GSTrace specifies all subfunctions of a function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
params	text[]	When **function_name** is set to **'STMT'**, the **params** values corresponding to different actions are set as follows: • **TRACK**: **'{"Normalized SQLID", "L0/L1/L2"}'** • **UNTRACK**: **'{"Normalized SQLID"}'** • **LIST**: **'{}'** • **CLEAN**: **'{}'** When **function_name** is set to **'GSTRACE'**, the **params** values corresponding to different actions are set as follows: • **TRACE_SQL**: For details, see the input parameter description of the gstrace_reg_sql() function. • **TRACE_FUNC**: For details, see the input parameter description of the gstrace_reg_func() function. • **UNTRACE_SQL**: For details, see the input parameter description of the gstrace_unreg_sql() function. • **UNTRACE_FUNC**: For details, see the input parameter description of the gstrace_unreg_func() function. • **UNTRACE_ALL_SQL**: For details, see the input parameter description of the gstrace_unreg_all_sql() function. • **UNTRACE_ALL_FUNC**: For details, see the input parameter description of the gstrace_unreg_all_func() function. • **LIST_TRACE_SQL**: For details, see the input parameter description of the gstrace_list_configured_sql() function. • **LIST_TRACE_FUNC**: For details, see the input parameter description of the gstrace_list_configured_func() function. • **SHOW_TRACED_FUNC**: For details, see the description of the input parameters of the gstrace_show_traced_func() function. • **SHOW_MANGLED_FUNC**: For details, see the description of the input parameters of the gstrace_show_mangled_func() function. • **SHOW_SUBFUNC**: For details, see the input parameter description of the gstrace_show_subfunc() function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_parse_page_bypath(path text, blocknum bigint, relation_type text, read_memory boolean)

  Description: Parses a specified table page and returns the path for storing the parsed content.

  Return type: text.

  Note: Only SYSADMIN or OPRADMIN can execute this function. For pages that are not flushed to disks, the actual pages on the disks are empty. Therefore, the parsing result is empty, and the **relation_type** parameter does not take effect.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-206 gs_parse_page_bypath parameters

Parameter	Type	Description
path	text	• For an ordinary table or a segment-page ordinary table, the relative path is *Tablespace name*/*Database OID*/*Relfilenode of the table (physical file name)*, for example, **base/16603/16394**. • For the visibility map of an ordinary table, the relative path is *Tablespace name*/*Database OID*/*Visibility map of the ordinary table*. For example, **base/16603/16394_vm**. • For clog files, the parsed content is stored in the **pg_clog** directory. For example: 000000000000. • For csnlog files, the parsed content is stored in the **pg_csnlog** directory. For example: 000000000000. • For undo record files, the relative path is **undo/UNDOPERSISTENCE/zoneid.segno** in the **undo** directory. Example: undo/permanent/00000.0000009. • For undo meta files, the relative path is **undo/UNDOPERSISTENCE/zoneid.meta.segno** in the **undo** directory. Example: undo/permanent/00000.meta.0000004. • You can run the **pg_relation_filepath(table_name text)** command to query the relative path of the table file. To obtain the path of the partitioned table, view the pg_partition system catalog and call **pg_partition_filepath(partition_oid)**. • Valid path formats are as follows: – global/relNode – base/dbNode/relNode – pg_tblspc/spcNode/version_dir/dbNode/relNode

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
blocknum	bigint	• **–1**: information about all blocks (forcibly parsed from disks) • **0**–*MaxBlockNumber*: information about the corresponding block • For B-tree/UB-tree indexes, **0** indicates the index meta-page. • For undo record files, the logical block number is used. The block number of the corresponding file is greater than or equal to **segno** x 128 and less than (**segno** + 1) x 128. • For undo meta files, the logical block number is used. The block number of the corresponding file is greater than or equal to **segno** x 4 and less than (**segno** + 1) x 4.
relation_type	text	• **heap**: Astore table • **uheap**: Ustore table • **btree**: B-tree index • **ubtree**: UB-tree index • **vm**: visibility map of the Astore ordinary table • **fsm**: freespace map file of a row-store table • **clog** (commit log): transaction status log • **csnlog** (commit sequence number log): snapshot timestamp log • **undo_slot**: transaction slot information • **undo_record**: undo record information • **indexurq**: indexurq page • **ugin_index**: UGIN • **gin_index**: GIN • **gist_index**: GiST index • **cmap**: cross-page transparent compression metadata • **pg_dw_turbo_compress**: doublewrite buffer for cross-page transparent compression
read_memory	Boolean	• **false**: The system parses the page from the disk file. • **true**: The system attempts to parse the page from the shared buffer. If the page does not exist in the shared buffer, the system parses the page from the disk file.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
# Parse the information of all pages in the B-tree index file.
# Before calling the function, ensure that the file path exists based on the parameter description.
gaussdb=# SELECT gs_parse_page_bypath('base/16603/16394', -1, 'btree', false);
                      gs_parse_page_bypath
-----------------------------------------------------------------------
 gs_log_dir/dump/1663_16603_16394_-1.page
(1 row)

# Parse the visibility result of all blocks in the visibility map file.
gaussdb=# SELECT gs_parse_page_bypath('base/12828/16771_vm', -1, 'vm', false);
                       gs_parse_page_bypath
--------------------------------------------------------------------------
 gs_log_dir/dump/1663_12828_16771_-1_vm.page
(1 row)

# Parse the commit log of block 0 in the Clog file.
gaussdb=# SELECT gs_parse_page_bypath('000000000000', 0, 'clog', false);
                      gs_parse_page_bypath
-----------------------------------------------------------------------
 gs_log_dir/dump/000000000000.clog
(1 row)
```

The following is an example of an exception error:

```
# An error is reported when the value of the block number exceeds the value range.
gaussdb=# SELECT gs_parse_page_bypath('base/12828/16777', -10, 'heap', false);
ERROR:  Blocknum should be between -1 and 4294967294.
CONTEXT:  referenced column: gs_parse_page_bypath
```

- gs_xlogdump_lsn(start_lsn text, end_lsn text)

  Description: Parses Xlogs within the specified LSN range and returns the path for storing the parsed content. You can use pg_current_xlog_location() to obtain the current Xlog position.

  Parameters: **start_lsn** (LSN start position) and **end_lsn** (LSN end position).

  Return type: text.

  Note: Only SYSADMIN or OPRADMIN can execute this function.

- gs_xlogdump_xid(c_xid xid)

  Description: Parses Xlogs of a specified XID and returns the path for storing the parsed content. You can use txid_current() to obtain the current XID.

  Parameter: **c_xid** (transaction ID).

  Return type: text.

  Note: Only SYSADMIN or OPRADMIN can execute this function.

- gs_xlogdump_tablepath(path text, blocknum bigint, relation_type text)

  Description: Parses logs corresponding to a specified table page and returns the path for storing the parsed content.

  Return type: text.

  Note: Only SYSADMIN or OPRADMIN can execute this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-207 gs_xlogdump_tablepath parameters

Parameter	Type	Description
path	text	• For an ordinary table or a segment-page ordinary table, the relative path is *Tablespace name*/*Database OID*/*Relfilenode of the table (physical file name)*. For example, **base/16603/16394**. • For a segment-page hash bucket table, the relative path is *Tablespace name*/*Database OID*/*Logical page number of the segment head*_**b(bucketid)**, for example, **base/16603/16394_b1437**. • You can run the **pg_relation_filepath(table_name text)** command to query the relative path of the table file. To obtain the path of the partitioned table, view the pg_partition system catalog and call **pg_partition_filepath(partition_oid)**. • Valid path formats are as follows: – global/relNode – base/dbNode/relNode – pg_tblspc/spcNode/version_dir/dbNode/relNode For hash bucket tables, add _**b(bucketid)** to the end of the path.
blocknum	bigint	• **–1**: information about all blocks (forcibly parsed from disks) • **0**–*MaxBlockNumber*: information about the corresponding block
relation_type	text	• **heap**: Astore table • **btree**: B-tree index • **segment**: segment-page table

- gs_xlogdump_parsepage_tablepath(path text, blocknum bigint, relation_type text, read_memory boolean)

  Description: Parses the specified table page and logs corresponding to the table page and returns the path for storing the parsed content. It can be regarded as one execution of **gs_parse_page_bypath** and **gs_xlogdump_tablepath**. The prerequisite for executing this function is that the table file exists. To view logs of deleted tables, call **gs_xlogdump_tablepath**.

  Return type: text.

  Note: Only SYSADMIN or OPRADMIN can execute this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-208 gs_xlogdump_parsepage_tablepath parameters

Parameter	Type	Description
path	text	• For an ordinary table or a segment-page ordinary table, the relative path is *Tablespace name*/*Database OID*/*Relfilenode of the table (physical file name)*, for example, **base/16603/16394**. • You can run the **pg_relation_filepath(table_name text)** command to query the relative path of the table file. To obtain the path of the partitioned table, view the pg_partition system catalog and call **pg_partition_filepath(partition_oid)**. • Valid path formats are as follows: – global/relNode – base/dbNode/relNode – pg_tblspc/spcNode/version_dir/dbNode/relNode For hash bucket tables, add **_b(bucketid)** to the end of the path.
blocknum	bigint	• **–1**: information about all blocks (forcibly parsed from disks) • **0**–*MaxBlockNumber*: information about the corresponding block
relation_type	text	• **heap**: Astore table • **btree**: B-tree index • **indexurq**: indexurq page
read_memory	Boolean	• **false**: The system parses the page from the disk file. • **true**: The system attempts to parse the page from the shared buffer. If the page does not exist in the shared buffer, the system parses the page from the disk file.

- gs_shared_storage_xlogdump_lsn(start_lsn text, end_lsn text, xlog_file_path text)

  Description: Parses Xlogs within the specified LSN range of a Xlog shared disk and returns the path for storing the parsed content. You can use pg_current_xlog_location() to obtain the current Xlog position.

  Return type: text.

  Note: Only SYSADMIN or OPRADMIN can execute this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-209 gs_shared_storage_xlogdump_lsn input parameters

Parameter	Type	Description
start_lsn	text	Start LSN of the Xlog to be parsed.
end_lsn	text	End LSN of the Xlog to be parsed.
xlog_file_path	text	(Optional) Path of the shared Xlog disk where the Xlog to be parsed is located. If not specified, Xlogs of all shared disks on the current CN/DN are parsed.

Examples:

```
# Two input parameters are supported: start_lsn and end_lsn.
gaussdb=# SELECT * FROM
gs_shared_storage_xlogdump_lsn('00000000/01000000','00000000/19B08128');
                        output_filepath
-------------------------------------------------------------------
 /var/lib/engine/data1/data/dn_6001/01000000_19b08128_6001.xlog
(1 row)

# Three input parameters are supported: start_lsn, end_lsn, and xlog_file_path.
gaussdb=# SELECT * FROM
gs_shared_storage_xlogdump_lsn('00000000/01000000','00000000/19B08128','/data/nfs/6001');
                        output_filepath
-------------------------------------------------------------------
 /var/lib/engine/data1/data/dn_6001/01000000_19b08128_6001.xlog
(1 row)
```

- gs_shared_storage_ctlinfo()

  Description: Obtains information about the metadata of the shared disk on the current CN/DN.

  Return type: record.

Table 1-210 gs_shared_storage_ctlinfo output parameters

Parameter	Type	Description
xlog_file_path	text	Path of the Xlog shared disk.
system_id	text	The **system_id** value stored on the current shared disk.
term	text	The **term** value stored on the current shared disk.
xlog_file_size	text	Size of the Xlog space on the Xlog shared disk.
insert_tail	text	Minimum valid LSN on the Xlog shared disk.
insert_head	text	Maximum valid LSN on the Xlog shared disk.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
shared_storage_flush_lsn	text	Actual flushed LSN on the Xlog shared disk.

Examples:

```
gaussdb=# SELECT * FROM gs_shared_storage_ctlinfo();
 xlog_file_path |      system_id       | term | xlog_file_size |    insert_tail    |    insert_head    |
shared_storage_flush_lsn
----------------+----------------------+------+----------------+-------------------+-------------------
+--------------------------
 /data/nfs/6001 | 18020659526041888084 | 1    | 1073741824     | 00000000/01000000 |
00000000/19B08128 | 00000000/19B08128
(1 row)
```

- gs_index_verify(oid oid, uint32 blkno)

  Description: Checks whether the sequence of keys on the UB-tree index page or index tree is correct.

  Return type: record.

  Table 1-211 gs_index_verify parameters

Parameter	Type	Description
oid	oid	• Index file's relfilenode. It can be queried using **SELECT relfilenode FROM pg_class where relname=***'name'*, where *'name'* indicates the name of the index file.
blkno	uint32	• **0**: indicates that all pages in the index tree are checked. • If the value is greater than 0, the index page whose page code is equal to the value of **blkno** is checked.

- gs_index_recycle_queue(Oid oid, int type, uint32 blkno)

  Description: Parses the UB-tree index recycling queue information.

  Return type: record.

  Table 1-212 gs_index_recycle_queue parameters

Parameter	Type	Description
oid	oid	• Index file relfilenode, which can be queried using **select relfilenode from pg_class where relname='***Index file name***'**.
type	int	• **0**: The entire queue to be recycled is parsed. • **1**: The entire empty page queue is parsed. • **2**: A single page is parsed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
blkno	uint32	ID of the recycling queue page. This parameter is valid only when **type** is set to **2**. The value of **blkno** ranges from 1 to 4294967294.

- gs_stat_wal_entrytable(int64 idx)

  Description: Exports the content of the write-ahead log insertion status table in the Xlog.

  Return type: record.

  Table 1-213 gs_stat_wal_entrytable parameters

Category	Parameter	Type	Description
Input parameter	idx	int64	• **–1**: queries all elements in an array. • **0**–*Maximum value*: content of a specific array element.
Output parameter	idx	uint64	Records the indexes in the corresponding array.
Output parameter	endlsn	uint64	Records the LSN label. Determines whether the Xlog corresponding to the current entry has been completely copied to the WAL buffer. • **0**: Not copied. • Other values: Copied.
Output parameter	lrc	int32	Records the corresponding LRC.
Output parameter	status	uint32	This parameter is reserved in the current version. The query result is **0**.

- gs_walwriter_flush_position()

  Description: Outputs the refresh position of write-ahead logs.

  Return type: record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-214 gs_walwriter_flush_position parameters

Category	Parameter	Type	Description
Output parameter	last_flush_status_entry	int32	Index obtained after the Xlog flushes the tblEntry of the last flushed disk.
Output parameter	last_scanned_lrc	int32	LRC obtained after the Xlog flushes the last tblEntry scanned last time.
Output parameter	curr_lrc	int32	Latest LRC usage in the WALInsertStatusEntry status table. The LRC indicates the LRC value corresponding to the WALInsertStatusEntry when the next Xlog record is written.
Output parameter	curr_byte_pos	uint64	The latest Xlog position after the Xlog is written to the WAL file, which is also the next Xlog insertion point.
Output parameter	prev_byte_size	uint32	Length of the previous Xlog record.
Output parameter	flush_result	uint64	Position of the current global Xlog flush.
Output parameter	send_result	uint64	Xlog sending position on the current host.
Output parameter	shm_rqst_write_pos	uint64	The write position of the LogwrtRqst request in the XLogCtl recorded in the shared memory.
Output parameter	shm_rqst_flush_pos	uint64	The flush position of the LogwrtRqst request in the XLogCtl recorded in the shared memory.
Output parameter	shm_result_write_pos	uint64	The write position of the LogwrtResult request in the XLogCtl recorded in the shared memory.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	shm_result_flush_pos	uint64	The flush position of the LogwrtResult request in the XLogCtl recorded in the shared memory.
Output parameter	curr_time	text	Current time.

- gs_walwriter_flush_stat(int operation)

  Description: Collects statistics on the frequency of writing and synchronizing WALs, data volume, and Xlog file information.

  Return type: record.

  Table 1-215 gs_walwriter_flush_stat parameters

Category	Parameter	Type	Description
Input parameter	operation	int	• **–1** (default): disables the statistics function. • **0**: enables the statistics function. • **1**: queries statistics. • **2**: resets statistics.
Output parameter	write_times	uint64	Number of times that the Xlog calls the write API.
Output parameter	sync_times	uint64	Number of times that the Xlog calls the sync API.
Output parameter	total_xlog_sync_bytes	uint64	Total number of backend thread requests for writing data to Xlogs.
Output parameter	total_actual_xlog_sync_bytes	uint64	Total amount of Xlogs flushed when the sync API is called.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	avg_write_bytes	uint32	Amount of Xlogs written each time the XLogWrite API is called.
Output parameter	avg_actual_write_bytes	uint32	Amount of Xlogs written each time the write API is called.
Output parameter	avg_sync_bytes	uint32	Average amount of Xlogs for each sync request.
Output parameter	avg_actual_sync_bytes	uint32	Actual amount of Xlogs flushed each time sync is called.
Output parameter	total_write_time	uint64	Total time of calling the write operation (unit: μs).
Output parameter	total_sync_time	uint64	Total time for calling the sync API (unit: μs).
Output parameter	avg_write_time	uint32	Average time for calling the write API each time (unit: μs).
Output parameter	avg_sync_time	uint32	Average time for calling the sync API each time (unit: μs).
Output parameter	curr_init_xlog_segno	uint64	ID of the latest Xlog segment file.
Output parameter	curr_open_xlog_segno	uint64	ID of the Xlog segment file that is being written.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	last_reset_time	text	Time when statistics were last collected.
Output parameter	curr_time	text	Current time.

- pg_ls_tmpdir()

  Description: Returns the name, size, and last modification time of each file in the temporary directory (**pgsql_tmp**) of the default tablespace.

  Parameter: nan.

  Return type: record.

  Note: Only SYSADMIN or MONADMIN can execute this function.

Category	Parameter	Type	Description
Output parameter	name	text	File name.
Output parameter	size	int8	File size (unit: byte).
Output parameter	modification	timestamptz	Last file modification time.

- pg_ls_tmpdir(oid)

  Description: Returns the name, size, and last modification time of each file in the temporary directory (**pgsql_tmp**) of the specified tablespace.

  Parameter: oid.

  Return type: record.

  Note: Only SYSADMIN or MONADMIN can execute this function.

Category	Parameter	Type	Description
Input parameter	oid	oid	Tablespace ID.
Output parameter	name	text	File name.
Output parameter	size	int8	File size (unit: byte).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	modification	timestamptz	Last file modification time.

- pg_ls_waldir()

  Description: Returns the name, size, and last modification time of each file in the WAL directory.

  Parameter: nan.

  Return type: record.

  Note: Only SYSADMIN or MONADMIN can execute this function.

Category	Parameter	Type	Description
Output parameter	name	text	File name.
Output parameter	size	int8	File size (unit: byte).
Output parameter	modification	timestamptz	Last file modification time.

- gs_stat_anti_cache()

  Description: Returns AntiCache statistics.

  Parameter: nan.

  Return type: record.

Category	Parameter	Type	Description
Output parameter	table_nums	int8	Number of partitions.
Output parameter	table_init_size	int8	Initial size of a partition.
Output parameter	table_max_size	int8	Upper limit of the partition size.
Output parameter	anti_cache_upper_limit_size	int8	Upper limit of the AntiCache size.
Output parameter	anti_cache_size	int8	Real-time AntiCache size.
Output parameter	anti_cache_max_table_size	int8	Real-time maximum partition size.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	anti_cache_min_table_size	int8	Real-time minimum partition size.

- gs_stat_vlog_buffer()

  Description: Returns verifyLog buffer statistics.

  Parameter: nan.

  Return type: record.

Category	Parameter	Type	Description
Output parameter	vbuffer_write_offset	int8	Vlog write location.
Output parameter	vbuffer_flushed_offset	int8	Vlog disk flushing location.
Output parameter	max_vbuffer_flushed_value	int8	Maximum number of bytes for vlog flushing.
Output parameter	min_vbuffer_flushed_vlaue	int8	Minimum number of bytes for vlog flushing.
Output parameter	ave_vbuffer_flushed_vlaue	int8	Average number of bytes for vlog flushing.
Output parameter	vbuffer_flush_latency	int8	Vlog refresh delay.

- gs_stat_vlog_related_io()

  Description: Returns I/O statistics about read and write operations on the verifylog file.

  Parameter: nan.

  Return type: record.

Category	Parameter	Type	Description
Output parameter	read_data_iops	float8	Number of pages read from files per second.
Output parameter	vlog_read_iops	float8	Number of pages read from vlogs per second.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	vlog_write_iops	float8	Number of pages written to vlogs per second.

- gs_stat_vlog_file()

  Description: Queries the Verifylog (Vlog) files that have not been recycled.

  Parameter: nan.

  Return type: record.

Category	Parameter	Type	Description
Output parameter	last_recycled_segno	int8	Number of the last Vlog file segment that has been recycled.
Output parameter	target_recycle_segno	int8	Number of the target Vlog file segment to be recycled.
Output parameter	max_segno	int8	Maximum number of Vlog file segments.

- pause_anti_cache_recycle(boolean pause)

  Description: Specifies whether to suspend AntiCache recycling. This function is for internal use only.

  Parameter: **pause** specifies whether to suspend AntiCache recycling. The value is of the Boolean type. If **pause** is set to **true**, AntiCache recycling is suspended. If **pause** is set to **false**, AntiCache recycling is normal.

  Return type: text.

  Examples:

```
# In the internal version, set the AntiCache to be recycled normally.
gaussdb=# select * from pause_anti_cache_recycle(false);
           result
----------------------------
 Anticache recycle resumed.
(1 row)

# In the internal version, set the AntiCache recycling to be suspended.
gaussdb=# select * from pause_anti_cache_recycle(true);
          result
--------------------------
 Anticache recycle paused.
(1 row)

# This parameter cannot be set in the official version.
gaussdb=# select * from pause_anti_cache_recycle(false);
                        result
-------------------------------------------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
This function is not allowed to use in release version.
(1 row)
```

- gs_write_term_log(void)

  Description: Writes a log to record the current **term** value of a DN. The standby DN returns **false**. After the data is successfully written to the primary DN, **true** is returned.

  Return type: Boolean.

- gs_stat_space(bool init)

  Description: Queries the status of extended pages when the INSERT operation is performed on a Ustore table.

  Return type: record.

Category	Parameter	Type	Description
Input parameter	init	bool	Specifies whether to reset the statistics.
Output parameter	access_func	int8	Total number of access times of the relation_get_buffer_for_utuple API.
Output parameter	cache_blk	int8	Number of times that the relation_get_buffer_for_utuple API obtains buffers.
Output parameter	cache_succ	int8	Number of times that the relation_get_buffer_for_utuple API successfully obtains buffers.
Output parameter	nblk_first	int8	Number of times that relation_get_buffer_for_utuple obtains **nblocks-1** for the first time.
Output parameter	nblk_first_succ	int8	Number of times that relation_get_buffer_for_utuple obtains **nblocks-1** successfully for the first time.
Output parameter	nblk_second	int8	Number of times that relation_get_buffer_for_utuple obtains **nblocks-1** for the second time.
Output parameter	nblk_second_succ	int8	Number of times that relation_get_buffer_for_utuple obtains **nblocks-1** successfully for the second time.
Output parameter	fsm_first	int8	Number of times that FSM is accessed for the first time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	fsm_first_success	int8	Number of times that FSM is accessed successfully for the first time.
Output parameter	fsm_rewrite	int8	Number of FSM writeback times.
Output parameter	fsm_second	int8	Number of times that FSM is accessed for the second time.
Output parameter	fsm_second_success	int8	Number of times that FSM is accessed successfully for the second time.
Output parameter	prune_count	int8	Number of prune times in relation_get_buffer_for_utuple.
Output parameter	prune_space	int8	Total prune space in relation_get_buffer_for_utuple.
Output parameter	coprune_count	int8	Number of times that joint prune is performed.
Output parameter	coprune_scan_blocks	int8	Total number of pages scanned for joint prune.
Output parameter	coprune_prune_count	int8	Number of joint prune times.
Output parameter	coprune_prune_space	int8	Total space for joint prune.
Output parameter	con_extend_count	int8	Number of concurrent extended pages.
Output parameter	con_extend_time	int8	Total time for concurrent page extension.
Output parameter	single_extend_time	int8	Number of single-page extension times.

Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT * FROM gs_stat_space(false);
 access_func | cache_blk | cache_succ | nblk_first | nblk_first_succ | nblk_second | nblk_second_succ |
fsm_first | fsm_first_success | fsm_rewrite | fsm_
second | fsm_second_success | prune_count | prune_space | coprune_count | coprune_scan_blocks |
coprune_prune_count | coprune_prune_space | con_extend_count
 | con_extend_time | single_extend_count
-------------+-----------+------------+------------+-----------------+--------------+-------------------+-----------
+-------------------+-------------+-----
-------+-------------------+-------------+-------------+---------------+--------------------+--------------------
+---------------------+-----------------
-+-----------------+--------------------
    10082397 |  10082397 |   10082235 |        88 |            78 |       47021 |           47021 |       162
|              74 |      360996 |
360996 |           270948 |       6711 |          0 |        45497 |           222619 |                0
|                0 |            3675
 |       25542884 |             26791
(1 row)
```

Note: Pay special attention to the value of **cache_succ**. If it is small, the system cache is invalid. If the value of **prune_space** is small, the Ustore data page cleaning mechanism may be faulty. If the value of **con_extend_time** is too large, the Ustore concurrent page extension takes a long time.

- gs_redo_upage(directory_path text, backup_path text, blocknum bigint, relation_type text, xlog_path text, lsn text)

  Description: Redoes a specific Ustore data page that is backed up to a specified LSN and verifies the page in this process. If a damaged page is detected, the page is flushed to the disk and the disk flushing path, page LSN, and damage information is returned; otherwise, the page is redone to the specified LSN and flushed to the disk, and the result is returned. Only SYSADMIN or OPRADMIN can execute this function.

  Return type: record.

Category	Parameter	Type	Description
Input parameter	directory_path	text	Specifies the directory for storing the backup file.
Input parameter	backup_path	text	Relative path of the backup table file, which is combined with the directory where the backup file is located to form the complete path of the table file, for example, **base/15635/12488**. If the backup file does not exist, set this parameter to null.
Input parameter	blocknum	bigint	**0** to *MaxBlockNumber*: block number of the corresponding page.
Input parameter	relation_type	text	• **uheap**: Ustore data page • **ubtree**: Ustore index page • **indexurq**: URQ page • **undo_record**: undo record page • **undo_slot**: transaction slot page
Input parameter	xlog_path	text	Absolute path of the archive log directory.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Input parameter	lsn	text	The LSN consists of two hexadecimal numbers (32 bits each), which are separated by a slash (/), for example, 2/962D1DF8. If the value is **0**, the latest version is used.
Output parameter	output_filename	text	Path and name of the file to be flushed to the disk.
Output parameter	output_lsn	text	LSN of the last page redo.
Output parameter	corruption_desc	text	Page damage description.

- gs_xlogdump_bylastlsn(last_lsn text, blocknum bigint, relation_type text)

  Description: Inputs a page LSN and block number, parses the WAL corresponding to the LSN, obtains the last LSN of the corresponding block number, continues parsing until the last LSN is 0 or the WAL of an earlier version has been reused and recycled, and flushes the parsed log to a specified path. Only SYSADMIN or OPRADMIN can execute this function. This system function cannot be called by the standby node.

  Return type: text.

Category	Parameter	Type	Description
Input parameter	last_lsn	text	Parses the LSN of a specified page in hexadecimal notation, for example, 12BA/32CDEDDD. The LSN can be obtained using a page parsing tool (such as gs_parse_page_bypath).
Input parameter	blocknum	bigint	Specifies the logical block number of a page. Value range: **–1** to *MaxBlockNumber*. If the block number is set to **–1**, the default block number is obtained from the WAL.
Input parameter	relation_type	text	Specifies the type of the page to be parsed. Valid value: **uheap**, **ubtree**, **heap**, **btree**, **undo_record**, and **undo_slot**.
Output parameter	output_filepath	text	Specifies the path for flushing WAL parsing results to disks.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
# Obtain the page LSN.
# Before calling the function, ensure that the file path exists based on the parameter description.
gaussdb=# SELECT * FROM gs_parse_page_bypath('base/15833/16768', 0, 'uheap', false);
                    output_filepath
-----------------------------------------------------------------
 data1/database/cluster/primary/data/1663_15833_16768_0.page
(1 row)
gaussdb=# SELECT * FROM gs_xlogdump_bylastlsn('0/4593570', -1, 'uheap');
                    output_filepath
-----------------------------------------------------------------------
 /data1/database/cluster/primary/data/gs_log/dump/4593570_-1.xlog
(1 row)
gaussdb=# SELECT * FROM gs_xlogdump_bylastlsn('0/4593570', 0, 'ubtree');
ERROR:  The input lsn 0/4593570 related xlog is not ubtree.
```

- dbe_perf.get_full_sql_by_parent_id_and_timestamp(parent_id bigint, start_timestamp timestamp with time zone, end_timestamp timestamp with time zone)

  Description: Obtains the records of database-level full SQL statements and their substatements that execute a stored procedure in a specified period based on **parent_id**. The result can be queried only in the system database but cannot be queried in the user database.

  Return type: record.

Parameter	Type	Description
parent_id	bigint	Specifies the unique_sql_id of the statement calling a stored procedure.
start_timestamp	timestamp with time zone	Start point of the SQL start time range.
end_timestamp	timestamp with time zone	End point of the SQL start time range.

  Example:

```
gaussdb=# CREATE TABLE test(a int,b int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# INSERT INTO test values(1,1);
INSERT 0 1
gaussdb=# CREATE PROCEDURE mypro1() as num int;
gaussdb$# begin
gaussdb$# INSERT INTO test values(2,2);
gaussdb$# DELETE FROM test where a = 2;
gaussdb$# end;
gaussdb$# /
CREATE PROCEDURE

-- Enable the parameter to trace the substatements of the stored procedure.
gaussdb=# SET instr_unique_sql_track_type = 'all';
SET

-- Enable the parameter. Full statement records are generated in the dbe_perf.statement_history
table.
gaussdb=# SET track_stmt_stat_level = 'L0,L0';
SET

gaussdb=# CALL mypro1();
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 mypro1
--------

(1 row)

gaussdb=# SET track_stmt_stat_level = 'off,L0';
SET

gaussdb=# SET instr_unique_sql_track_type = 'top';
SET

-- Query key information, which is used as a function parameter.
gaussdb=# SELECT query,unique_query_id,start_time,finish_time FROM dbe_perf.statement_history;
                query                 | unique_query_id |          start_time           |          finish_time
--------------------------------------+-----------------+-------------------------------+-------------------------------
 set track_stmt_stat_level = 'L0,L0'; |       636388010 | 2023-06-02 17:40:49.176155+08 | 2023-06-02 17:40:49.176543+08
 call mypro1();                       |       536458473 | 2023-06-02 17:40:59.028144+08 | 2023-06-02 17:40:59.032027+08
 delete from test where a = ?         |       583323884 | 2023-06-02 17:40:59.029955+08 | 2023-06-02 17:40:59.031577+08
 insert into test values(?,?)         |       769279931 | 2023-06-02 17:40:59.029219+08 | 2023-06-02 17:40:59.029947+08
(4 rows)

-- Use unique_query_id, start time, and end time of the outer query statement as parameters to query information about the specified stored procedure and its substatements within the period.
gaussdb=# SELECT query FROM
dbe_perf.get_full_sql_by_parent_id_and_timestamp(536458473,'2023-06-02 17:40:59.028144+08','2023-06-02 17:40:59.032027+08');
            query
------------------------------
 call mypro1();
 delete from test where a = ?
 insert into test values(?,?)
(3 rows)

gaussdb=# DROP PROCEDURE mypro1();
DROP PROCEDURE
gaussdb=# DROP TABLE test;
DROP TABLE
```

- gs_index_dump_read(int8 reset, text out_type)

  Description: Queries the buffer read information generated in the cyclic queue when an index is used to obtain a new page. The buffer read information traverses leaf pages from left to right using the same key as the index page.

  Return type: record.

Category	Parameter	Type	Description
Input parameter	reset	int8	• **0**: The statistics are reset to the initial value 0 and the statistics are collected again. • **1**: The current statistics are displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Input parameter	out_type	text	• **urq**: outputs statistics about cyclic queues. • **ubtree**: outputs statistics on index pages. • **all**: outputs all statistics about cyclic queues and index pages.
Output parameter	relfilenode	oid	Index relfilenode corresponding to the maximum buffer read value.
Output parameter	max_count	int8	Maximum buffer read value.
Output parameter	ave_count	int8	Average buffer read value.

NOTE

- Currently, this API supports only Ustore index tables.
- When this API is executed, the reset operation is performed to clear all records and set all records to **0**. If you query the information again, the query result is always 0 until the information is collected next time. The following are query examples:

```
gaussdb=# SELECT * FROM gs_index_dump_read(0, 'all');
relfilenode | max_count | ave_count
--------------+-------------+-------------
|                 |
(1 row)
gaussdb=# SELECT * FROM gs_index_dump_read(1, 'all');
relfilenode | max_count | ave_count
--------------+-------------+-------------
0 |                 0 |                 0
0 |                 0 |                 0
(2 rows)
```

- pg_get_ilmdef(pidx integer)

  Description: Specifies a row-store compression system function, which returns the corresponding policy information based on the input ILM policy index.

  Return type: text.

Category	Parameter	Type	Description
Input parameter	pidx	integer	Sequence number of an ILM policy.
Output parameter	ilm_policy_info	text	Specifies the policy information of an ILM policy.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_shared_storage_flush_stat(int operation)

  Description: Collects statistics on the amount of written data, write speed, write time, amount of read data, read speed, and read time of shared disks.

  Return type: record.

Category	Parameter	Type	Description
Input parameter	operation	int	• **–1**: disables the statistics function. • **0** (default): enables the statistics function. • **1**: queries statistics. • **2**: resets statistics. After the statistics function is disabled and then enabled again, the previous statistics are cleared.
Output parameter	stat_switch	bool	Specifies whether the statistics function is enabled.
Output parameter	write_times	uint64	Number of times that the shared_storage_xlog_copy thread calls the dorado_write_xlog API.
Output parameter	avg_write_bytes	uint32	Amount of Xlogs to be written each time the dorado_write_xlog API is called, in byte.
Output parameter	avg_actual_write_bytes	uint32	Actual amount of Xlogs to be written each time the dorado_write_xlog API is called, in byte.
Output parameter	total_write_time	uint64	Total time for calling the dorado_write_xlog API, in μs.
Output parameter	avg_write_time	uint32	Average time for calling the dorado_write_xlog API each time, in μs.
Output parameter	avg_write_speed	uint32	Average speed of calling the dorado_write_xlog API to write Xlogs, in KB/s.
Output parameter	avg_actual_write_speed	uint32	Actual average speed of calling the dorado_write_xlog API to write Xlogs, in KB/s.
Output parameter	total_write_sleep_time	uint64	Total sleep time of the shared_storage_xlog_copy thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	read_times	uint64	Number of times that the walreceiver thread calls the dorado_read_xlog API.
Output parameter	avg_read_bytes	uint32	Amount of Xlogs to be written each time the dorado_read_xlog API is called, in byte.
Output parameter	avg_actual_read_bytes	uint32	Actual amount of Xlogs to be written each time the dorado_read_xlog API is called, in byte.
Output parameter	total_read_time	uint64	Total time for calling the dorado_read_xlog API, in μs.
Output parameter	avg_read_time	uint32	Average time for calling the dorado_read_xlog API each time, in μs.
Output parameter	avg_read_speed	uint32	Average speed of calling the dorado_read_xlog API to write Xlogs, in KB/s.
Output parameter	avg_actual_read_speed	uint32	Actual average speed of calling the dorado_read_xlog API to write Xlogs, in KB/s.
Output parameter	total_read_sleep_time	uint64	Total sleep time of the walreceiver thread.
Output parameter	stat_start_time	timestamptz	Start time of the statistics. If the statistics are not reset, the value is the start time of GaussDB. If the statistics are reset, the value is the last reset time.
Output parameter	stat_end_time	timestamptz	End time of the current statistics collection. If the statistics collection function is enabled, the value is the current time. If the statistics collection function is disabled, the value is the last time when the statistics collection function is disabled.

Example:

```
gaussdb=# SELECT * FROM gs_shared_storage_flush_stat(1);
 stat_switch | write_times | avg_write_bytes | avg_actual_write_bytes | total_write_time |
avg_write_time | avg_write_speed | avg_actual_write_speed | total_write_sleep_time | read_times | avg
_read_bytes | avg_actual_read_bytes | total_read_time | avg_read_time | avg_read_speed |
avg_actual_read_speed | total_read_sleep_time |        stat_start_time        |        stat_end_time
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-------------+-------------+-----------------+------------------------+------------------+----------------
+-----------------+-----------------------+-----------------------+------------+----
------------+---------------------+----------------+--------------+---------------+-----------------------
+----------------------+-----------------------------+-----------------------
-------
 t           |       43644 | 8624            | 8624                  |        263922553 | 6047           | 1392           |
1392                  |           26918503404 |       7451 | 140
13          | 14013               |         5321082 | 714           | 19163          | 19163
|                    0 | 2023-12-18 07:18:39.766855+08 | 2023-12-18 16:29:22.757
535+08
(1 row)
```

- gs_stat_freespace(Oid relid, Oid partid, char type, int sample_ratio)

  Description: Estimates the bloat rate of a table, partitioned table, or index based on statistics or sampling, and outputs space usage statistics and bloat rate information.

  Return type: record.

Category	Parameter	Type	Description
Input parameter	relation_oid	OID	OID of a data table, main table of the partitioned table, or index to be queried.
Input parameter	partition_oid	OID	OID of a partitioned table, subpartitioned table, or local partitioned index. ● If **relation_oid** refers to a non-partitioned table, non-local partitioned index, or GPI, this parameter is meaningless. ● If **relation_oid** refers to a partitioned table, subpartitioned table, or local partitioned index, and **partition_oid** is set to **0**, information about all partitions will be output.
Input parameter	relation_type	CHAR	Table type: ● **'r'**: The target table is a row-store data table. ● **'i'**: The target table is a row-store index table.
Input parameter	sample_ratio	INT8	Sampling ratio. The value ranges from 0 to 100. ● When **sample_ratio** is set to **0**, **INFO** indicates the bloat rate estimated based on statistics. ● When the value of **sample_ratio** ranges from 1 to 100, **INFO** indicates the bloat rate estimated through sampling.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Input parameter	read_memory	BOOL	Specifies whether to read pages from the memory. • **true**: indicates that the page is read from the memory. • **false**: indicates that pages are read from disks. Currently, the value can only be **false**.
Output parameter	OID	OID	OID of the queried data table, partitioned table, subpartitioned table, or index.
Output parameter	INFO	TEXT	• When **sample_ratio** is set to **0**, **INFO** indicates the estimated page bloat rate. • When the value of **sample_ratio** is greater than 0, **INFO** indicates the page usage statistics based on sampling, including the page header length, average tuple length, average number of normal tuples, average number of dead tuples, average number of unused tuples, average number of redirected tuples, and average number of TDs per page (for Ustore data tables), average potential_freespace per page, actual freespace (page upper – page lower), total number of pages in the relation, estimated space occupied by data tables/indexes based on sampling information, and estimated bloat rate of the data table/index based on sampling information.

NOTE

- Currently, this API supports only row-store tables and does not support time series tables, segment-page tables, hash bucket tables, local temporary tables, or global temporary tables. Indexes that do not support table types also do not support space statistics estimation.
- This API is used to query data on a CN. The query result only indicates the statistics and bloat rate estimation of the node, excluding the information about the corresponding DN. To query the DN statistics and bloat rate estimation result, you need to connect to the DN.
- When this API is used on the standby node and the value of **sample_ratio** is **0**, the estimation is based on statistics and the real-time accuracy cannot be ensured.

Example:

```
-- Query ordinary table information.
-- Create tables.
gaussdb=# DROP TABLE IF EXISTS t;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
NOTICE:  table "t" does not exist, skipping
DROP TABLE
gaussdb=# CREATE TABLE t(a int) WITH (storage_type=ustore);
CREATE TABLE
gaussdb=# INSERT INTO t VALUES(generate_series(1, 110000));
INSERT 0 110000
-- Update statistics.
gaussdb=# VACUUM ANALYZE t;
VACUUM
-- Estimate bloat rate.
gaussdb=# SELECT * FROM gs_stat_freespace((SELECT oid FROM pg_class WHERE relname='t'), 0, 'r',
0, false);
  oid  |                                                
```

```
-- Estimate bloat rate.
gaussdb=# SELECT * FROM gs_stat_freespace(16802, 16806, 'r', 0, false);
  oid  |                                                
```

```
rate: 0.83
(2 rows)

-- Sampling information:
gaussdb=# SELECT * FROM gs_stat_freespace(16802, 0, 'r', 60, false);
  oid  |
                        info

-------
+-------------------------------------------------------------------------------------------------------------------------
-------------------------
-------------------------------------------------------------------------------------------------------------------------
-------------------------------
-----------------------------------------
 16806 | Average page stat info: page_header_size: 56, tuple has storage: 249.92, normal tuple:
249.92, dead tuple: 0.00, unused tuple: 0.00, redirected tuple: 0.00, td_count: 4.00, potential_freespace:
1034.33, actual_freespace: 1034.33. Freespace stat info: nblocks: 12, approximate_table_size: 84636
(Bytes), approximate expansion rate: 1.16.
 16807 | Average page stat info: page_header_size: 56, tuple has storage: 259.30, normal tuple:
259.30, dead tuple: 0.00, unused tuple: 0.00, redirected tuple: 0.00, td_count: 4.00, potential_freespace:
771.70, actual_freespace: 771.70. Freespace stat info: nblocks: 27, approximate_table_size:
197532(Bytes), approximate expansion rate: 1.12.
(2 rows)

-- Query ordinary index information.
-- Create tables.
gaussdb=# DROP TABLE IF EXISTS t;
DROP TABLE
gaussdb=# CREATE TABLE t(a int, b int, c varchar(1000)) WITH (storage_type=ustore);
CREATE TABLE
gaussdb=# CREATE INDEX ON t(a);
CREATE INDEX
gaussdb=# INSERT INTO t VALUES(generate_series(1, 100000), generate_series(1, 100000),
'kjgkjature');
INSERT 0 100000

-- Update statistics.
gaussdb=# VACUUM ANALYZE t;
VACUUM

-- Estimate bloat rate.
gaussdb=# SELECT * FROM gs_stat_freespace((SELECT oid FROM pg_class WHERE relname='t_a_idx'),
0, 'i', 0, false);
  oid  |                                                                                                    info

-------
+-------------------------------------------------------------------------------------------------------------------------
-------------------------
-----------------------------------------------------------------------------------------------------
 16776 | live_tuples: 100000, dead_tuples: 0, origin_avg_tuple_len 4, max_align_avg_tuple_len: 8,
tuple_header_size: 16, rowptr_size: 401548, page_header_size: 24, nblocks: 387,
approximate_table_size: 2838700 (Bytes), approximate expansion rate: 1.12
(1 row)

-- Sampling information:
gaussdb=# SELECT * FROM gs_stat_freespace((SELECT oid FROM pg_class WHERE relname='t_a_idx'),
0, 'i', 60, false);
  oid  |
                        info

-------
+-------------------------------------------------------------------------------------------------------------------------
-------------------------
-------------------------------------------------------------------------------------------------------------------------
-------------------------------
-----------------------------------------
 16776 | Average page stat info: page_header_size: 24, tuple has storage: 260.39, normal tuple:
260.39, dead tuple: 0.00, unused tuple: 0.02, redirected tuple: 0.00, td_count: 0.00, potential_freespace:
0.00, actual_freespace: 845.05. Freespace stat info: nblocks: 387, approximate_table_size:
```

```
3044529(Bytes), approximate expansion rate: 1.04.
(1 row)

-- Query information about a single partition in a local partitioned index.
-- Create tables.
gaussdb=# DROP TABLE IF EXISTS t;
DROP TABLE
gaussdb=# CREATE TABLE t(a int) WITH (storage_type=ustore) PARTITION BY range(a) (PARTITION
p1 VALUES LESS THAN(3000), PARTITION p2 VALUES LESS THAN(maxvalue));
CREATE TABLE
gaussdb=# CREATE INDEX ON t(a) LOCAL;
CREATE INDEX
gaussdb=# INSERT INTO t VALUES(generate_series(1, 10000));
INSERT 0 10000

-- Update statistics.
gaussdb=# VACUUM ANALYZE t;
VACUUM

-- Query the OID.
gaussdb=# SELECT oid FROM pg_class WHERE relname='t_a_idx';
  oid
-------
 16814
(1 row)

gaussdb=# SELECT oid, relname, parttype FROM pg_partition;
  oid  | relname  | parttype
-------+----------+----------
 16811 | t        | r
 16812 | p1       | p
 16813 | p2       | p
 16815 | p1_a_idx | x
 16816 | p2_a_idx | x
(5 rows)

-- Estimate bloat rate.
gaussdb=# SELECT * FROM gs_stat_freespace(16814, 16815, 'i', 0, false);
  oid  |                                                
```

```
DROP TABLE
gaussdb=# CREATE TABLE t(a int) WITH (storage_type=ustore) PARTITION BY range(a) (PARTITION
p1 VALUES LESS THAN(3000), PARTITION p2 VALUES LESS THAN(maxvalue));
CREATE TABLE
gaussdb=# CREATE INDEX ON t(a) LOCAL;
CREATE INDEX
gaussdb=# INSERT INTO t VALUES(generate_series(1, 10000));
INSERT 0 10000
Update statistics:
gaussdb=# VACUUM ANALYZE t;
VACUUM
-- Query the OID.
gaussdb=# SELECT oid FROM pg_class WHERE relname='t_a_idx';
  oid
-------
 16814
(1 row)

-- Estimate bloat rate.
gaussdb=# SELECT * FROM gs_stat_freespace(16814, 0,  'i', 0, false);
  oid  |                                                
```

execute this function for multiple times until the returned result is 0. If the fastupdate option is not enabled for the index, the pending list will not be cleared, and the result is **0**.

Return type: bigint.

- gs_walrcv_writer_stat(int operation)

  Description: Collects statistics on the frequency of writing and synchronizing WALs received by the standby node, data volume, and Xlog file information.

  Return type: record.

Category	Parameter	Type	Description
Input parameter	operation	int	• **–1** (default): disables the statistics function. • **0**: enables the statistics function. • **1**: queries statistics. • **2**: resets statistics. After the statistics function is disabled and then enabled again, the previous statistics are cleared.
Output parameter	enable_stat	bool	Specifies whether the statistics function is enabled.
Output parameter	total_write_bytes	uint64	Total number of WAL files written to the standby node (unit: byte).
Output parameter	write_count	uint64	Number of times that the standby node writes WAL files.
Output parameter	total_write_time	uint64	Total time for writing WAL files to the standby node (unit: μs).
Output parameter	avg_write_time	uint64	Average time for calling the write API when the standby node writes WAL files each time (unit: μs).
Output parameter	max_write_time	uint64	Maximum duration for calling the write API when the standby node writes WAL files (unit: μs).
Output parameter	avg_write_bytes	uint64	Average value of the WAL file written to the standby node each time (unit: byte).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	total_sync_time	uint64	Total time for flushing WAL files on the standby node to disk (unit: μs).
Output parameter	avg_sync_time	uint64	Average time for calling the sync API when WAL files on the standby node are flushed to disk each time (unit: μs).
Output parameter	max_sync_time	uint64	Maximum duration for calling the sync API when the standby node writes WAL files (unit: μs).
Output parameter	current_xlog_segno	uint64	ID of the Xlog segment file that is being written on the standby node.
Output parameter	newest_xlog_segno	uint64	ID of the latest Xlog segment file on the standby node.
Output parameter	stat_start_time	timestamptz	Start time for collecting statistics.
Output parameter	stat_end_time	timestamptz	End time of the statistics collection.

Example:

```
gaussdb=# SELECT * FROM gs_walrcv_writer_stat(1);
-[ RECORD 1 ]------+------------------------------
enable_stat        | t
total_write_bytes  | 100754464
write_count        | 88426
total_write_time   | 3811972
avg_write_time     | 43
max_write_time     | 381739
avg_write_bytes    | 1139
total_sync_time    | 22280240
avg_sync_time      | 251
max_sync_time      | 655533
current_xlog_segno | 8154
newest_xlog_segno  | 8274
stat_start_time    | 2024-12-05 09:29:15.324159+08
stat_end_time      | 2024-12-05 09:29:45.92897+08
```

- gstrace_reg_sql(scope text, unique_sql_id text)

  Description: Adds the SQL statements to be traced. A maximum of 1,000 SQL statements can be traced for a database instance. After successful execution, the result (success or failure) of adding the SQL statements to be traced is returned. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following lists related parameters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Input Parameter	Description	Value Range
scope	Required. Scope of a function.	Text type. The value can be **'LOCAL'** or **'GLOBAL'**. **'LOCAL'** means that the configuration takes effect only on the current node, while **'GLOBAL'** means that it takes effect on all CNs and DNs.
unique_sql_id	Required. Unique SQL ID of the traced SQL statement to be added.	Text type. The **unique_query_id** value retrieved from the statement_history table.

Return type: text.

Example:

```
gaussdb=# SELECT pg_catalog.gstrace_reg_sql('LOCAL', '3545025713');
-[ RECORD 1 ]------+------------------------------
 (datanode,"ACTION:      [GSTRACE]->[TRACE_SQL]          +
 RESULT:                                              +
 SUCCESSFUL - etrace unique_sql_id (3545025713) tracked")
```

- gstrace_reg_func(scope text, function_name text, so_index text, so_offset text, level text)

  Description: Adds the functions to be traced. A maximum of 1,000 functions can be traced for a database instance. After successful execution, the result (success or failure) of adding the functions to be traced is returned. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following lists related parameters.

Input Parameter	Description	Value Range
scope	Required. Scope of a function.	Text type. The value can be **'LOCAL'** or **'GLOBAL'**. **'LOCAL'** means that the configuration takes effect only on the current node, while **'GLOBAL'** means that it takes effect on all CNs and DNs.
function_name	Required. Name of a traced function to be added.	Text type. Only the name of a system function is supported.
so_index	Optional. SO index of a function.	Text type. The **so_index** value of the system function retrieved using the pg_catalog.gstrace_show_mangled_func() function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Input Parameter	Description	Value Range
so_offset	Required when **so_index** is set; otherwise, it is not required. SO offset of a function.	Text type. The **so_offset** value of the system function retrieved using the pg_catalog.gstrace_show_mangled_func() function.
level	Required. Drill-down level of a function.	Text type. The value can be **'0'** or **'1'**. **'0'** indicates that only the function itself is traced, while **'1'** indicates that the function and its subfunctions are traced.

Return type: text.

Example 1: The input parameters do not contain the SO index and SO offset.

```
gaussdb=# SELECT pg_catalog.gstrace_reg_func('LOCAL', 'pg_sleep', '0');
------------------------------------------------------------------------------
 (datanode,"ACTION:      [GSTRACE]->[TRACE_FUNC]                          +
 RESULT:                                                                  +
 SUCCESSFUL - etrace function (_Z8pg_sleepP20FunctionCallInfoData) tracked")
```

Example 2: The input parameters contain the SO index and SO offset.

```
gaussdb=# SELECT pg_catalog.gstrace_show_mangled_func('LOCAL', 'pg_sleep');
                    gstrace_show_mangled_func
------------------------------------------------------------------------------
 (datanode,"ACTION:      [GSTRACE]->[SHOW_MANGLED_FUNC]                   +
 RESULT:                                                                  +
 SUCCESSFUL - function (pg_sleep)'s gangled func are these:               +
 function_name                          so_index so_name        so_offset+
 _Z8pg_sleepP20FunctionCallInfoData       0        gaussdb        59877694 +
 ")

gaussdb=# SELECT pg_catalog.gstrace_reg_func('LOCAL', 'pg_sleep', '0', '59877694', '0');
                       gstrace_reg_func
------------------------------------------------------------------------------
 (datanode,"ACTION:      [GSTRACE]->[TRACE_FUNC]                          +
 RESULT:                                                                  +
 SUCCESSFUL - etrace function (_Z8pg_sleepP20FunctionCallInfoData) tracked")
```

- gstrace_unreg_sql(scope text, unique_sql_id text)

  Description: Deletes a traced SQL statement. After successful execution, the result (success or failure) of deleting the traced SQL statement is returned. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following lists related parameters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Input Parameter	Description	Value Range
scope	Required. Scope of a function.	Text type. The value can be **'LOCAL'** or **'GLOBAL'**. **'LOCAL'** means that the configuration takes effect only on the current node, while **'GLOBAL'** means that it takes effect on all CNs and DNs.
unique_sql_id	Required. Unique SQL ID of the traced SQL statement to be deleted.	Text type. The **unique_query_id** value retrieved from the statement_history table.

Return type: text.

Example:

```
gaussdb=# SELECT pg_catalog.gstrace_unreg_sql('LOCAL', '3545025713');
                      gstrace_unreg_sql
-------------------------------------------------------------------
(datanode,"ACTION:      [GSTRACE]->[UNTRACE_SQL]              +
RESULT:                                                       +
SUCCESSFUL - etrace unique_sql_id (3545025713) delete tracked")
```

- gstrace_unreg_func(scope text, function_name text, so_index text, so_offset text)

  Description: Deletes a traced function. After successful execution, the result (success or failure) of deleting the traced function is returned. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following lists related parameters.

Input Parameter	Description	Value Range
scope	Required. Scope of a function.	Text type. The value can be **'LOCAL'** or **'GLOBAL'**. **'LOCAL'** means that the configuration takes effect only on the current node, while **'GLOBAL'** means that it takes effect on all CNs and DNs.
function_name	Required. Name of a traced function to be deleted.	Text type. Only the name of a system function is supported.
so_index	Optional. SO index of a function.	Text type. The **so_index** value of the system function retrieved using the pg_catalog.gstrace_show_mangled_func() function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Input Parameter	Description	Value Range
so_offset	Required when **so_index** is set; otherwise, it is not required. SO offset of a function.	Text type. The **so_offset** value of the system function retrieved using the pg_catalog.gstrace_show_mangled_func() function.

Return type: text.

Example 1: The input parameters do not contain the SO index and SO offset.

```
gaussdb=# SELECT pg_catalog.gstrace_unreg_func('LOCAL', 'pg_sleep');
                          gstrace_unreg_func
-------------------------------------------------------------------------------
 (datanode,"ACTION:      [GSTRACE]->[UNTRACE_FUNC]                          +
 RESULT:                                                                    +
 SUCCESSFUL - etrace function (_Z8pg_sleepP20FunctionCallInfoData) untracked")
```

Example 2: The input parameters contain the SO index and SO offset.

```
gaussdb=# SELECT pg_catalog.gstrace_show_mangled_func('LOCAL', 'pg_sleep');
                       gstrace_show_mangled_func
-------------------------------------------------------------------------------
 (datanode,"ACTION:      [GSTRACE]->[SHOW_MANGLED_FUNC]                     +
 RESULT:                                                                    +
 SUCCESSFUL - function (pg_sleep)'s gangled func are these:                 +
 function_name                          so_index so_name        so_offset+
 _Z8pg_sleepP20FunctionCallInfoData      0        gaussdb        59877694 +
 ")

gaussdb=# SELECT pg_catalog.gstrace_unreg_func('LOCAL', 'pg_sleep', '0', '59877694');
                          gstrace_unreg_func
-------------------------------------------------------------------------------
 (datanode,"ACTION:      [GSTRACE]->[UNTRACE_FUNC]                          +
 RESULT:                                                                    +
 SUCCESSFUL - etrace function (_Z8pg_sleepP20FunctionCallInfoData) untracked")
```

- gstrace_unreg_all_sql(scope text)

  Description: Deletes all traced SQL statements. After successful execution, the result (success or failure) of deleting the traced SQL statements is returned. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following lists related parameters.

Input Parameter	Description	Value Range
scope	Required. Scope of a function.	Text type. The value can be **'LOCAL'** or **'GLOBAL'**. **'LOCAL'** means that the configuration takes effect only on the current node, while **'GLOBAL'** means that it takes effect on all CNs and DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: text.

Example:

```
gaussdb=# SELECT pg_catalog.gstrace_unreg_all_sql('LOCAL');
              gstrace_unreg_all_sql
-----------------------------------------------------------
 (datanode,"ACTION:      [GSTRACE]->[UNTRACE_ALL_SQL]+
 RESULT:                                 +
 SUCCESSFUL - all traced SQL have been deleted!")
```

- gstrace_unreg_all_func(scope text)

  Description: Deletes all traced functions. After successful execution, the result (success or failure) of deleting the traced functions is returned. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following lists related parameters.

Input Parameter	Description	Value Range
scope	Required. Scope of a function.	Text type. The value can be **'LOCAL'** or **'GLOBAL'**. **'LOCAL'** means that the configuration takes effect only on the current node, while **'GLOBAL'** means that it takes effect on all CNs and DNs.

  Return type: text.

  Example:

```
gaussdb=# SELECT pg_catalog.gstrace_unreg_all_func('LOCAL');
              gstrace_unreg_all_func
-----------------------------------------------------------
 (datanode,"ACTION:      [GSTRACE]->[UNTRACE_ALL_FUNC]+
 RESULT:                                 +
 SUCCESSFUL - all tracked functions have been deleted!+
 ")
```

- gstrace_list_configured_sql(scope text)

  Description: Views the SQL statements that are being traced. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following lists related parameters.

Input Parameter	Description	Value Range
scope	Required. Scope of a function.	Text type. The value can be **'LOCAL'** or **'GLOBAL'**. **'LOCAL'** means that the configuration takes effect only on the current node, while **'GLOBAL'** means that it takes effect on all CNs and DNs.

  Return type: SETOF record

  The following describes the return columns.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
unique_sql_id	bigint	Unique SQL ID of the SQL statement to be traced.

Example:

```
gaussdb=# SELECT pg_catalog.gstrace_list_configured_sql('LOCAL');
            gstrace_list_configured_sql
-----------------------------------------------------
 (datanode,"ACTION:      [GSTRACE]->[LIST_TRACE_SQL]+
 RESULT:                                           +
 SUCCESSFUL - these unique_sql_id are tracking:     +
 unique_sql_id                                      +
 3545025713                                          +
 ")
```

- gstrace_list_configured_func(scope text)

  Description: Views the functions that are being traced. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following lists related parameters.

Input Parameter	Description	Value Range
scope	Required. Scope of a function.	Text type. The value can be **'LOCAL'** or **'GLOBAL'**. **'LOCAL'** means that the configuration takes effect only on the current node, while **'GLOBAL'** means that it takes effect on all CNs and DNs.

  Return type: SETOF record

  The following describes the return columns.

Column	Type	Description
function_name	text	Name of the function being traced.
so_index	int	SO index of a function. **0** corresponds to GaussDB bin, while **1** corresponds to **libgtmclient.so**.
so_name	text	SO name.
so_offset	int	SO offset of a function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
level	int	Drill-down level of a function. **0** indicates that only the function itself is traced, while **1** indicates that the function and its subfunctions are traced.

Example:

```
gaussdb=# SELECT pg_catalog.gstrace_list_configured_func('LOCAL');
                          gstrace_list_configured_func
-------------------------------------------------------------------------------------
(datanode,"ACTION:      [GSTRACE]->[LIST_TRACE_FUNC]                                +
RESULT:                                                                             +
SUCCESSFUL - these function are tracking:                                           +
function_name                       so_index so_name       so_offset       level+
_Z8pg_sleepP20FunctionCallInfoData      0       gaussdb        59877694        0    +
")
```

- gstrace_show_traced_func(scope text)

  Description: Views the status of the functions being traced. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following lists related parameters.

Input Parameter	Description	Value Range
scope	Required. Scope of a function.	Text type. The value can be **'LOCAL'** or **'GLOBAL'**. **'LOCAL'** means that the configuration takes effect only on the current node, while **'GLOBAL'** means that it takes effect on all CNs and DNs.

  Return type: SETOF record

  The following describes the return columns.

Column	Type	Description
function_name	text	Name of the function being traced.
so_index	int	SO index of a function. **0** corresponds to GaussDB bin, while **1** corresponds to **libgtmclient.so**.
so_name	text	SO name.
so_offset	int	SO offset of a function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
level	int	Drill-down level of a function. **0** indicates that only the function itself is traced, while **1** indicates that the function and its subfunctions are traced.
module	text	Module requesting to trace this function.
if_traced	bool	Specifies whether the function assembly has a jump tracing instruction.
if_hp_conflict	bool	Specifies whether the tracing is automatically canceled due to a conflict with the hot patch.
if_blacklist	bool	Specifies whether a function is in the traced function blacklist.
ref_count	int	Number of times that a function is traced.

Example:

```
gaussdb=# SELECT pg_catalog.gstrace_show_traced_func('LOCAL');
                                                
```

Input Parameter	Description	Value Range
function_name	Required. Name of a traced function to be added.	Text type. Only the name of a system function is supported.

Return type: SETOF record

The following describes the return columns.

Column	Type	Description
function_name	text	Name of the function being traced.
so_index	int	SO index of a function. **0** corresponds to GaussDB bin, while **1** corresponds to **libgtmclient.so**.
so_name	text	SO name.
so_offset	int	SO offset of a function.

Example:

```
gaussdb=# SELECT pg_catalog.gstrace_show_mangled_func('LOCAL',
'_Z8pg_sleepP20FunctionCallInfoData');
                            gstrace_show_mangled_func
-------------------------------------------------------------------------------------
 (datanode,"ACTION:       [GSTRACE]->[SHOW_MANGLED_FUNC]                          +
 RESULT:                                                                          +
 SUCCESSFUL - function (_Z8pg_sleepP20FunctionCallInfoData)'s gangled func are these:+
 function_name                          so_index so_name        so_offset         +
 _Z8pg_sleepP20FunctionCallInfoData       0        gaussdb        59877694          +
 '')
```

- gstrace_show_subfunc(scope text, function_name text, so_index text, so_offset text)

  Description: Views all subfunctions of a specified function. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following lists related parameters.

Input Parameter	Description	Value Range
scope	Required. Scope of a function.	Text type. **'LOCAL'** and **'GLOBAL'** are supported in distributed mode. **'LOCAL'** means that the configuration takes effect only on the current node, while **'GLOBAL'** means that it takes effect on all CNs and DNs.
function_name	Required. Name of a traced function to be added.	Text type. Only the name of a system function is supported.
so_index	Optional. SO index of a function.	Text type. The **so_index** value of the system function retrieved using the pg_catalog.gstrace_show_mangled_func() function.
so_offset	Required when **so_index** is set; otherwise, it is not required. SO offset of a function.	Text type. The **so_offset** value of the system function retrieved using the pg_catalog.gstrace_show_mangled_func() function.

Return type: SETOF record

The following describes the return columns.

Column	Type	Description
function_name	text	Name of the function being traced.
so_index	int	SO index of a function. **0** corresponds to GaussDB bin, while **1** corresponds to **libgtmclient.so**.
so_name	text	SO name.
so_offset	int	SO offset of a function.

Example:

```
gaussdb=# SELECT pg_catalog.gstrace_show_subfunc('LOCAL', '_Z8pg_sleepP20FunctionCallInfoData');
                                    gstrace_show_subfunc
-------------------------------------------------------------------------------------------------
 (datanode,"ACTION:      [GSTRACE]->[SHOW_SUBFUNC]                                              +
 RESULT:                                                                                        +
 SUCCESSFUL - function (_Z8pg_sleepP20FunctionCallInfoData)'s sub func are these:               +
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
function_name                                                so_index so_name        so_offset+
_Z14DatumGetFloat8m                                             0        gaussdb        68889422 +
_Z21get_current_timestampv                                      0        gaussdb        63347226 +
_ZL32optutil_gsstat_report_waitstatus23OptUtilExplainWaitStateb  0        gaussdb        59848101 +
_Z17ProcessInterruptsv                                          0        gaussdb        94358107 +
_Z9pg_usleepl                                                   0        gaussdb        55249960 +
")
```

- gs_last_demote_spent_time()

  Description: Obtains the time consumed in key steps of primary-to-standby switchover, such as thread exit and initialization, checkpoint, shared memory exit and initialization, doublewrite initialization, and buffer pool initialization. You need to query the primary-to-standby switchover information on the primary node of the cluster where the switchover is performed. After a successful switchover between clusters or within a cluster, you can query the related information. This information is stored in memory and will be reset after a restart.

  Return type: SETOF record

  The following describes the return columns.

Column	Type	Description
key_step_name	text	Key steps of primary-to-standby switchover: • **thread_exit**: The thread exits during primary-to-standby switchover. • **checkpoint**: flushes dirty page during primary-to-standby switchover. • **shmem_exit**: The shared memory exits during primary-to-standby switchover. • **shmem_init**: initializes the shared memory during primary-to-standby switchover. • **bufferpool_init**: initializes the buffer pool during primary-to-standby switchover. • **dw_init**: initializes the doublewrite during primary-to-standby switchover. • **thread_init**: initializes the thread during primary-to-standby switchover.
spent_time	int	Time consumed in key steps of primary-to-standby switchover.

  Example:

```
gaussdb=# SELECT * FROM gs_last_demote_spent_time();
  key_step_name  | spent_time
-----------------+------------
 thread_exit     |    1.00299
 checkpoint      |    3.19896
 shmem_exit      |    .397594
 shmem_init      |    1.74218
 bufferpool_init |    1.24905
 dw_init         |    .038082
 thread_init     |    .400442
(7 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_tcache_memory_flush()

  Description: Manually wakes up idle worker threads and stream worker threads, and executes **mallctl("thread.tcache.flush", NULL, NULL, NULL, 0);** on the corresponding threads to clear the jemalloc tcache cache.

  Return type: Boolean. If the operation is successful, **t** (true) is returned. If the operation fails, **f** (false) is returned.

  NOTE

  - This function takes effect only when the thread pool is enabled.
  - Only users with the SYSADMIN permission can use this function.
  - This function can reduce the other memory, but only reduces the jemalloc tcache. This function is invalid in scenarios where the other memory increases due to other causes (for example, the other memory increases due to memory fragments caused by a large number of hard parsing operations).

1.6.28 SPM Functions

CAUTION

Currently, the SPM feature is not supported in the distributed system. Therefore, when the following functions are called in the distributed system, an error message is displayed, indicating that the SPM feature is not supported in the distributed system.

- GS_SPM_EVOLUTE_PLAN(sql_hash, plan_hash)

  Description: GS_SPM_EVOLUTE_PLAN belongs to the DBE_SQL_UTIL schema and is a function used by the SPM feature to plan evolution.

  Parameters: For details, see **Table 1-216**.

  Table 1-216 GS_SPM_EVOLUTE_PLAN input parameters and return values

Parameter	Type	Description	Value Range
sql_hash	IN bigint	Specifies a hash value of SQL text.	-
plan_hash	IN bigint	Specifies a hash value of the outline text of the SQL plan.	-
evolute_status	OUT boolean	Determines whether the evolution is complete. **t** indicates normal. If an exception occurs, an error is reported.	t/f

  Return type: Boolean

- GS_SPM_SET_PLAN_STATUS(sql_hash, plan_hash, plan_status)

  Description: GS_SPM_SET_PLAN_STATUS belongs to the DBE_SQL_UTIL schema. It is an API used by the SPM feature to modify the baseline status.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

When the API takes effect, the **invalid** column in the gs_spm_baseline table is updated to **false**.

Parameters: For details, see **Table 1-217**.

Table 1-217 GS_SPM_SET_PLAN_STATUS input parameters and return values

Parameter	Type	Description	Value Range
sql_hash	IN bigint	Specifies a hash value of SQL text.	-
plan_hash	IN bigint	Specifies a hash value of the outline text of the SQL plan.	-
plan_status	IN text	Specifies the status of a plan.	• **ACC**: indicates that the plan has been accepted. • **UNACC**: indicates that the plan is not accepted. • **FIXED**: indicates a special ACC plan. The matching priority of this plan is higher than that of other ACC plans.
execute_status	OUT boolean	Determines whether the plan status change is complete. **t** indicates normal. If an exception occurs, an error is reported.	t/f

Return type: Boolean

- GS_SPM_DISPLAY_PLANS(sql_hash)

  Description: GS_SPM_DISPLAY_PLANS belongs to the DBE_SQL_UTIL schema and is a function used by the SPM feature to view all baselines of a single SQL statement.

  Parameters: For details, see **Table 1-218**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-218 GS_SPM_DISPLAY_PLANS input parameters and return values

Parameter	Type	Description	Value Range
sql_hash	IN bigint	Unique ID of an SQL statement.	-
sql_hash	OUT bigint	Specifies a hash value of SQL text.	-
plan_hash	OUT bigint	Specifies a hash value of the outline text of the SQL plan.	-
outline	OUT text	Specifies the combination character string of all hints of the outline corresponding to the current plan.	-
cost	OUT double	Specifies the cost of the current plan.	-
status	OUT text	Specifies the status of the current plan.	• **ACC**: indicates that the plan has been accepted. • **UNACC**: indicates that the plan is not accepted. • **FIXED**: indicates a special ACC plan. The matching priority of this plan is higher than that of other ACC plans.
gplan	OUT boolean	Determines whether the current plan is a gplan. **t** indicates gplan. If an exception occurs, an error is reported.	t/f

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description	Value Range
namespace_oids	OUT text	OID list in the current **search_path**. The OID must meet the following requirements: • OID of the schema on which the user has the permission. • OID of an existing schema. • OID of the schema where the non-local temporary table belongs.	-

Return type: bigint, text, double, text, boolean

- GS_SPM_RELOAD_PLAN(sql_hash, plan_hash)

  Description: GS_SPM_RELOAD_PLAN belongs to the DBE_SQL_UTIL schema and is a function used by the SPM feature to load a baseline in the baseline system catalog to the SPM global cache.

  Parameters: For details, see **Table 1-219**.

  Table 1-219 GS_SPM_RELOAD_PLAN input parameters and return values

Parameter	Type	Description	Value Range
sql_hash	IN bigint	Specifies a hash value of SQL text.	-
plan_hash	IN bigint	Specifies a hash value of the outline text of the SQL plan.	-
execute_status	OUT boolean	Determines whether the baseline loading is complete. **t** indicates normal. If an exception occurs, an error is reported.	t/f

  Return type: Boolean

- GS_SPM_VALIDATE_PLAN(sql_hash, plan_hash)

  Description: GS_SPM_VALIDATE_PLAN belongs to the DBE_SQL_UTIL schema and is a function used by the SPM feature to verify plan availability.

  Parameters: For details, see **Table 1-220**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-220 GS_SPM_VALIDATE_PLAN input parameters and return values

Parameter	Type	Description	Value Range
sql_hash	IN bigint	Specifies a hash value of SQL text.	-
plan_hash	IN bigint	Specifies a hash value of the outline text of the SQL plan.	-
execute_status	OUT boolean	Determines whether the verified plan is available: **t**: The verified plan is available. **f**: The verified plan is unavailable.	t/f

Return type: Boolean

- GS_SPM_DELETE_PLAN(sql_hash, plan_hash)

  Description: GS_SPM_DELETE_PLAN belongs to DBE_SQL_UTIL schema and is a function used by the SPM feature to delete the plan baseline.

  Parameters: For details, see **Table 1-221**.

Table 1-221 GS_SPM_DELETE_PLAN input parameters and return values

Parameter	Type	Description	Value Range
sql_hash	IN bigint	Specifies a hash value of SQL text.	-
plan_hash	IN bigint	Specifies a hash value of the outline text of the SQL plan.	-
execute_status	OUT boolean	Determines whether the plan deletion is complete. **t** indicates normal. If an exception occurs, an error is reported.	t/f

Return type: Boolean

1.6.29 Statistics Information Functions

Statistics information functions are divided into the following two categories: functions that access databases, using the OID of each table or index in a database to mark the database for which statistics are generated; functions that access servers, identified by the server thread ID, whose value ranges from 1 to the number of currently active servers.

- pg_stat_get_db_conflict_tablespace(oid)

  Description: Specifies the number of queries canceled due to a conflict between the restored tablespace and the deleted tablespace in the database.

  Return type: bigint.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_stat_get_db_conflict_truncate(oid)

  Description: Number of queries canceled due to conflicts with TRUNCATE operations that are being replayed in the database.

  Return type: bigint.

- pg_stat_get_db_conflict_force_recycle(oid)

  Description: Number of queries canceled due to forcible recycling triggered by read on standby (ROS) with ultimate RTO.

  Return type: bigint.

- pg_stat_get_db_conflict_standby_query_timeout(oid)

  Description: Number of queries canceled because the query time exceeds the upper limit set by ROS with ultimate RTO.

  Return type: bigint.

- pg_stat_get_db_conflict_bufferpin(oid)

  Description: Number of conflicting buffers.

  Return type: bigint.

- pg_stat_get_db_conflict_snapshot(oid)

  Description: Number of conflicting snapshots.

  Return type: bigint.

- pg_stat_get_db_conflict_startup_deadlock(oid)

  Description: Number of conflicting deadlocks.

  Return type: bigint.

- pg_stat_get_db_conflict_all(oid)

  Description: Number of queries canceled due to database recovery conflicts (conflicts occurring only on the standby node).

  Return type: bigint.

- pg_control_group_config()

  Description: Prints Cgroup configurations on the current node. Only users with the SYSADMIN permission can execute this function.

  Return type: record.

- pg_stat_get_db_stat_reset_time(oid)

  Description: Specifies the most recent time when database statistics were reset. If the database is queried for the first time, the system time when the database is connected is returned. The reset time is updated when you call pg_stat_reset on the database and execute pg_stat_reset_single_table_counters against any table or index in it.

  Return type: timestamptz.

- pg_stat_get_function_total_time(oid)

  Description: Specifies the total wall clock time spent in the function, in microseconds. The time spent on this function calling other functions is included.

  Return type: bigint.

- pg_stat_get_xact_tuples_returned(oid)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Specifies the number of rows read through sequential scans when the parameter is a table in the current transaction or the number of index entries returned when the parameter is an index.

Return type: bigint.

- pg_stat_get_xact_numscans(oid)

  Description: Specifies the number of sequential scans performed when the parameter is a table in the current transaction or the number of index scans performed when the parameter is an index.

  Return type: bigint.

- pg_stat_get_xact_blocks_fetched(oid)

  Description: Specifies the number of disk block fetch requests for a table or an index in the current transaction.

  Return type: bigint.

- pg_stat_get_xact_blocks_hit(oid)

  Description: Specifies the number of disk block fetch requests for tables or indexes found in cache in the current transaction.

  Return type: bigint.

- pg_stat_get_xact_function_calls(oid)

  Description: Specifies the number of times the function is called in the current transaction.

  Return type: bigint.

- pg_stat_get_xact_function_self_time(oid)

  Description: Specifies the time spent only on this function in the current transaction. The time spent on other function call inside this function is not included.

  Return type: bigint.

- pg_stat_get_xact_function_total_time(oid)

  Description: Specifies the total wall clock time spent on this function in the current transaction, in microseconds. The time spent on other function call inside this function is included.

  Return type: bigint.

- pg_lock_status()

  Description: Queries information about locks held by open transactions. All users can execute this function.

  Return type: For details, see the return result in **PG_LOCKS**, which is obtained by querying this function.

- bucket_lock_status()

  Description: Queries information about bucket locks held by open transactions. All users can execute this function.

  Return type: For details about the returned columns, see the columns except **node_name** in the view in **GS_BUCKET_LOCKS**.

- gs_bucket_lock_status()

  Description: Queries information about bucket locks held by open transactions on all nodes. All users can execute this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: For details, see the return result in **GS_BUCKET_LOCKS**, which is obtained by querying this function.

- gs_lwlock_status()

  Description: Queries information about all lightweight locks in the database system, including lock waiting and lock holding information. All users can execute this function.

  Return type: SETOF record

- pg_stat_get_wal_senders()

  Description: Queries walsender information on the primary node.

  Return type: SETOF record

  The following table describes return columns.

Table 1-222 Return columns

Column	Type	Description
pid	bigint	Thread ID of the WAL sender.
sender_pid	integer	Lightweight thread ID of the WAL sender.
local_role	text	Type of the primary node.
peer_role	text	Type of the standby node.
peer_state	text	Status of the standby node.
state	text	Status of the WAL sender.
catchup_start	timestamp with time zone	Startup time of a catchup task.
catchup_end	timestamp with time zone	End time of a catchup task.
sender_sent_location	text	Sending position of the primary node.
sender_write_location	text	Writing position of the primary node.
sender_flush_location	text	Flushing position of the primary node.
sender_replay_location	text	Redo position of the primary node.
receiver_received_location	text	Receiving position of the standby node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
receiver_write_location	text	Writing position of the standby node.
receiver_flush_location	text	Flushing position of the standby node.
receiver_replay_location	text	Redo position of the standby node.
sync_percent	text	Synchronization percentage.
sync_state	text	Synchronization status.
sync_group	text	Group to which the synchronous replication belongs.
sync_priority	text	Priority of synchronous replication.
sync_most_available	text	Maximum availability mode.
channel	text	Channel information of the WAL sender.

- get_paxos_replication_info()

  Description: Queries the primary/standby node replication status in Paxos mode.

  Return type: SETOF record

  The following table describes return columns.

  Table 1-223 Return columns

Column	Type	Description
paxos_write_location	text	Location of the Xlog that has been written to the Distribute Consensus Framework (DCF).
paxos_commit_location	text	Location of the Xlog agreed in the DCF.
local_write_location	text	Writing position of a node.
local_flush_location	text	Flushing position of a node.
local_replay_location	text	Redo position of a node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
dcf_replication_info	text	DCF module information of a node. The character string is in JSON format. The fields in the character string are described as follows: **role**: role of the current node. The value can be **LEADER**, **FOLLOWER**, **LOGGER**, **PASSIVE**, or **UNKNOW**. **term**: election term. **run_mode**: DCF running mode. The value **0** indicates that the automatic election mode is enabled, the value **1** indicates the manual election mode, and the value **2** indicates that the automatic election mode is disabled. **work_mode**: DCF working mode. **hb_interval**: heartbeat interval between DCF nodes, in milliseconds. **elc_timeout**: DCF election timeout period, in milliseconds. **applied_index**: position of the log that is applied to the state machine. **commit_index**: position of the log that has been saved by most DCF nodes. Logs before **commit_index** have been made persistent. **first_index**: position of the first log saved on the DCF node. This position is moved backward when the DN calls **dcf_truncate**. The previous logs will be cleared. **last_index**: position of the last log saved by the DCF node. This position refers to the logs that are stored in the memory of the DCF node but are not made persistent. **cluster_min_apply_idx**: minimum location of the applied log in the cluster. **leader_id**: ID of the leader node. **leader_ip**: IP address of the leader node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
		leader_port: port of the leader node, for DCF internal use. **nodes**: information of other nodes in the cluster, including the node ID, IP address, port number, and role.

Example:

```
gaussdb=# select * from get_paxos_replication_info();
 paxos_write_location | paxos_commit_location | local_write_location | local_flush_location |
local_replay_location |

                                                                dcf_replication_info

                                                                
```

Table 1-224 Return values

Parameter	Return Type	Description
local_role	text	Local role.
static_connections	integer	Connection statistics.
db_state	text	Database status.
detail_information	text	Detailed information.

- pg_stat_get_db_numbackends(oid)
  Description: Specifies the number of active server threads for a database.
  Return type: integer
- pg_stat_get_db_xact_commit(oid)
  Description: Specifies the number of transactions committed in a database.
  Return type: bigint.
- pg_stat_get_db_xact_rollback(oid)
  Description: Specifies the number of transactions rolled back in a database.
  Return type: bigint.
- pg_stat_get_db_blocks_fetched(oid)
  Description: Specifies the number of disk blocks fetch requests for a database.
  Return type: bigint.
- pg_stat_get_db_blocks_hit(oid)
  Description: Specifies the number of disk block fetch requests found in cache for a database.
  Return type: bigint.
- pg_stat_get_db_tuples_returned(oid)
  Description: Specifies the number of tuples returned by a database.
  Return type: bigint.
- pg_stat_get_db_tuples_fetched(oid)
  Description: Specifies the number of tuples fetched from a database.
  Return type: bigint.
- pg_stat_get_db_tuples_inserted(oid)
  Description: Specifies the number of tuples inserted in a database.
  Return type: bigint.
- pg_stat_get_db_tuples_updated(oid)
  Description: Specifies the number of tuples updated in a database.
  Return type: bigint.
- pg_stat_get_db_blk_read_time(oid)
  Description: Time spent in reading data file blocks by backends in this database (unit: ms).
  Return type: double.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_stat_get_db_blk_write_time(oid)

  Description: Time spent in writing data file blocks by backends in this database (unit: ms).

  Return type: double.

- pg_stat_get_db_tuples_deleted(oid)

  Description: Specifies the number of tuples deleted in a database.

  Return type: bigint.

- pg_stat_get_db_conflict_lock(oid)

  Description: Specifies the number of lock conflicts in a database.

  Return type: bigint.

- pg_stat_get_db_deadlocks(oid)

  Description: Specifies the number of deadlocks in a database.

  Return type: bigint.

- pg_stat_get_numscans(oid)

  Description: Specifies the number of sequential scans done if parameters are in a table, or the number of index scans done if parameters are in an index.

  Return type: bigint.

- pg_stat_get_role_name(oid)

  Description: Obtains the username based on the user OID. Only users with the SYSADMIN o MONADMIN permission can access the information.

  Return type: text.

  Example:

  ```
  gaussdb=# SELECT pg_stat_get_role_name(10);
   pg_stat_get_role_name
  -----------------------
   aabbcc
  (1 row)
  ```

- pg_stat_get_tuples_returned(oid)

  Description: Specifies the number of active rows read by sequential scans when the argument is a table, or the number of index entries returned when the argument is an index.

  Return type: bigint.

- pg_stat_get_tuples_fetched(oid)

  Description: Specifies the number of rows fetched by bitmap scans if parameters are in a table, or the number of rows fetched by simple index scans in the original table if parameters are in an index.

  Return type: bigint.

- pg_stat_get_tuples_inserted(oid)

  Description: Specifies the number of rows inserted into a table.

  Return type: bigint.

- pg_stat_get_tuples_updated(oid)

  Description: Specifies the number of rows updated in a table.

  Return type: bigint.

- pg_stat_get_tuples_deleted(oid)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Specifies the number of rows deleted from a table.

Return type: bigint.

- pg_stat_get_tuples_changed(oid)

  Description: Specifies the total number of inserted, updated, and deleted rows after a table was last analyzed or autoanalyzed.

  Return type: bigint.

- pg_stat_get_tuples_hot_updated(oid)

  Description: Specifies the number of rows hot updated in a table.

  Return type: bigint.

- pg_stat_get_live_tuples(oid)

  Description: Specifies the number of live rows in a table.

  Return type: bigint.

- pg_stat_get_dead_tuples(oid)

  Description: Specifies the number of dead rows in a table. It indicates the number of inactive row pointers in Ustore tables.

  Return type: bigint.

- pg_stat_get_blocks_fetched(oid)

  Description: Specifies the number of disk block fetch requests for a table or an index.

  Return type: bigint.

- pg_stat_get_blocks_hit(oid)

  Description: Specifies the number of disk block requests found in cache for a table or an index.

  Return type: bigint.

- pg_stat_get_xact_tuples_fetched(oid)

  Description: Specifies the number of tuple rows scanned in a transaction.

  Return type: bigint.

- pg_stat_get_xact_tuples_inserted(oid)

  Description: Specifies the number of tuple inserted into the active subtransactions related to a table.

  Return type: bigint.

- pg_stat_get_xact_tuples_deleted(oid)

  Description: Specifies the number of deleted tuples in the active subtransactions related to a table.

  Return type: bigint.

- pg_stat_get_xact_tuples_hot_updated(oid)

  Description: Specifies the number of hot updated tuples in the active subtransactions related to a table.

  Return type: bigint.

- pg_stat_get_xact_tuples_updated(oid)

  Description: Specifies the number of updated tuples in the active subtransactions related to a table.

  Return type: bigint.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_stat_get_last_vacuum_time(oid)

  Description: Specifies the most recent time when the autovacuum thread is manually started to clear a table.

  Return type: timestamptz.

- pg_stat_get_last_autovacuum_time(oid)

  Description: Specifies the time of the last vacuum initiated by the autovacuum daemon thread on a table.

  Return type: timestamptz.

- pg_stat_get_vacuum_count(oid)

  Description: Specifies the number of times a table is manually cleared.

  Return type: bigint.

- pg_stat_get_autovacuum_count(oid)

  Description: Specifies the number of times the autovacuum daemon thread is started to clear a table.

  Return type: bigint.

- pg_stat_get_last_analyze_time(oid)

  Description: Specifies the last time when a table starts to be analyzed manually or by the autovacuum thread.

  Return type: timestamptz.

- pg_stat_get_last_autoanalyze_time(oid)

  Description: Specifies the time when the last analysis initiated by the autovacuum daemon thread on a table.

  Return type: timestamptz.

- pg_stat_get_analyze_count(oid)

  Description: Specifies the number of times a table is manually analyzed.

  Return type: bigint.

- pg_stat_get_autoanalyze_count(oid)

  Description: Specifies the number of times the autovacuum daemon thread analyzes a table.

  Return type: bigint.

- gs_stat_get_all_tabentries()

  Description: Obtains PgStat statistics of other nodes when the GUC parameter **enable_gsstat_cache** is set to **on**. This is an internal function. You are advised not to use it directly.

  Return type: SETOF record

  For return parameters, see **Table 1-225**.

Table 1-225 Description

Parameter	Return Type	Description
node_name	name	Node name (**node_name** in **pgxc_node**).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Return Type	Description
nspname	name	Schema name.
relname	name	Name of a table or index.
tuples_returned	bigint	If it is a table, the number of rows read by a sequential scan is returned. If it is an index, the number of index rows is returned.
tuples_fetched	bigint	If it is a table, the number of rows fetched with a bitmap scan is returned. If it is an index, the number of rows fetched with a simple index scan is returned.
tuples_inserted	bigint	Number of rows inserted into a table.
tuples_updated	bigint	Number of rows updated in a table.
tuples_deleted	bigint	Number of rows deleted from a table.
tuples_hot_updated	bigint	Number of rows HOT-updated in a table.
n_live_tuples	bigint	Number of live rows in a table.
n_dead_tuples	bigint	Number of dead rows in a table. It indicates the number of inactive row pointers in Ustore tables.
blocks_fetched	bigint	Number of disk block fetch requests for a table or index.
blocks_hit	bigint	Number of disk block requests found in cache for a table or index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Return Type	Description
numscans	bigint	If it is a table, the number of rows read by a sequential scan is returned. If it is an index, the number of index rows is returned.
autovac_vacuum_timestamp	timestamp with time zone	Time of the last vacuum initiated by the autovacuum daemon on this table.
autovac_vacuum_count	bigint	Number of times that the autovacuum daemon starts cleanup on the table.
autovac_analyze_timestamp	timestamp with time zone	Time of the last analyze initiated by the autovacuum daemon on this table.
autovac_analyze_count	bigint	Number of times the autovacuum daemon analyzes a table.
state_change_timestamp	timestamp with time zone	The update time of monitoring indicator fields in each table in the DBE_PERF.stat_all_tables view is returned.
index_stat_change_timestamp	timestamp with time zone	The update time of monitoring indicator fields in each table in the DBE_PERF.stat_all_indexes view is returned.
state_io_change_timestamp	timestamp with time zone	The update time of monitoring indicator fields in each table in the DBE_PERF.statio_all_indexes and DBE_PERF.statio_all_tables views is returned.

- pg_total_autovac_tuples(bool)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Returns tuple records related to the total autovac, such as **nodename**, **nspname**, **relname**, and tuple IUDs. The input parameter specifies whether to query the **relation** information.

Return type: SETOF record

The following table describes return parameters.

Table 1-226 Description

Parameter	Return Type	Description
nodename	name	Node name.
nspname	name	Name of a namespace.
relname	name	Name of an object, such as a table, an index, or a view.
partname	name	Partition name.
n_dead_tuples	bigint	Number of dead rows in a table partition. It indicates the number of inactive row pointers in Ustore tables.
n_live_tuples	bigint	Number of live rows in a table partition.
changes_since_analyze	bigint	Number of changes generated by ANALYZE.

- pg_total_gsi_autovac_tuples(bool)

  Description: Returns tuple records related to **total autovac**, such as **nodename**, **nspname**, **relname**, and number of GSI records that have changed since the last ANALYZE. The input parameter specifies whether to query relation information. This function assists AUTOVACUUM on GSIs. Avoid direct use by users.

  Return type: SETOF record

- pg_autovac_status(oid)

  Description: Returns autovac information, such as **nodename**, **nspname**, **relname**, **analyze**, **vacuum**, thresholds for the ANALYZE and VACUUM operations, and the number of analyzed or vacuumed tuples. Only users with the SYSADMIN permission can use this function.

  Return type: SETOF record

  The following table describes return parameters.

  Table 1-227 Return parameters

Parameter	Return Type	Description
nspname	text	Name of a namespace.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Return Type	Description
relname	text	Name of an object, such as a table, an index, or a view.
nodename	text	Node name.
doanalyze	Boolean	Specifies whether to execute **ANALYZE**.
anltuples	bigint	Number of ANALYZE tuples.
anlthresh	bigint	ANALYZE threshold.
dovacuum	Boolean	Specifies whether to execute **VACUUM**.
vactuples	bigint	Number of VACUUM tuples.
vacthresh	bigint	VACUUM threshold.

- pg_autovac_timeout(oid)

  Description: Returns the number of consecutive timeouts during the autovac operation on a table. If the table information is invalid or the node information is abnormal, **NULL** will be returned.

  Return type: bigint.
- pg_autovac_coordinator(oid)

  Description: Returns the name of the CN performing the autovac operation on a table. If the table information is invalid or the node information is abnormal, **NULL** is returned.

  Return type: text.
- pg_stat_get_last_data_changed_time(oid)

  Description: Returns the time when INSERT, UPDATE, DELETE, or TRUNCATE was last performed on a table, or the time when EXCHANGE, TRUNCATE, or DROP was last performed on a partition or subpartition. The data in the **last_data_changed** column of the **PG_STAT_ALL_TABLES** view is calculated by using this function. The performance of obtaining the last modification time by using the view is poor when the table has a large amount of data. In this case, you are advised to use functions. The input parameter is a table OID.

  Return type: timestamptz.
- pg_stat_set_last_data_changed_time(oid)

  Description: Manually changes the time when **INSERT**, **UPDATE**, **DELETE**, or **EXCHANGE**/**TRUNCATE**/**DROP PARTITION** was last performed.

  Return type: void.
- pg_stat_get_last_updated(oid, text)

  Description: Returns the update time of table monitoring metrics in the DBE_PERF.stat_all_tables, DBE_PERF.stat_all_indexes,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

DBE_PERF.statio_all_indexes, and DBE_PERF.statio_all_tables views. The input parameter 1 is the table OID, and the input parameter 2 is of the text type. The value can be **"stat_table"**, **"stat_index"**, or **"stat_io"**.

Return type: timestamptz.

- pg_backend_pid()

  Description: Specifies the thread ID of the server thread attached to the current session.

  Return type: integer

- pg_stat_get_activity(integer)

  Description: Returns a record about the backend thread with the specified PID. A record for each active backend in the system is returned if **NULL** is specified. The returned result does not contain the **connection_info** column. The initial user and users with the SYSADMIN or MONADMIN permission can view all data. Common users can only query their own results.

  Example:

```
gaussdb=# SELECT pg_backend_pid();
 pg_backend_pid
-----------------
 139881386280704
(1 row)

-- The query result depends on the preceding return value.
gaussdb=# SELECT * FROM pg_stat_get_activity(139881386280704);
 datid |       pid       | sessionid | usesysid | application_name | state  |
query                       | waiting |          xact_start           |          query_start          |
    backend_start         |         state_change          | client_addr | client_hostname | client_port | enqueue
|     query_id      |   srespool   | global_sessionid | unique_sql_id | trace_id
-------+-----------------+-----------+----------+------------------+--------
+-----------------------------------------------------+---------+-------------------------------
+--------------------------------+-----
-------------------------+------------------------------+-------------+-----------------+-------------+---------
+-------------------+--------------+------------------+---------------+----------
 16545 | 139881386280704 |        69 |       10 | gsql             | active | select * from
pg_stat_get_activity(139881386280704); | f       | 2022-01-18 19:43:05.167718+08 | 2022-01-18
19:43:05.167718+08 | 2022
-01-18 19:42:33.513507+08 | 2022-01-18 19:43:05.16773+08 |                  |                  |          -1 |         |
72620543991624410 | default_pool | 1938253334#69#0  |    3751941862 |
(1 row)
```

  Return type: SETOF record

  The following table describes return parameters.

Table 1-228 Description

Parameter	Return Type	Description
datid	oid	OID of the database that the user session connects to in the backend.
pid	bigint	Backend thread ID.
sessionid	bigint	Session ID.
usesysid	oid	OID of the user logged in to the backend.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Return Type	Description
application_name	text	Name of the application connected to the backend.
state	text	Overall status of the backend.
query	text	Latest query at the backend. If the value of **state** is **active**, this column shows the ongoing query. In all other states, it shows the last query that was executed.
waiting	Boolean	Specifies whether the backend is currently waiting for a lock. If the backend is currently waiting for a lock, the value is **true**.
xact_start	timestamp with time zone	Time when current transaction was started (null if no transaction is active). If the current query is the first of its transaction, the value of this column is the same as that of the **query_start** column.
query_start	timestamp with time zone	Time when the currently active query was started, or time when the last query was started if **state** is not **active** For a stored procedure, function, or package, the first query time is queried and does not change with the running of statements in the stored procedure.
backend_start	timestamp with time zone	Time when this process was started, that is, when the client connected to the server.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Return Type	Description
state_change	timestamp with time zone	Time when **state** was last modified.
client_addr	inet	IP address of the client connected to the backend. If this column is **NULL**, it indicates either the client is connected via a Unix socket on the server or this is an internal thread, such as **AUTOVACUUM**.
client_hostname	text	Host name of the connected client, as reported by a reverse DNS lookup of **client_addr**. This column will be non-null only for IP connections and only when **log_hostname** is enabled.
client_port	integer	TCP port number that the client uses for communication with this backend (**–1** if a Unix socket is used).
enqueue	text	Unsupported currently.
query_id	bigint	ID of a query.
srespool	name	Name of the resource pool.
global_sessionid	text	Global session ID.
unique_sql_id	bigint	Unique SQL statement ID.
trace_id	text	Driver-specific trace ID, which is associated with an application request.

- pg_stat_get_activity_with_conninfo(integer)

  Description: Returns a record about the backend thread with the specified PID. A record for each active backend in the system is returned if **NULL** is specified. The initial user, system administrators and users with the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MONADMIN permission can view all data. Common users can only query their own results.

Return type: SETOF record

The following table describes return values.

Return Value	Return Type	Return values
datid	oid	OID of the database that the user session connects to in the backend.
pid	bigint	Backend thread ID.
sessionid	bigint	Session ID.
usesysid	oid	OID of the user logged in to the backend.
application_name	text	Name of the application connected to the backend.
state	text	Overall status of the backend.
query	text	Latest query at the backend. If the value of **state** is **active**, this column shows the ongoing query. In all other states, it shows the last query that was executed.
waiting	Boolean	Specifies whether the backend is currently waiting for a lock. If the backend is currently waiting for a lock, the value is **true**.
xact_start	timestamp with time zone	Time when current transaction was started (null if no transaction is active). If the current query is the first of its transaction, the value of this column is the same as that of the **query_start** column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return Value	Return Type	Return values
query_start	timestamp with time zone	Time when the currently active query was started, or time when the last query was started if **state** is not **active** For a stored procedure, function, or package, the first query time is queried and does not change with the running of statements in the stored procedure.
backend_start	timestamp with time zone	Time when this process was started, that is, when the client connected to the server.
state_change	timestamp with time zone	Time when **state** was last modified.
client_addr	inet	IP address of the client connected to the backend. If this column is **NULL**, it indicates either the client is connected via a Unix socket on the server or this is an internal thread, such as **AUTOVACUUM**.
client_hostname	text	Host name of the connected client, as reported by a reverse DNS lookup of **client_addr**. This column will be non-null only for IP connections and only when **log_hostname** is enabled.
client_port	integer	TCP port number that the client uses for communication with this backend (**–1** if a Unix socket is used).
enqueue	text	Unsupported currently.
query_id	bigint	ID of a query.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return Value	Return Type	Return values
connection_info	text	A string in JSON format recording the driver type, driver version, driver deployment path, and thread owner of the connected database.
srespool	name	Name of the resource pool.
global_sessionid	text	Global session ID.
unique_sql_id	bigint	Unique SQL statement ID.
trace_id	text	Driver-specific trace ID, which is associated with an application request.
top_xid	xid	Top-level transaction ID of a transaction.
current_xid	xid	Current transaction ID of a transaction.
xlog_quantity	bigint	Size of Xlogs currently used by a transaction, in bytes.
control_status	text	If CPU resources are controlled due to abnormal SQL statements, the value will be **under_cpu_throttle**. Otherwise, it will be **normal**.

- gs_get_explain(integer)

  Description: Returns a running plan for the backend thread with the specified PID. The PID cannot be empty. This function takes effect only when the GUC parameter **track_activities** is set to **on**. Only explainable SQL statements whose plans do not contain stream operators are supported. Details are as follows:

  – If the GUC parameter **plan_collect_thresh** is set to **-1**, the return result of the function is always empty.

  – If the GUC parameter **plan_collect_thresh** is set to **0**, the current SQL execution time is greater than or equal to the value of **log_min_duration_statement**, and the total number of tuples processed by all operators in the plan is greater than or equal to 10000, the system

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

starts to collect plans in running state. Each time the total number of tuples processed by all operators exceeds 10000, a collection is performed.

- If the GUC parameter **plan_collect_thresh** is set to a value greater than 0, running plans are collected incrementally based on the threshold specified by this parameter.
- In addition, this function can view only the plans generated on CNs. To view the plans generated on DNs, see the gs_get_dn_explain function.

The return value type is text. The following table describes the types and meanings of the fields.

Return Value	Return Type	Return values
Character string of the plans in the running state	text	**A-rows** in the plan string indicates the number of rows returned by the operator in real time.

- gs_get_dn_explain(text, bigint)

  Description: Returns a running plan for the backend thread of the DN with the specified node name (from the **node_name** field in the **pgxc_node** system catalog) and global session ID. The global session ID cannot be empty. This function takes effect only when the GUC parameter **track_activities** is set to **on**. Only explainable SQL statements whose plans do not contain stream operators are supported. Details are as follows:

  - Same as the gs_get_explain function. The global session ID is calculated based on the **global_sessionid** field in **pg_stat_activity**. The format of **global_sessionid** is *${number1}.${number2}#{number3}*. Note that *number1* is a signed number. When *number1* is a positive number, for example, **1938253334:1#0**, the parameter value of the gs_get_dn_explain function should be *${number2}* x 10000000000 + *${number1}*. When *number1* is a negative number, convert it to a 32-bit unsigned number. For example, if the value of **global_sessionid** is **-1736975100:1#0**, the parameter value of the gs_get_dn_explain function should be *${number2}* x 10000000000 + (*${number1}* + 4294967296).
  - In addition, this function can be executed only with the ADMIN or MONADMIN permission.

  The return value type is text. The following table describes the types and meanings of the fields.

Return Value	Return Type	Return values
Character string of the plans in the running state	text	**A-rows** in the plan string indicates the number of rows returned by the operator in real time.

- pg_stat_get_activity_ng(integer)

  Description: Returns a record about the active backend thread with the specified PID. A record for each active backend thread is returned if **NULL** is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

specified. System administrators and users with the MONADMIN permission can view all data. Common users can query only their own data.

Return type: SETOF record

The following table describes return fields.

Name	Type	Description
datid	oid	Database OID.
pid	bigint	Backend thread ID.
sessionid	bigint	Session ID.
node_group	text	Node group of the user to which the data belongs.

- pg_stat_get_function_calls(oid)

  Description: Specifies the number of times the function has been called.

  Return type: bigint.

- pg_stat_get_function_self_time(oid)

  Description: Specifies the time spent on only this function. The time spent on nested functions to call other functions is excluded.

  Return type: bigint.

- pg_stat_get_backend_idset()

  Description: Sets the number of currently active server threads (from 1 to the number of active server threads).

  Return type: setofinteger

- pg_stat_get_backend_pid(integer)

  Description: Specifies the ID of the given server thread.

  Return type: bigint.

- pg_stat_get_backend_dbid(integer)

  Description: Specifies the ID of the database connected to the given server thread.

  Return type: oid.

- pg_stat_get_backend_userid(integer)

  Description: Specifies the user ID of the given server process. This function can be called only by system administrators.

  Return type: oid.

- pg_stat_get_backend_activity(integer)

  Description: Active command of the given server thread, but only if the current user is a system administrator or the same user as that of the session being queried and **track_activities** is enabled.

  Return type: text.

- pg_stat_get_backend_waiting(integer)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Returns true if the given server thread is waiting for a lock, but only if the current user is a system administrator or the same user as that of the session being queried and **track_activities** is enabled.

Return type: Boolean

- pg_stat_get_backend_activity_start(integer)

  Description: Specifies the time when the given server thread's currently executing query is started only if the current user is a system administrator or the user of the session being queried and **track_activities** is enabled.

  Return type: timestamp with time zone

- pg_stat_get_backend_xact_start(integer)

  Description: Specifies the time when the given server thread's currently executing transaction is started only if the current user is a system administrator or the user of the session being queried and **track_activities** is enabled.

  Return type: timestamp with time zone

- pg_stat_get_backend_start(integer)

  Description: Specifies the time when the given server thread is started. If the current user is neither a system administrator nor the same user as that of the session being queried, **NULL** is returned.

  Return type: timestamp with time zone

- pg_stat_get_backend_client_addr(integer)

  Description: Specifies the IP address of a backend connected to a specified client. If the connection is over a UDS, or if the current user is neither a system administrator nor the same user as that of the session being queried, **NULL** will be returned.

  Return type: inet.

- pg_stat_get_backend_client_port(integer)

  Description: Specifies the TCP port of a backend connected to a specified client. If the connection is over a UDS, **–1** will be returned. If the current user is neither a system administrator nor the same user as that of the session being queried, **NULL** will be returned.

  Return type: integer

- pg_stat_get_bgwriter_timed_checkpoints()

  Description: Specifies the time when the background writer thread starts scheduled checkpoints (because **checkpoint_timeout** has expired).

  Return type: bigint.

- pg_stat_get_bgwriter_requested_checkpoints()

  Description: Specifies the time when the background writer thread starts checkpoints based on requests from the backend because **checkpoint_segments** has been exceeded or the **CHECKPOINT** command has been executed.

  Return type: bigint.

- pg_stat_get_bgwriter_buf_written_checkpoints()

  Description: Specifies the number of buffers written by the background writer thread during checkpoints.

  Return type: bigint.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_stat_get_bgwriter_buf_written_clean()

  Description: Specifies the number of buffers written by the background writer thread for routine cleaning of dirty pages.

  Return type: bigint.

- pg_stat_get_bgwriter_maxwritten_clean()

  Description: Specifies the time when the background writer thread stops its cleaning scan because it has written more buffers than specified in the **bgwriter_lru_maxpages** parameter.

  Return type: bigint.

- pg_stat_get_buf_written_backend()

  Description: Specifies the number of buffers written by the backend thread because they need to allocate a new buffer.

  Return type: bigint.

- pg_stat_get_buf_alloc()

  Description: Specifies the total number of the allocated buffers.

  Return type: bigint.

- pg_stat_clear_snapshot()

  Description: Discards the current statistics snapshot. Only users with the SYSADMIN or MONADMIN permission can execute this function.

  Return type: void.

- pg_stat_reset()

  Description: Resets all statistics counters for the current database to zero (the SYSADMIN permission is required).

  Return type: void.

- gs_stat_reset()

  Description: Resets all statistics counters for the current database on each node to zero (the SYSADMIN permission is required).

  Return type: void.

- pg_stat_reset_shared(text)

  Description: Resets all statistics counters for the current database in each node in a shared cluster to zero (the SYSADMIN permission is required).

  Return type: void.

- pg_stat_reset_single_table_counters(oid)

  Description: Resets statistics for a single table or index in the current database to zero (the SYSADMIN permission is required).

  Return type: void.

- pg_stat_reset_single_function_counters(oid)

  Description: Resets statistics for a single function in the current database to zero (the SYSADMIN permission is required).

  Return type: void.

- pgxc_fenced_udf_process(integer)

  Description: Displays the number of UDF master and worker threads. Only users with the SYSADMIN or MONADMIN permission can execute this

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

function. If the input parameter is set to **1**, the number of Master threads is queried. If the input parameter is set to **2**, the number of Worker threads is queried. If the input parameter is set to **3**, all Worker threads are terminated.

Return type: text.

- fenced_udf_process(integer)

  Description: Shows the number of local UDF Master and Work processes. Only users with the SYSADMIN or MONADMIN permission can terminate all worker processes. If the input parameter is set to **1**, the number of master processes is queried. If the input parameter is set to **2**, the number of worker processes is queried. If the input parameter is set to **3**, all worker processes are terminated.

  Return type: text.

- total_cpu()

  Description: Obtains the CPU time used by the current node, in jiffies.

  Return type: bigint.

- total_memory()

  Description: Obtains the size of the virtual memory used by the current node, in KB.

  Return type: bigint.

- pgxc_terminate_all_fenced_udf_process()

  Description: Kills all UDF worker threads. Only users with the SYSADMIN or MONADMIN permission can execute this function.

  Return type: Boolean.

- gs_get_nodegroup_tablecount(name)

  Description: Obtains the total number of user tables in all databases in a node group.

  Return type: integer

- pgxc_max_datanode_size(name)

  Description: Obtains the maximum disk space occupied by database files on all DNs in a node_group. The unit is byte.

  Return type: bigint.

- gs_check_tables_distribution()

  Description: Checks whether the user table distribution in the system is consistent. If no record is returned, table distribution is consistent. This function cannot be called during redistribution in scale-in or scale-out.

  Return type: record.

- pg_stat_bad_block()

  Description: Obtains damage information about pages after the current node is started.

  Return type: record.

- pgxc_stat_bad_block()

  Description: Obtains damage information about pages after all nodes in the cluster are started.

  Return type: record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_stat_bad_block_clear()

  Description: Deletes the page damage information that is read and recorded on the node (the SYSADMIN permission is required).

  Return type: void.

- pgxc_stat_bad_block_clear

  Description: Deletes the page damage information that is read and recorded on all nodes in the cluster (the SYSADMIN permission is required).

  Return type: void.

- pgxc_log_comm_status(void)

  Description: When the SCTP communication mode is used, the PGXC system view exports the communication layer status of DNs to each log file. (Due to specification changes, the current version no longer supports this feature. Do not use it.)

  Return type: void.

- gs_respool_exception_info(pool text)

  Description: Queries the query rule of a specified resource pool.

  Return type: record.

- gs_control_group_info(pool text)

  Description: Queries information about Cgroups associated with a resource pool. Only users with the SYSADMIN permission can execute this function.

  Return type: record.

  The command output is as follows:

Attribute	Value	Description
name	class_a:workload_a1	Class name and workload name.
class	class_a	Class Cgroup name.
workload	workload_a1	Workload Cgroup name.
type	DEFWD	Cgroup type (**Top**, **CLASS**, **BAKWD**, **DEFWD**, or **TSWD**).
gid	87	Cgroup ID.
shares	30	Percentage of CPU resources to those on the parent node.
limits	0	Percentage of CPU cores to those on the parent node.
rate	0	Allocation ratio in Timeshare.
cpucores	0-3	Number of CPU cores.

- gs_all_control_group_info()

  Description: Collects information about all Cgroups in the database. For details about the columns returned by the function, see **GS_ALL_CONTROL_GROUP_INFO**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record.

- gs_get_control_group_info()

  Description: Collects information about all Cgroups. For details about the columns returned by the function, see **GS_GET_CONTROL_GROUP_INFO**. Only users with the SYSADMIN permission can execute this function.

  Return type: record.

- gs_get_session_sql_memory()

  Description: Views the memory usage of currently running statements. For details about the columns returned by the function, see **DBE_PERF.SESSION_SQL_MEMORY**.

  Return type: record.

- gs_get_global_session_sql_memory()

  Description: Views the memory usage of currently running statements on all nodes. For details about the columns returned by the function, see **DBE_PERF.GLOBAL_SESSION_SQL_MEMORY**.

  Return type: record.

- gs_blackbox_dump()

  Description: Exports black box files from the system to the *$GAUSSLOG*/**gs_blackbox/***{nodename}* directory. If the command is successfully executed, the black box file names generated in the *$GAUSSLOG*/**gs_blackbox/***{nodename}* directory is returned.

  Return type: text.

- gs_blackbox_show(blackBoxFile cstring)

  Description: Parses black box files and displays them in a table.

  Parameters:

Parameter	Description	Value Range
blackBoxFile	The black box files to be parsed. If no parameter is specified, the black box files collected by the current process are displayed.	The value is a string or is not specified. If a valid black box file name is specified, the file with the specified name in the *$GAUSSLOG*/**gs_blackbox/***nodename* directory is read. Only the file name is supported. An error is reported if the file with the specified name does not exist in the *$GAUSSLOG*/**gs_blackbox/***nodename* directory.

  Return type: SETOF record

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The following table describes return values.

Return Value	Return Type	Return values
type	text	Black box field type.
tid	bigint	Thread ID.
sample_time	timestamp	Collection time.
value	text	Content recorded in the black box.

The following table describes query results.

type	tid	sample_time	value
ERRNUMBER	123456	2024-03-19 09:33:17.386	19460
BLACK_BOX_PLSQL_EXCEPTION	45678	2024-03-19 09:33:18.396	message: "Dropped rowtype entry for non-dropped column when make tuple." funcname: basic_make_tuple_from_row filename:pl_exec.cpp context:xxxx

- gs_blackbox_list()

  Description: Lists black box files in the black box directory of the current instance. The default directory is *$GAUSSLOG*/**gs_blackbox/***{nodename}*.

  Return type: SETOF record

  The following table describes return values.

Return Value	Return Type	Return values
file	text	Name of the black box file.

- get_instr_workload_info(integer)

  Description: Obtains the transaction volume and time information on the current CN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record.

Attribute	Type	Description
user_oid	oid	User ID.
commit_counter	bigint	Number of frontend transactions that were committed.
rollback_counter	bigint	Number of frontend transactions that were rolled back.
resp_min	bigint	Minimum response time of frontend transactions (unit: μs).
resp_max	bigint	Maximum response time of frontend transactions (unit: μs).
resp_avg	bigint	Average response time of frontend transactions (unit: μs).
resp_total	bigint	Total response time of frontend transactions (unit: μs).
bg_commit_counter	bigint	Number of backend transactions that were committed.
bg_rollback_counter	bigint	Number of backend transactions that were rolled back.
bg_resp_min	bigint	Minimum response time of backend transactions (unit: μs).
bg_resp_max	bigint	Maximum response time of backend transactions (unit: μs).
bg_resp_avg	bigint	Average response time of backend transactions (unit: μs).
bg_resp_total	bigint	Total response time of backend transactions (unit: μs).
dist_commit_counter	bigint	Number of distributed transactions that are committed.
dist_rollback_counter	bigint	Number of distributed transactions that are rolled back.
dist_resp_min	bigint	Minimum response time of distributed transactions (unit: μs).
dist_resp_max	bigint	Maximum response time of distributed transactions (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Attribute	Type	Description
dist_resp_avg	bigint	Average response time of distributed transactions (unit: μs).
dist_resp_total	bigint	Total response time of distributed transactions (unit: μs).
dbid	oid	Database ID.

NOTE

Only the information about the main transaction is collected. The information about the sub-transaction is not collected.

- pv_instance_time()

  Description: Obtains the time consumed in each execution phase on the current node.

  Return type: record.

Stat_name Attribute	Value	Description
DB_TIME	1062385	Total end-to-end time consumed by all threads (unit: μs).
CPU_TIME	311777	Total CPU time consumed by all threads (unit: μs).
EXECUTION_TIME	380037	Total time spent on the executor (unit: μs).
PARSE_TIME	6033	Total time spent in parsing SQL statements (unit: μs).
PLAN_TIME	173356	Total time spent in generating an execution plan (unit: μs).
REWRITE_TIME	2274	Total time spent on query rewriting (unit: μs).
PL_EXECUTION_TIME	0	Total time spent in executing PL/pgSQL statements (unit: μs).
PL_COMPILATION_TIME	557	Total time spent in compiling SQL statements (unit: μs).
NET_SEND_TIME	1673	Total time spent in sending data over the network (unit: μs).
DATA_IO_TIME	426622	Total time spent on data reads/writes (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_PERF.get_global_instance_time()

  Description: Provides the time consumed in each key phase in the entire cluster. The time consumed can be queried only on the CN. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- get_instr_unique_sql()

  Description: Obtains information about normalized SQL statements executed on the current node. Users with the SYSADMIN or MONADMIN permission can query all information. Other users can query only information about the statements executed by themselves. For details about the columns returned by the function, see **DBE_PERF.STATEMENT**.

  Return type: record.

- get_instr_wait_event(integer)

  Description: Obtains the statistics on wait events of the current node.

  Return type: record.

- get_instr_user_login()

  Description: Obtains the number of user login and logout times on the current node. Only users with the SYSADMIN or MONADMIN permission can execute this function.

  Return type: record.

- get_instr_rt_percentile(integer)

  Description: Obtains the distribution information about the response time of 80% and 95% SQL statements in the CCN. The unified cluster information is stored on the CCN. The query result from other nodes is **0**.

  Return type: record.

- get_node_stat_reset_time()

  Description: Obtains statistics about reset (restart, primary/standby switchover, and database deletion) time of the current node.

  Return type: record.

- gs_paxos_stat_replication()

  Description: Queries the standby node information on the primary node.

  Return type: SETOF record

  The following table describes return columns.

Column	Type	Description
local_role	text	Role of the sender node.
peer_role	text	Role of the receiver node.
local_dcf_role	text	DCF role of the sender node.
peer_dcf_role	text	DCF role of the receiver node.
peer_state	text	Status of the receiver node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
sender_write_location	text	Location in the Xlog buffer where the node that sends logs is written.
sender_commit_location	text	Barrier reached for the DCF logs of the sender node.
sender_flush_location	text	Location in the Xlog disk where the node that sends logs is written.
sender_replay_location	text	Location where the sender node replays logs.
receiver_write_location	text	Location in the Xlog buffer where the node that receives logs is written.
receiver_commit_location	text	Barrier reached for the DCF logs of the receiver node.
receiver_flush_location	text	Location in the Xlog disk where the node that receives logs is written.
receiver_replay_location	text	Location where the receiver node replays Xlogs.
sync_percent	text	Synchronization percentage.
dcf_run_mode	int4	DCF synchronization mode.
channel	text	Channel information.

- get_paxos_replication_info()

  Description: Queries the primary/standby node replication status in Paxos mode.

  Return type: SETOF record

  The following table describes return columns.

Column	Type	Description
paxos_write_location	text	Location of the Xlog that has been written to the Distribute Consensus Framework (DCF).
paxos_commit_location	text	Location of the Xlog agreed in the DCF.
local_write_location	text	Writing position of a node.
local_flush_location	text	Flushing position of a node.
local_replay_location	text	Redo position of a node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
dcf_replication_info	text	DCF module information of a node.

- gs_wlm_get_user_info(int)

  Description: Obtains information about all users. The input parameter is of the int type and can be any int value or **NULL**. Only users with the SYSADMIN permission can execute this function.

  Return type: record.

- gs_wlm_readjust_user_space(oid)

  Description: Corrects the storage space usage of a specified user. Common users can modify only their own usage. Administrators can modify the usage of all users. If **oid** is set to **0**, the usage of all users needs to be modified.

  Return type: record.

- gs_wlm_readjust_user_space_through_username(text name)

  Description: Corrects the storage space usage of a specified user. Common users can use this function to modify only their own usage. Only the administrator can modify the usage of all users. If the value of **name** is **0000**, the usage of all users needs to be modified.

  Return type: record.

- gs_wlm_readjust_user_space_with_reset_flag(text name, boolean isfirst)

  Description: Corrects the storage space usage of a specified user. If the input parameter **isfirst** is set to **true**, statistics are collected from 0. Otherwise, statistics are collected from the previous result. Common users can use this function to modify only their own usage. Only the administrator can modify the usage of all users. If the value of **name** is **0000**, the usage of all users needs to be modified.

  Return type: record.

- gs_io_wait_status()

  Description: Returns the real-time statistics on I/O control on the current node.

  Return type: SETOF record

Name	Type	Description
node_name	text	Node name.
device_name	text	Name of the data disk mounted to the node.
read_per_second	float	Number of read completions per second.
write_per_second	float	Number of write completions per second.
write_ratio	float	Ratio of the disk write I/Os to the total I/Os.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
io_util	float	Percentage of the I/O time to the total CPU time per second.
total_io_util	integer	Level of the CPU time occupied by the last three I/Os. The value ranges from 0 to 6.
tick_count	integer	Interval for updating disk I/O information. The value is fixed to 1 second. The value is cleared each time before data is read.
io_wait_list_len	integer	Size of the I/O request thread wait queue. If the value is **0**, no I/O is under control.

- gs_get_shared_memctx_detail(text)

  Description: Returns the memory allocation details of the specified memory context, including the file, line number, and size of each memory allocation (the size of the same line in the same file is accumulated). The memory context is limited to the one queried through the PG_SHARED_MEMORY_DETAIL view (see **PG_SHARED_MEMORY_DETAIL**). The input parameter is the memory context name (that is, the **contextname** column in the result returned by the PG_SHARED_MEMORY_DETAIL view). To execute this function, you must have the SYSADMIN or MONADMIN permission.

  Return type: SETOF record

Name	Type	Description
file	text	Name of the file where the memory is applied for.
line	int8	Line number of the code in the file where the requested memory is located.
size	int8	Size of the allocated memory. The value is accumulated when memory is allocated multiple times to the same line in the same file.

- gs_get_session_memctx_detail(text)

  Description: Returns the memory allocation details of the specified memory context, including the file, line number, and size of each memory allocation (the size of the same line in the same file is accumulated). This parameter is valid only in thread pool mode. Only the memory context queried through the PV_SESSION_MEMORY_CONTEXT view (see **PV_SESSION_MEMORY_CONTEXT**) can be queried. The input parameter is the memory context name (that is, the **contextname** column in the result returned by the PV_SESSION_MEMORY_CONTEXT view). To execute this function, you must have the SYSADMIN or MONADMIN permission.

  Return type: SETOF record

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
file	text	Name of the file where the memory is applied for.
line	int8	Line number of the code in the file where the requested memory is located.
size	int8	Size of the allocated memory, in bytes. The value is accumulated if the memory is allocated for multiple times to the same line of the same file.

NOTE

This function takes effect only in thread pool mode.

- gs_get_thread_memctx_detail(tid,text)

  Description: Returns the memory allocation details of the specified memory context, including the file, line number, and size of each memory allocation (the size of the same line in the same file is accumulated). Only the memory context queried through the PV_THREAD_MEMORY_CONTEXT view (see **PV_THREAD_MEMORY_CONTEXT**) can be queried. The first input parameter is the thread ID (the **tid** column of the data returned by the PV_THREAD_MEMORY_CONTEXT view), and the second parameter is the memory context name (the **contextname** column of the data returned by the PV_THREAD_MEMORY_CONTEXT view). To execute this function, you must have the SYSADMIN or MONADMIN permission.

  Return type: SETOF record

Name	Type	Description
file	text	Name of the file where the memory is applied for.
line	int8	Line number of the code in the file where the requested memory is located.
size	int8	Size of the allocated memory, in bytes. The value is accumulated if the memory is allocated for multiple times to the same line of the same file.

- gs_get_history_memory_detail(cstring)

  Description: Queries historical memory snapshot information. The input parameter type is cstring. The value can be **NULL** or the name of the memory snapshot log file.

  a. If the value of the input parameter is **NULL**, the list of all memory snapshot log files on the current node is displayed.

  b. If the value of the input parameter is the name of the memory snapshot log file in the list queried in **a**, the detailed information about the memory snapshot recorded in the log file is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

c. If you enter any other input parameter, the system displays a message indicating that the input parameter is incorrect or the file fails to be opened.

To execute this function, you must have the SYSADMIN or MONADMIN permission.

Return type: text.

Name	Type	Description
memory_info	text	Memory information. If the input parameter of the function is set to **NULL**, the memory snapshot file list is displayed. If the input parameter is set to the name of the memory snapshot file, the content of the file is displayed.

- gs_stat_get_hotkeys_info()

  NOTE

  If the GUC parameter **enable_hotkeys_collection** is set to **off**, the gs_stat_get_hotkeys_info and global_stat_get_hotkeys_info functions as well as the global_stat_hotkeys_info view cannot be queried. The use of the gs_stat_clean_hotkeys and global_stat_clean_hotkeys APIs is not affected.

  Description: Obtains the hotspot key statistics on the current node.

  Return type: record.

```
gaussdb=# SELECT * FROM gs_stat_get_hotkeys_info() order by count, hash_value;
 database_name | schema_name |    table_name     | key_value | hash_value | count
---------------+-------------+-------------------+-----------+------------+-------
 regression    | public      | hotkey_single_col | {22}      | 1858004829 |     2
 regression    | public      | hotkey_single_col | {11}      | 2011968649 |     2
(2 rows)
```

Name	Type	Description
database_name	text	Name of the database where the hotspot key is located.
schema_name	text	Name of the schema where the hotspot key is located.
table_name	text	Name of the table where the hotspot key is located.
key_value	text	Value of the hotspot key.
hash_value	bigint	Hash value of the hotspot key in the database. If the table is a list or range distributed table, the value of this column is **0**.
count	bigint	Frequency of accessing the hotspot key.

- gs_stat_clean_hotkeys()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- Hot key detection is designed for high-concurrency and heavy-traffic scenarios. In the scenario where the access is performed for several times, the query result may be inaccurate.
- The clearing API is designed to clear only the statistics in the LRU queue but not the historical data in the FIFO. Therefore, if the historical key value in the FIFO is accessed again after the clearing, the historical key value is still processed as a hotspot key. This rule also applies to **global_stat_clean_hotkeys**.

Description: Clears statistics on hotspot keys on the current node.

Return type: Boolean.

```
gaussdb=# SELECT * FROM gs_stat_clean_hotkeys();
 gs_stat_clean_hotkeys
-----------------------
 t
(1 row)
```

- gs_stat_progress_vacuum(pid)

  Description: Displays the current VACUUM progress. Only the VACUUM progress of Astore and Ustore page tables is collected. Hash bucket and segment-page tables are not supported.

  Return type: record.

Table 1-229 Return parameters

Name	Type	Description
pid	bigint	PID of the VACUUM thread.
dataname	text	Name of the database that is being vacuumed.
relid	oid	OID of the table that is being vacuumed.
relname	text	Name of the table that is being vacuumed.
phase	text	Current VACUUM phase.
detail	text	Vacuum progress details, including the total number of pages, number of scanned pages, number of vacuumed pages, and number of vacuumed indexes.

Table 1-230 Description of phase values

phase	Type	Description
invalid	text	VACUUM is starting and a table lock will be obtained.
initialize	text	VACUUM is preparing to scan the heap table. This phase is expected to be very short.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

phase	Type	Description
vacuuming main part	text	VACUUM is vacuuming the primary partitioned table (usually the GPI of the primary partitioned table). You can view the vacuum progress in the **detail** column.
scanning heap	text	VACUUM is scanning the heap table and defragmenting or freezing each page if necessary. You can view the vacuum progress in the **detail** column.
vacuuming indexes	text	VACUUM is vacuuming indexes. If the table has indexes, this phase occurs at least once and may continue scanning the heap table only after the indexes are vacuumed multiple times as required. You can view the vacuum progress in the **detail** column.
vacuuming heap	text	VACUUM is vacuuming the heap table (different from scanning the heap table) and is executed after each index vacuum. If the heap table is not completely scanned, the scanning will continue after this phase. You can view the vacuum progress in the **detail** column.
vacuuming part cache	text	VACUUM is vacuuming dead tuples in cached partitions. You can view the vacuum progress in the **detail** column.
cleaning up indexes	text	VACUUM is vacuuming indexes. This phase is executed after the heap table is completely scanned and all indexes and heap tables are vacuumed.
truncating heap	text	VACUUM is truncating the tail empty pages of the heap table and returning the empty pages to the OS.
vacuuming urq	text	VACUUM is recycling the index pages of the Ustore table.
unknown phase	text	The current state is unknown and is not defined in the preceding phases. Generally, this state does not occur.

```
-- Set GUC parameters.
gaussdb=# SET vacuum_cost_delay=100;
SET
gaussdb=# CREATE TABLE t1 (a int, b int, c text) with (storage_type=astore)
gaussdb-# partition by range(a) (
gaussdb(# partition p1 values less than(10000),
gaussdb(# partition p2 values less than(20000),
gaussdb(# partition p3 values less than(30000),
gaussdb(# partition p4 values less than(40000),
gaussdb(# partition p5 values less than(50000),
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb(# partition p6 values less than(60000),
gaussdb(# partition p7 values less than(70000),
gaussdb(# partition p8 values less than(80000),
gaussdb(# partition p9 values less than(90000),
gaussdb(# partition p10 values less than(maxvalue));
CREATE TABLE
gaussdb=# CREATE INDEX idx11 on t1(a) global;
CREATE INDEX
gaussdb=#
gaussdb=# CREATE INDEX idx12 on t1(b) global;
CREATE INDEX
gaussdb=#
gaussdb=# CREATE INDEX idx13 on t1(c) global;
CREATE INDEX
gaussdb=# INSERT INTO t1 VALUES (generate_Series(1,100000), generate_series(1,3), 'abcde');
INSERT 0 300000
gaussdb=# DELETE t1;
DELETE 300000
gaussdb=# VACUUM t1;
VACUUM
-- Create a session to connect to the current node.
gaussdb=#  SELECT pid,query FROM pg_stat_activity where query = 'vacuum t1;';
       pid       |   query
-----------------+------------
 140242746078976 | vacuum t1;
(1 row)
gaussdb=# SELECT * FROM gs_stat_progress_vacuum(140242746078976);
       pid       | dataname | relid | relname |     phase
|                                                
```

- global_comm_get_recv_stream()

  Description: Obtains the status of the stream received by all communications libraries on all DNs. For details about the columns returned by the function, see **PG_COMM_RECV_STREAM**.

  Return type: record.

- global_comm_get_send_stream()

  Description: Obtains the status of the stream sent by all communications libraries on all DNs. For details about the columns returned by the function, see **PG_COMM_SEND_STREAM**.

  Return type: record.

- global_comm_get_status()

  Description: Obtains the communications library status on all DNs. For details about the columns returned by the function, see **PG_COMM_STATUS**.

  Return type: record.

- global_comm_client_info()

  Description: Obtains information about active client connections of global nodes. For details about the columns returned by the function, see **COMM_CLIENT_INFO**.

  Return type: record.

- global_comm_get_client_info()

  Description: Obtains information about client connections of global nodes. For details about the columns returned by the function, see **COMM_CLIENT_INFO**.

  Return type: record.

- pgxc_stat_activity()

  Description: Displays information about all CNs in the current cluster queried by the current user. Only users with the SYSADMIN or MONADMIN permission can execute this function, and common users can view only their own information. Three columns are added to version 505.0.0. During the gray upgrade observation period, an error is reported. You need to use the PG_STAT_ACTIVITY view (see **PG_STAT_ACTIVITY**) to directly connect to the target node for query.

  Return type: record.

Name	Type	Description
coorname	text	Name of a CN in the current cluster.
datid	oid	OID of the database that the user session connects to in the backend.
datname	text	Name of the database that the user session connects to in the backend.
pid	bigint	Backend thread ID.
sessionid	bigint	Session ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
usesysid	oid	OID of the user logged in to the backend.
usename	text	Name of the user logged in to the backend.
application_name	text	Name of the application connected to the backend.
client_addr	inet	IP address of the client connected to the backend. If this column is **NULL**, it indicates either the client is connected via a Unix socket on the server or this is an internal thread, such as **AUTOVACUUM**.
client_hostname	text	Host name of the connected client, as reported by a reverse DNS lookup of **client_addr**. This column will be non-null only for IP connections and only when **log_hostname** is enabled.
client_port	integer	TCP port number that the client uses for communication with this backend (**–1** if a Unix socket is used).
backend_start	timestamp with time zone	Time when this process was started, that is, when the client connected to the server.
xact_start	timestamp with time zone	Time when current transaction was started (null if no transaction is active). If the current query is the first of its transaction, the value of this column is the same as that of the **query_start** column.
query_start	timestamp with time zone	Time when the currently active query was started, or time when the last query was started if **state** is not **active** For a stored procedure, function, or package, the first query time is queried and does not change with the running of statements in the stored procedure.
state_change	timestamp with time zone	Time when **state** was last modified.
waiting	Boolean	Specifies whether the backend is currently waiting for a lock. If the backend is currently waiting for a lock, the value is **true**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
enqueue	text	Queuing status of a statement. The value can be: • **waiting in queue**: The statement is in the queue. • **Empty**: The statement is running.
state	text	Overall status of the backend. The value can be: • **active**: The backend is executing a query. • **idle**: The backend is waiting for a new client command. • **idle in transaction**: The backend is in a transaction, but there is no statement being executed in the transaction. • **idle in transaction** (aborted): The backend is in a transaction, but there are statements failed in the transaction. • **fastpath function call**: The backend is executing a fast-path function. • **disabled**: This state is reported if **track_activities** is disabled in this backend. **NOTE** Only system administrators can view the session status of their accounts. The state information of other accounts is empty. For example, after user **judy** is connected to the database, the **state** information of user **joe** and the initial user **omm** in the PGXC_STAT_ACTIVITY view is empty. SELECT datname, usename, usesysid, state,pid FROM pgxc_stat_activity; datname | usename | usesysid | state | pid ----------+---------+----------+--------+----------------- testdb | omm | 10 | | 139968752121616 testdb | omm | 10 | | 139968903116560 db_tpcds | judy | 16398 | active | 139968391403280 testdb | omm | 10 | | 139968643069712 testdb | omm | 10 | | 139968680818448 testdb | joe | 16390 | | 139968563377936 (6 rows)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
resource_pool	name	Resource pool used by the user.
query_id	bigint	ID of a query.
query	text	Latest query at the backend. If the value of **state** is **active**, this column shows the ongoing query. In all other states, it shows the last query that was executed.
global_sessionid	text	Global session ID.
unique_sql_id	bigint	Unique SQL statement ID.
trace_id	text	Driver-specific trace ID, which is associated with an application request.
top_xid	xid	Top-level transaction ID of a transaction.
current_xid	xid	Current transaction ID of a transaction.
xlog_quantity	bigint	Size of Xlogs currently used by a transaction, in bytes.

- pgxc_stat_activity_with_conninfo()

  Description: Displays query information about the current user on all CNs in the current cluster. For details, see the **PGXC_STAT_ACTIVITY** view. Only users with the SYSADMIN or MONADMIN permission can execute this function, and common users can view only their own information.

  Return type: record.

- pgxc_stat_all_tables()

  Description: Displays statistics on a row in each table (including TOAST tables) on each node. Only users with the SYSADMIN or MONADMIN permission can execute this function.

  Return type: record.

- pgxc_get_thread_wait_status()

  Description: Queries the call hierarchy between threads generated by all SQL statements on each node in a cluster and the blocking status of each thread.

  Return type: record.

- pv_session_memory

  Description: Collects statistics on memory usage at the session level, including all the memory allocated to GaussDB and stream threads on DNs for jobs currently executed by users.

  NOTE

  If the GUC parameter **enable_memory_limit** is set to **off**, this function cannot be used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record.

Table 1-231 Return values

Name	Type	Description
sessid	text	Thread start time and ID.
init_mem	integer	Memory allocated to the currently executed jobs before they enter the executor, in MB.
used_mem	integer	Memory allocated to the currently executed jobs, in MB.
peak_mem	integer	Peak memory allocated to the currently executed jobs, in MB.

- DBE_PERF.gs_stat_activity_timeout(int)

  Description: Obtains information about query jobs whose execution time exceeds the timeout threshold on the current node. The correct result can be returned only when the GUC parameter **track_activities** is set to **on**. The timeout threshold ranges from 0 to 2147483. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: SETOF record

Name	Type	Description
database	name	Name of the database to which a user session is connected.
pid	bigint	Backend thread ID.
sessionid	bigint	Session ID.
usesysid	oid	OID of the user logged in to the backend.
application_name	text	Name of the application connected to the backend.
query	text	Query that is being executed on the backend.
xact_start	timestamptz	Time when the current transaction is started.
query_start	timestamptz	Time when the current query starts For a stored procedure, function, or package, the first query time is queried and does not change with the running of statements in the stored procedure.
query_id	bigint	ID of a query.

- DBE_PERF.global_stat_activity_timeout(int)

  Description: Obtains information about query jobs whose execution time exceeds the timeout threshold in the current system (all CNs). The correct

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

result can be returned only when the GUC parameter **track_activities** is set to **on**. The timeout threshold ranges from 0 to 2147483. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

Return type: SETOF record

Name	Type	Description
nodename	text	Name of the CN connected to the user session.
database	name	Name of the database to which a user session is connected.
pid	bigint	Backend thread ID.
sessionid	bigint	Session ID.
usesysid	oid	OID of the user logged in to the backend.
application_name	text	Name of the application connected to the backend.
query	text	Query that is being executed on the backend.
xact_start	timestamptz	Time when the current transaction is started.
query_start	timestamptz	Time when the current query starts For a stored procedure, function, or package, the first query time is queried and does not change with the running of statements in the stored procedure.
query_id	bigint	ID of a query.

- DBE_PERF.get_global_active_session()

  Description: Displays a summary of samples in the ACTIVE SESSION PROFILE memory on all CNs and primary DNs. After a cluster is created, by default, you must have the MONADMIN permission to query this function. The **event_start_time**, **current_xid**, and **top_xid** columns are added to GaussDB Kernel 505.0. After GaussDB Kernel is upgraded to GaussDB Kernel 505.0 or later, the columns cannot be queried during the upgrade observation period.

  Return type: record.

- DBE_PERF.get_global_os_runtime()

  Description: Displays the running status of the current OS. This function can be queried only on CNs. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_os_threads()

  Description: Provides thread status information on all normal nodes in the entire cluster. The information can be queried only on CNs. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record.

- DBE_PERF.get_global_os_threads()

  Description: Provides thread status information on all normal nodes in the entire cluster. The information can be queried only on CNs. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_workload_sql_count()

  Description: Provides the count information of SELECT, UPDATE, INSERT, DELETE, DDL, DML, and DCL in different loads in the entire cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_workload_sql_elapse_time()

  Description: Provides SELECT, UPDATE, INSERT, DELETE, and response time information (TOTAL, AVG, MIN, and MAX) in different loads in the entire cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_workload_transaction()

  Description: Obtains the transaction volume and time information on all nodes in the cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_session_stat()

  Description: Obtains the session status information on all nodes in the cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

  NOTE

  The status information contains the following 14 items: commit, rollback, sql, table_scan, blocks_fetched, physical_read_operation, shared_blocks_dirtied, local_blocks_dirtied, shared_blocks_read, local_blocks_read, blocks_read_time, blocks_write_time, sort_imemory, and sort_idisk.

- DBE_PERF.get_global_session_time()

  Description: Provides the time consumed in each key phase on each node in the entire cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_session_memory()

  Description: Aggregates statistics on memory usage at the session level on each node in the unit of MB, including all the memory allocated to GaussDB and stream threads on DNs for jobs currently executed by users. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_PERF.get_global_session_memory_detail()

  Description: Aggregates statistics on thread memory usage on each node by the MemoryContext node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_session_stat_activity()

  Description: Aggregates information about running threads on each node in the cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_thread_wait_status()

  Description: Aggregates the blocking waiting status of the backend thread and auxiliary thread on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_operator_history_table()

  Description: Aggregates the operator records (persistent) after jobs are executed by the current user on all CNs. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_operator_history()

  Description: Aggregates the operator records after jobs are executed by the current user on all CNs. After a cluster is created, by default, you must have the MONADMIN or SYSADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_operator_runtime()

  Description: Aggregates real-time operator records of jobs executed by the current user on all CNs. After a cluster is created, by default, you must have the MONADMIN or SYSADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_statement_complex_history()

  Description: Aggregates the historical records of complex queries executed by the current user on all CNs. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_statement_complex_history_table()

  Description: Aggregates the historical records of complex queries (persistent) executed by the current user on all CNs. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_statement_complex_runtime()

  Description: Aggregates real-time information about complex queries executed by the current user on all CNs. After a cluster is created, by default, you must have the MONADMIN or SYSADMIN permission to query this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record.

- DBE_PERF.get_global_memory_node_detail()

  Description: Aggregates the memory usage of a database on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_shared_memory_detail()

  Description: Aggregates the usage information about the shared memory contexts on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_comm_delay()

  Description: Aggregates the communications library delay status on all DNs. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_comm_recv_stream()

  Description: Aggregates the status of the stream received by the communications library on all DNs. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_comm_send_stream()

  Description: Aggregates the status of the stream sent by the communications library on all DNs. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_comm_status()

  Description: Aggregates the status of the communications library on all DNs. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_statio_all_indexes()

  Description: Aggregates index information and I/O statistics in the current database on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_local_toastname_and_toastindexname()

  Description: Provides the mapping between the name and index of the local TOAST table and its associated tables. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_statio_all_indexes()

  Description: Collects I/O statistics on specific indexes, covering all index lines in the current database on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record.

- DBE_PERF.get_global_statio_all_sequences()

  Description: Provides I/O status information about all sequences in the namespace. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_statio_all_tables()

  Description: Aggregates I/O statistics on each table in the database on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_statio_all_tables()

  Description: Collects statistics on I/Os of each table in the cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_local_toast_relation()

  Description: Provides the mapping between the name of the local TOAST table and its associated tables. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_statio_sys_indexes()

  Description: Aggregates I/O status information about all system catalog indexes in namespaces on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_statio_sys_indexes()

  Description: Collects statistics on I/O status information about all system catalog indexes in the namespaces on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_statio_sys_sequences()

  Description: Provides I/O status information about all system catalog sequences in namespaces. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_statio_sys_tables()

  Description: Provides I/O status information about all system catalogs in the namespaces on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_statio_sys_tables()

  Description: Collects I/O status information about all system catalogs in the namespaces of the cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record.

- DBE_PERF.get_global_statio_user_indexes()

  Description: Provides I/O status information about all user relationship table indexes in the namespaces on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_statio_user_indexes()

  Description: Aggregates I/O status information about all user relationship table indexes in the namespaces of the cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_statio_user_sequences()

  Description: Displays I/O status information about all user sequences in the namespaces on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_statio_user_tables()

  Description: Displays I/O status information about all user relationship tables in the namespaces on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_statio_user_tables()

  Description: Aggregates I/O status information about all user relationship tables in the namespaces of the cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_dn_stat_all_tables()

  Description: Aggregates statistics on all tables in the database on each DN. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_cn_stat_all_tables()

  Description: Aggregates statistics on all tables in the database on each CN. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_dn_stat_all_tables()

  Description: Collects statistics on all tables in the database on each DN. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_cn_stat_all_tables()

  Description: Collects statistics on all tables in the database on each CN. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record.

- DBE_PERF.get_global_stat_all_indexes()

  Description: Aggregates statistics on all indexes in the database on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_stat_all_indexes()

  Description: Collects statistics on all indexes in the database on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_stat_sys_tables()

  Description: Aggregates statistics on the system catalogs of all the namespaces in the pg_catalog or information_schema schema on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_stat_sys_tables()

  Description: Collects statistics on the system catalogs of all the namespaces in the pg_catalog or information_schema schema on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_stat_sys_indexes()

  Description: Aggregates index status information about all system catalogs in the pg_catalog or information_schema schema on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_stat_sys_indexes()

  Description: Collects index status information about all system catalogs in the pg_catalog or information_schema schema on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_stat_user_tables()

  Description: Aggregates status information about user-defined ordinary tables in all namespaces. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_stat_user_tables()

  Description: Collects status information about user-defined ordinary tables in all namespaces. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_PERF.get_global_stat_user_indexes()

  Description: Aggregates status information about the indexes of user-defined ordinary tables in all databases. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_stat_user_indexes()

  Description: Collects status information about the indexes of user-defined ordinary tables in all databases. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_stat_database()

  Description: Aggregates statistics on databases on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_stat_database_conflicts()

  Description: Collects statistics on databases on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_stat_xact_all_tables()

  Description: Aggregates transaction status information about all ordinary tables and TOAST tables in namespaces. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_stat_xact_all_tables()

  Description: Collects transaction status information about all ordinary tables and TOAST tables in the namespace. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_stat_xact_sys_tables()

  Description: Aggregates transaction status information about system catalogs in namespaces on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_stat_xact_sys_tables()

  Description: Collects transaction status information about system catalogs in namespaces on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_stat_xact_user_tables()

  Description: Aggregates transaction status information about user tables in namespaces on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_PERF.get_summary_stat_xact_user_tables()

  Description: Collects transaction status information about user tables in namespaces on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_stat_user_functions()

  Description: Aggregates transaction status information about user-defined functions in namespaces on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_stat_xact_user_functions()

  Description: Collects transaction status information about user-defined functions in namespaces on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_stat_bad_block()

  Description: Aggregates information about the failure to read files such as tables and indexes on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_file_redo_iostat()

  Description: Collects information about the failure to read files such as tables and indexes on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_file_iostat()

  Description: Collects I/O statistics of data files on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_locks()

  Description: Aggregates lock information on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_replication_slots()

  Description: Aggregates logical replication information on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.GET_GLOBAL_PARALLEL_DECODE_STATUS()

  Description: Displays the parallel decoding information of replication slots on all primary DNs in a cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function. This function can be executed only on CNs. An error is reported when it is executed on DNs. The

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

returned value is the same as that of the view in **DBE_PERF.GLOBAL_PARALLEL_DECODE_STATUS**.

Return type: record.

- DBE_PERF.GET_GLOBAL_PARALLEL_DECODE_THREAD_INFO()

  Description: Displays the parallel decoding thread information of replication slots on all primary DNs in a cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function. This function can be executed only on CNs. An error is reported when it is executed on DNs. The returned value is the same as that of the view in **DBE_PERF.GLOBAL_PARALLEL_DECODE_THREAD_INFO**.

  Return type: record.

- DBE_PERF.get_global_bgwriter_stat()

  Description: Aggregates statistics on the background write thread activities on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_replication_stat()

  Description: Aggregates status information about log synchronization on all nodes, such as the location where the sender sends logs and the location where the receiver receives logs. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_pooler_status()

  Description: Aggregates cache connection status in the pooler on all CNs. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_transactions_running_xacts()

  Description: Aggregates information about running transactions on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_transactions_running_xacts()

  Description: Collects information about running transactions on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_transactions_prepared_xacts()

  Description: Aggregates information about transactions that are currently prepared for two-phase commit on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_transactions_prepared_xacts()

  Description: Collects information about transactions that are currently prepared for two-phase commit on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record.

- DBE_PERF.get_summary_statement()

  Description: Aggregates the status of historical statements executed on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function. For details about the columns returned by the function, see **DBE_PERF.SUMMARY_STATEMENT**.

  Return type: record.

- DBE_PERF.get_global_statement_count()

  Description: Aggregates SELECT, UPDATE, INSERT, DELETE, and response time information (TOTAL, AVG, MIN, and MAX) on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_config_settings()

  Description: Aggregates GUC parameter settings on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_wait_events()

  Description: Aggregates status information about the wait events on each node. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_statement_responsetime_percentile()

  Description: Obtains the distribution information about the response time of 80% and 95% SQL statements in the cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_summary_user_login()

  Description: Collects statistics on the number of user login and logout times on each node in the cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.get_global_record_reset_time()

  Description: Aggregates statistics on reset (restart, primary-standby switchover, and database deletion) time in the cluster. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- DBE_PERF.standby_statement_history(only_slow[, time1, time2])

  Description: Queries full SQL statements on the standby node. The primary node queries full SQL statements using the statement_history table, while the standby node queries using this function. To execute this function, you must have the MONADMIN permission. For details about the columns returned by the function, see **STATEMENT_HISTORY**.

  Parameters: See **Table 1-232**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record.

Table 1-232 standby_statement_history parameters

Parameter	Type	Description
only_slow	Boolean	Specifies whether to query only slow SQL statements. • **true** indicates yes, which is equivalent to **select .. where is_slow_sql = true**. • **false** or **NULL** indicates that all SQL statements are queried, that is, **is_slow_sql** is not used as a filter condition.
time1	timestamptz	Minimum time specified by **finish_time** for querying SQL statements. This parameter is optional.
time2	timestamptz	Maximum time specified by **finish_time** for querying SQL statements. This parameter is optional. An error is reported during function execution if **time1** is greater than or equal to **time2**.

NOTE

- The two time parameters **time1** and **time2** indicate the time segment to which **finish_time** of the queried SQL statement belongs. They indicate the start time and end time respectively. If **NULL** or no value is entered, there is no limit. The function of **time1** and **time2** is the same as that of **select .. where finish_time between time1 and time2**.
- The data generated from this function on the standby node is not stored in a table, and there is no index on the **start_time** column. You are advised to use the parameter to search for **finish_time**.
- Full/Slow SQL statements on the standby node are written to disks asynchronously. Therefore, the storage of user SQL information may be delayed. You are advised to query this API to expand the query time range.

- DBE_PERF.global_io_wait_info()

  Description: Queries real-time I/O control statistics on all CNs and DNs. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- pg_stat_get_mem_mbytes_reserved(tid)

  Description: Collects statistics on variables related to resource management, which is used only for fault locating.

  Parameter: thread ID.

  Return type: text.

- pg_stat_get_file_stat()

  Description: Collects statistics on data file I/Os to indicate I/O performance and detect performance problems such as abnormal I/O operations.

  Return type: record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_stat_get_redo_stat()

  Description: Displays statistics on the replay of session thread logs.

  Return type: record.

- pg_stat_get_status(int8)

  Description: Tests the blocking status about the backend thread and auxiliary thread of the current instance.

  Return type: record.

- get_local_rel_iostat()

  Description: Queries the accumulated I/O status of data files on the current node.

  Return type: record.

- DBE_PERF.get_global_rel_iostat()

  Description: Aggregates I/O statistics on data files on all nodes. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- pg_catalog.plancache_status(is_with_plan)

  Description: Specifies whether to display the GPC status information on the current node, that is, the **plan_info** column. The input parameter **is_with_plan** is of the Boolean type. The default value is **false**. If the input parameter is **true**, the **plan_info** information is displayed. The information returned by the function is the same as that in **DBE_PERF.GLOBAL_PLANCACHE_STATUS**. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

  NOTE

  - When the input parameter of **pg_catalog.plancache_status** is set to **true** and there are a large number of plans, the performance may deteriorate. You are advised to use the pg_catalog.global_plancache_show_plan function to query the detailed plan information of a single SQL statement.
  - If you do not have the permission on the table on which the plan depends, an error message indicating insufficient permission is displayed in the **plan_info** column.

- DBE_PERF.global_plancache_status(is_with_plan)

  Description: Specifies whether to display the GPC status information on all nodes, that is, the **plan_info** column. The input parameter **is_with_plan** is of the Boolean type. The default value is **false**. If the input parameter is **true**, the **plan_info** information is displayed. For details about the information returned by the function, see **DBE_PERF.GLOBAL_PLANCACHE_STATUS**. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When the input parameter of **DBE_PERF.global_plancache_status** is set to **true** and there are a large number of plans, the performance may deteriorate. You are advised to use the pg_catalog.global_plancache_show_plan function to query the detailed plan information of a single SQL statement.
- If you do not have the permission on the table on which the plan depends, an error message indicating insufficient permission is displayed in the **plan_info** column.

- pg_catalog.gs_plancache_stat()

  Description: Displays information about parameterized statements of all sessions in the current process. The information returned by the function is consistent with **DBE_PERF.GS_PLANCACHE_STAT**. To query this function, you must have the MONADMIN permission.

  Return type: record.

- pg_catalog.gs_dist_plancache_stat()

  Description: Displays information about parameterized statements of all processes on the standby node. The information returned by the function is consistent with **DBE_PERF.GS_DIST_PLANCACHE_STAT**. To query this function, you must have the MONADMIN permission.

  Return type: record.

- DBE_PERF.global_threadpool_status()

  Description: Displays the status of worker threads and sessions in thread pools on all nodes. For details about the information returned by the function, see **DBE_PERF.GLOBAL_THREADPOOL_STATUS**. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Return type: record.

- comm_check_connection_status()

  Description: Returns the connection status between the CN and all active nodes (CNs and primary DNs). This function can be queried only on CNs and can be used by common users.

  Parameter: nan.

  Return types: node_name text, remote_name text, remote_host text, remote_port integer, is_connected boolean, and no_error_occur boolean.

- DBE_PERF.global_comm_check_connection_status()

  Description: Returns the connection status between all CNs and all active nodes (CNs and primary DNs). This function can be queried only on CNs. Permission control is inherited from the **DBE_PERF** schema. After a cluster is created, by default, you must have the MONADMIN permission to query this function.

  Parameter: nan.

  Return types: node_name text, remote_name text, remote_host text, remote_port integer, is_connected boolean, and no_error_occur boolean.

- remote_candidate_stat()

  Description: Displays the number of pages in the candidate buffer chain of this instance and buffer elimination information, including the normal buffer pool and segment buffer pool.

  Return type: record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-233 remote_candidate_stat parameters

Name	Type	Description
node_name	text	Node name.
candidate_slots	integer	Number of pages in the candidate buffer chain of the current normal buffer pool.
get_buf_from_list	bigint	Number of times that pages are obtained from the candidate buffer chain during buffer eviction in the current normal buffer pool.
get_buf_clock_sweep	bigint	Number of times that pages are obtained from the original eviction solution during buffer eviction in the current normal buffer pool.
seg_candidate_slots	integer	Number of pages in the candidate buffer chain of the current segment buffer pool.
seg_get_buf_from_list	bigint	Number of times that pages are obtained from the candidate buffer chain during buffer eviction in the current segment buffer pool.
seg_get_buf_clock_sweep	bigint	Number of times that pages are obtained from the original eviction solution during buffer eviction in the current segment buffer pool.

- remote_ckpt_stat()

  Description: Displays the checkpoint information and log flushing information about all instances in the cluster (unavailable on DNs, except for the current node).

  Return type: record.

  Table 1-234 remote_ckpt_stat parameters

Parameter	Type	Description
node_name	text	Instance name.
ckpt_redo_point	text	Checkpoint of the current instance.
ckpt_clog_flush_num	int8	Number of Clog flushing pages from the startup time to the current time.
ckpt_csnlog_flush_num	int8	Number of CSN log flushing pages from the startup time to the current time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
ckpt_multixact_flush_num	int8	Number of MultiXact flushing pages from the startup time to the current time.
ckpt_predicate_flush_num	int8	Number of predicate flushing pages from the startup time to the current time.
ckpt_twophase_flush_num	int8	Number of two-phase flushing pages from the startup time to the current time.

- remote_double_write_stat()

  Description: Displays doublewrite file status of all instances in the cluster (unavailable on DNs, except for the current node).

  Return type: record.

  Table 1-235 remote_double_write_stat parameters

Parameter	Type	Description
node_name	text	Instance name.
curr_dwn	int8	Sequence number of the doublewrite file.
curr_start_page	int8	Start page for restoring the doublewrite file.
file_trunc_num	int8	Number of times that the doublewrite file is reused.
file_reset_num	int8	Number of reset times after the doublewrite file is full.
total_writes	int8	Total number of I/Os of the doublewrite file.
low_threshold_writes	int8	Number of I/Os for writing doublewrite files with low efficiency (the number of I/O flushing pages at a time is less than 16).
high_threshold_writes	int8	Number of I/Os for writing doublewrite files with high efficiency (the number of I/O flushing pages at a time is more than 421).
total_pages	int8	Total number of pages that are flushed to the doublewrite file area.
low_threshold_pages	int8	Number of pages that are flushed with low efficiency.
high_threshold_pages	int8	Number of pages that are flushed with high efficiency.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
file_id	int8	ID of the current doublewrite file.

- remote_single_flush_dw_stat()

  Description: Displays the single-page doublewrite file eviction status of all instances in the cluster (unavailable on DNs, except for the current node).

  Return type: record.

  Table 1-236 remote_single_flush_dw_stat parameters

Parameter	Type	Description
node_name	text	Instance name.
curr_dwn	integer	Sequence number of the doublewrite file.
curr_start_page	integer	Start position of the current doublewrite file.
total_writes	bigint	Total number of data write pages in the current doublewrite file.
file_trunc_num	bigint	Number of times that the doublewrite file is reused.
file_reset_num	bigint	Number of reset times after the doublewrite file is full.

- remote_pagewriter_stat()

  Description: Displays the page flushing information and checkpoint information about all instances in the cluster (unavailable on DNs, except for the current node).

  Return type: record.

  Table 1-237 remote_pagewriter_stat parameters

Parameter	Type	Description
node_name	text	Instance name.
pgwr_actual_flush_total_num	int8	Total number of dirty pages flushed from the startup time to the current time.
pgwr_last_flush_num	int4	Number of dirty pages flushed in the previous batch.
remain_dirty_page_num	int8	Estimated number of dirty pages that are not flushed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
queue_head_page_rec_lsn	text	**recovery_lsn** of the first dirty page in the dirty page queue of the current instance.
queue_rec_lsn	text	**recovery_lsn** of the dirty page queue of the current instance.
current_xlog_insert_lsn	text	Position of writing Xlogs in the current instance.
ckpt_redo_point	text	Checkpoint of the current instance.

- remote_recovery_status()

  Description: Displays log flow control information about the primary and standby nodes (except the current node and DNs).

  Return type: record.

  Table 1-238 remote_recovery_status parameters

Parameter	Type	Description
node_name	text	Node name (including the primary and standby nodes).
standby_node_name	text	Name of the standby node.
source_ip	text	IP address of the primary node.
source_port	int4	Port number of the primary node.
dest_ip	text	IP address of the standby node.
dest_port	int4	Port number of the standby node.
current_rto	int8	Current log flow control time of the standby node (unit: s).
target_rto	int8	Expected flow control time of the standby node specified by the corresponding GUC parameter (unit: s).
current_sleep_time	int8	Sleep time required to achieve the expected flow control time (unit: μs).

- remote_rto_stat()

  Description: Displays log flow control information about the primary and standby nodes (except the current node and DNs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record.

Table 1-239 remote_rto_stat parameters

Parameter	Type	Description
node_name	text	Node name (including the primary and standby nodes).
rto_info	text	Flow control information, including the current log flow control time (unit: second) of the standby node, the expected flow control time (unit: second) specified by the GUC parameter, and the primary node sleep time (unit: μs) required to reach the expectation.

- remote_redo_stat()

  Description: Displays the log replay status of all instances in the cluster (unavailable on DNs, except for the current node).

  Return type: record.

  Table 1-240 remote_redo_stat parameters

Parameter	Type	Description
node_name	text	Instance name.
redo_start_ptr	int8	Start point for replaying the instance logs.
redo_start_time	int8	Start time (UTC) when the instance logs are replayed.
redo_done_time	int8	End time (UTC) when the instance logs are replayed.
curr_time	int8	Current time (UTC) of the instance.
min_recovery_point	int8	Position of the minimum barrier for the instance logs.
read_ptr	int8	Position for reading the instance logs.
last_replayed_read_ptr	int8	Position for replaying the instance logs.
recovery_done_ptr	int8	Replay position after the instance is started.
read_xlog_io_counter	int8	Number of I/Os when the current instance reads and replays logs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
read_xlog_io_total_dur	int8	Total I/O latency when the current instance reads and replays logs.
read_data_io_counter	int8	Number of data page I/O reads during replay in the current instance.
read_data_io_total_dur	int8	Total I/O latency of data page reads during replay in the current instance.
write_data_io_counter	int8	Number of data page I/O writes during replay in the current instance.
write_data_io_total_dur	int8	Total I/O latency of data page writes during replay in the current instance.
process_pending_counter	int8	Number of synchronization times of log distribution threads during replay in the instance.
process_pending_total_dur	int8	Total synchronization latency of log distribution threads during replay in the instance.
apply_counter	int8	Number of synchronization times of redo threads during redo in the current instance
apply_total_dur	int8	Total synchronization latency of redo threads during redo in the current instance
speed	int8	Log replay rate of the current instance.
local_max_ptr	int8	Maximum number of replay logs received by the localhost after the instance is started.
primary_flush_ptr	int8	Log point where the host flushes logs to a disk.
worker_info	text	Replay thread information of the instance. If parallel replay is not enabled, the value is null.

- pgxc_gtm_snapshot_status()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Queries for transaction information on the current GTM. This function is supported only in GTM mode, and is not supported in GTM-LITE or GTM-FREE mode.

Return type: record.

The following table describes return parameters.

Table 1-241 PGXC_GTM_SNAPSHOT_STATUS return parameters

Name	Type	Description
xmin	xid	Minimum XID of the running transactions.
xmax	xid	XID of the transaction next to the executed transaction with the maximum XID.
csn	integer	Specifies the sequence number of the transaction to be committed.
oldestxmin	xid	Minimum XID of the executed transactions.
xcnt	integer	Number of the running transactions.
running_xids	text	XID of the running transaction.

- pv_os_run_info()

  Description: Displays the running status of the current OS. For details about the columns, see **PV_OS_RUN_INFO**.

  Parameter: nan.

  Return type: SETOF record
- pv_session_stat()

  Description: Collects session status information by session thread or autovacuum thread. For details about the columns, see **PV_SESSION_STAT**.

  Parameter: nan.

  Return type: SETOF record
- pv_session_time()

  Description: Collects statistics on the running time of session threads and the time consumed in each execution phase. For details about the columns, see **PV_SESSION_TIME**.

  Parameter: nan.

  Return type: SETOF record
- pg_stat_get_db_temp_bytes()

  Description: Collects statistics on the total amount of data written to temporary files through database query. All temporary files are calculated, regardless of why they were created or how **log_temp_files** is configured.

  Parameter: oid.

  Return type: bigint.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_stat_get_db_temp_files()

  Description: Queries the number of temporary files created in the database. All temporary files are calculated, regardless of why they were created (such as sorting or hashing) or how **log_temp_files** is configured.

  Parameter: oid.

  Return type: bigint.

- gs_prepared_statements()

  Description: Displays prepared statements that are available in all sessions. Only users with the SYSADMIN permission can execute this function. The columns in the information returned by the function are the same as those in **GS_ALL_PREPARED_STATEMENTS**.

  Return type: record.

- local_redo_time_count([int])

  Description: Returns the time consumption statistics of the replay process on the current node (valid data exists only on the standby node). In parallel replay mode, it returns the process duration statistics of each replayer thread. In ultimate RTO mode, it returns the process duration statistics of each replayer thread and the process duration statistics of different log types.

  Parameters:

Parameter	Type	Description
operation	int	Optional. If no value is entered, statistics are queried. ● **–1** (default): disables the statistics function. ● **0**: enables the statistics function. ● **1**: queries statistics. ● **2**: resets statistics. After the statistics function is disabled and then enabled again, the previous statistics are cleared.

  The return values are as follows.

  Table 1-242 local_redo_time_count return parameters

Column	Description
thread_name	Thread name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
step1_total	Total duration of step 1. The process of each thread is as follows: ● Ultimate RTO: **redo batch**: obtains a log from a queue. **redo manager**: obtains a log from a queue. **redo worker**: obtains a log from a queue. **trxn manager**: reads a log from a queue. **trxn worker**: reads a log from a queue. **read worker**: reads an Xlog page (overall) from a file. **read page worker**: obtains a log from a queue. **startup**: obtains a log from a queue. ● Parallel replay: **page redo**: obtains a log from a queue. **startup**: reads a log.
step1_count	Number of accumulated execution times of step 1.
step2_total	Total duration of step 2. The process of each thread is as follows: ● Ultimate RTO: **redo batch**: processes logs (overall). **redo manager**: processes logs (overall). **redo worker**: processes logs (overall). **trxn manager**: processes logs (overall). **trxn worker**: processes logs (overall). **read worker**: specifies the time required for reading the Xlog page. **read page worker**: generates and sends LSN forwarders. **startup**: checks whether to replay to the specified position. ● Parallel replay: **startup**: checks whether to replay to the specified position.
step2_count	Number of accumulated execution times of step 2.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
step3_total	Total duration of step 3. The process of each thread is as follows: ● Ultimate RTO: **redo batch**: updates the standby state. **redo manager**: processes data logs. **redo worker**: replays page logs (overall). **trxn manager**: updates the flushing LSN. **trxn worker**: replays logs. **read worker**: updates the Xlog segment. **read page worker**: obtains a new item. **startup**: collects statistics on the wait time of delayed replay feature. ● Parallel replay: **page redo**: updates the standby state. **startup**: collects statistics on the wait time of delayed replay feature.
step3_count	Number of accumulated execution times of step 3.
step4_total	Total duration of step 4. The process of each thread is as follows: ● Ultimate RTO: **redo batch**: parses Xlogs. **redo manager**: processes DDL operations. **redo worker**: reads data pages. **trxn manager**: synchronizes the wait time. **trxn worker**: updates the LSN of the current thread. **read page worker**: stores logs in the distribution thread. **startup**: distributes logs (overall). ● Parallel replay: **page redo**: replays undo logs. **startup**: distributes logs (overall).
step4_count	Number of accumulated execution times of step 4.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
step5_total	Total duration of step 5. The process of each thread is as follows: ● Ultimate RTO: **redo manager**: processes database logs. **redo worker**: replays data page logs. **trxn manager**: distributes data to the trxn worker. **trxn worker**: completes waiting for the notification of strong synchronization logs. **read page worker**: updates the LSN of the current thread. **startup**: decodes logs. ● Parallel replay: **page redo**: replays share trxn logs. **startup**: replays logs.
step5_count	Number of accumulated execution times of step 5.
step6_total	Total duration of step 6. The process of each thread is as follows: ● Ultimate RTO: **redo worker**: replays non-data page logs. **trxn manager**: updates global LSNs. ● Parallel replay: **page redo**: replays synctrxn logs. **startup**: forcibly synchronizes the wait time.
step6_count	Number of accumulated execution times of step 6.
step7_total	Total duration of step 7. The process of each thread is as follows: ● Ultimate RTO: **redo manager**: creates tablespaces. **redo worker**: updates FSM. ● Parallel replay: **page redo**: replays a single log.
step7_count	Number of accumulated execution times of step 7.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
step8_total	Total duration of step 8. The process of each thread is as follows: ● Ultimate RTO: **redo worker**: forcibly synchronizes the wait time. **redo manager**: processes segment-page logs. ● Parallel replay: **page redo**: replays all workers do logs.
step8_count	Number of accumulated execution times of step 8.
step9_total	Total duration of step 9. The process of each thread is as follows: ● Ultimate RTO: **redo manager**: processes segment-page undo logs. ● Parallel replay: **page redo**: replays multi-worker do logs.
step9_count	Number of accumulated execution times of step 9.

Process duration statistics of the log type (If this function is disabled, replayed logs are sampled for statistics.)

Column	Process Description
thread_name	Name of the sample.
step1_total	Ultimate RTO: ● For data logs, it specifies the total time required for processing logs, including step 2. ● For transaction logs, it replays transaction logs.
step1_count	Number of accumulated execution times of step 1.
step2_total	Ultimate RTO: ● For data logs, it specifies the total time required for replaying page logs, including step 3 and step 4. The main difference between step 2 and (step 3+step 4) is the construction of multi-version pages that perform read on standby. ● For transaction logs, it specifies the synchronization waiting time before transaction thread replay.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Process Description
step2_count	Number of accumulated execution times of step 2.
step3_total	Ultimate RTO: • For data logs, it reads data pages.
step3_count	Number of accumulated execution times of step 3.
step4_total	Ultimate RTO: • For data logs, it replays data page logs.
step4_count	Number of accumulated execution times of step 4.
step5_total	Ultimate RTO: • For transaction logs, it records the replay time of the redo_worker thread.
step5_count	Number of accumulated execution times of step 5.
step6_total	Ultimate RTO: • For data logs, it updates FSM.
step6_count	Number of accumulated execution times of step 6.
Step7_total	Ultimate RTO: • For transaction logs, it records the synchronization wait time of the redo_worker thread.
Step7_count	Number of accumulated execution times of step 7.

- local_xlog_redo_statics()

  Description: Returns the statistics on each type of logs that have been replayed on the current node (valid data exists only on the standby node).

  The return values are as follows.

  Table 1-243 local_xlog_redo_statics parameters

Column	Description
xlog_type	Log types.
rmid	Resource manager ID.
info	Xlog operation.
num	Number of logs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
extra	Valid values are available for page replay logs and xact logs. • Number of pages read from the disk if the log is of the page type. • Number of deleted files if the log is of the xact type.

- remote_bgwriter_stat()

  Description: Displays information about pages flushed by the bgwriter threads of all instances in the cluster, number of pages in the candidate buffer chain, and buffer eviction information (not available on the DN, except for the current node).

  Return type: record.

  Table 1-244 remote_bgwriter_stat parameters

Parameter	Type	Description
node_name	text	Instance name.
bgwr_actual_flush_total_num	bigint	Total number of dirty pages flushed by the bgwriter thread from the startup time to the current time.
bgwr_last_flush_num	integer	Number of dirty pages flushed by the bgwriter thread in the previous batch.
candidate_slots	integer	Number of pages in the current candidate buffer chain.
get_buffer_from_list	bigint	Number of times that pages are obtained from the candidate buffer chain during buffer eviction.
get_buf_clock_sweep	bigint	Number of times that pages are obtained from the original eviction solution during buffer eviction.

  Example:

  The remote_bgwriter_stat function is used to query the page refreshing information of the bgwriter thread.

```
gaussdb=# SELECT * FROM remote_bgwriter_stat();
 node_name | bgwr_actual_flush_total_num | bgwr_last_flush_num | candidate_slots |
get_buffer_from_list | get_buf_clock_sweep
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-----------+-----------------------------+---------------------+-----------------+----------------------
+---------------------
 datanode3 |                           0 |                   0 |          266232 |              404 |                   0
 datanode2 |                           0 |                   0 |          266232 |              424 |                   0
 datanode1 |                           0 |                   0 |          266232 |              393 |                   0
(3 rows)
```

- gs_stack()

  Description: Displays the call stack of a thread. To query this function, you must have the SYSADMIN or MONADMIN permission.

  Parameter: **tid** (thread ID). **tid** is an optional parameter. If it is specified, the function returns the call stack of the thread corresponding to **tid**. If it is not specified, the function returns the call stacks of all threads.

  Return value: If **tid** is specified, the return value is of the TEXT type. If **tid** is not specified, the return value is a SETOF record.

  Example:

  Obtain the call stack of a specified thread.

```
gaussdb=# SELECT * FROM gs_stack(139663481165568);
                        gs_stack
------------------------------------------------------------------------
 __poll + 0x2d                                                         +
 WaitLatchOrSocket(Latch volatile*, int, int, long) + 0x29f        +
 WaitLatch(Latch volatile*, int, long) + 0x2e                    +
 JobScheduleMain() + 0x90f                                     +
 int GaussDbThreadMain<(knl_thread_role)9>(knl_thread_arg*) + 0x456+
 InternalThreadFunc(void*) + 0x2d                               +
 ThreadStarterFunc(void*) + 0xa4                                +
 start_thread + 0xc5                                          +
 clone + 0x6d                                                 +
(1 row)
```

  Obtain the call stacks of all threads.

```
gaussdb=# SELECT * FROM gs_stack();
-[ RECORD
1 ]----------------------------------------------------------------------------------------------------------
tid   | 139670364324352
lwtid | 308
stack | __poll + 0x2d
      | CommWaitPollParam::caller(int (*)(pollfd*, unsigned long, int), unsigned long) + 0x34
      | int comm_socket_call<CommWaitPollParam, int (*)(pollfd*, unsigned long,
int)>(CommWaitPollParam*, int (*)(pollfd*, unsigned long
, int)) + 0x28
      | comm_poll(pollfd*, unsigned long, int) + 0xb1
      | ServerLoop() + 0x72b
      | PostmasterMain(int, char**) + 0x314e
      | main + 0x617
      | __libc_start_main + 0xf5
      | 0x55d38f8db3a7
[ RECORD 2 ]----------------------------------------------------------------------------------------------------
tid   | 139664851859200
lwtid | 520
stack | __poll + 0x2d
      | WaitLatchOrSocket(Latch volatile*, int, int, long) + 0x29f
      | SysLoggerMain(int) + 0xc86
      | int GaussDbThreadMain<(knl_thread_role)17>(knl_thread_arg*) + 0x45d
      | InternalThreadFunc(void*) + 0x2d
      | ThreadStarterFunc(void*) + 0xa4
      | start_thread + 0xc5
      | clone + 0x6d
```

  NOTE

  Concurrent execution of multiple gs_stack() functions is not supported.

- gs_perf_start(duration integer[, freq integer])

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Collects the stack call data of the on-CPU flame graph of each thread on the current node. The function returns the flame graph flushing address. To query this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** or **DBUSER** mode and unavailable in **DISABLE** mode.

The parameters are described as follows.

Parameter	Description	Type	Range
duration	Stack collection duration, in seconds. If a floating-point number is entered, the first digit after the decimal point is rounded off.	integer	1–60
freq	(Optional) Stack collection frequency, in Hz. The default value is **100**. If a floating-point number is entered, the first digit after the decimal point is rounded off.	integer	10–1000

Return type: text.

Example:

Set the frequency to 100 Hz and collect on-CPU stack information for 10s.

```
gaussdb=# SELECT gs_perf_start(10,100);
                                gs_perf_start
-------------------------------------------------------------------------------
 Successfully collected perf flamegraph data, the result filepaths are :          +
 $GAUSSLOG/gs_flamegraph/cn_5001/flamegraph-manual-oncpu-2025-03-03_104812.html.gz;+
 Record 886 cpu clock samples.                                                    +

(1 row)
```

NOTICE

Before executing this function and the following gs_perf_start_offcpu(), gs_perf_start_all(), and gs_perf_start_detail() functions, ensure that gs_perf_service is running properly. During data collection, gs_perf_service needs to apply for a ring buffer. The buffer size is controlled by **/proc/sys/kernel/perf_event_mlock_kb** in the OS. If the error message "perf mmap failed" is displayed during the collection, adjust the size of **/proc/sys/kernel/perf_event_mlock_kb** and perform the collection again.

- gs_perf_start_offcpu(duration integer)

  Description: Collects the stack call data of the off-CPU flame graph of each thread on the current node. The function returns the flame graph flushing address. To query this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** mode and unavailable in **DBUSER** or **DISABLE** mode.

  The parameters are described as follows.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Type	Range
duration	Stack collection duration, in milliseconds. If a floating-point number is entered, the first digit after the decimal point is rounded off.	integer	50~3000

Return type: text.

Example:

Collect off-CPU stack information for 2000 ms.

```
gaussdb=# SELECT gs_perf_start_offcpu(2000);
                                gs_perf_start_offcpu
-------------------------------------------------------------------------------------
 Successfully collected perf flamegraph data, the result filepaths are :          +
 $GAUSSLOG/gs_flamegraph/cn_5001/flamegraph-manual-offcpu-2025-03-03_104918.html.gz;+
 Record 48 sched switch samples.                                                  +

(1 row)
```

- gs_perf_start_detail(duration integer[, freq integer])

  Description: Collects the on-CPU and off-CPU call stack details of all threads on all CPU cores on the host where the current node is located. The function returns the address of the compressed file of multi-core and multi-thread raw data. Each time this command is executed, the historical multi-core and multi-thread data is deleted. To query this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** mode and unavailable in **DBUSER** or **DISABLE** mode.

  The parameters are described as follows.

Parameter	Description	Type	Range
duration	Stack collection duration, in milliseconds. If a floating-point number is entered, the first digit after the decimal point is rounded off.	integer	50~3000
freq	(Optional) Stack collection frequency, in Hz. The default value is **100**. If a floating-point number is entered, the first digit after the decimal point is rounded off.	integer	10–1000

Return type: text.

Example:

Collect multi-core and multi-thread flame graph data for 2000 ms.

```
gaussdb=# SELECT gs_perf_start_detail(2000);
                               gs_perf_start_detail
```

```
-----------------------------------------------------------------------------------------------------
 Successfully collected perf detail data, the result filepath is :                                  +
 $GAUSSLOG/gs_flamegraph/cn_5001/detail-2025-03-03_105043.lz4                                       +
 Please save the file locally, otherwise the file will be deleted during the next data collection.+
 Record 1735211 samples.                                                                            +

(1 row)
```

- gs_perf_start_all(duration integer[, freq integer])

  Description: Collects the stack call data of the on-CPU and off-CPU flame graphs on all nodes of the host where the current node is located. The function returns the flame graph flushing addresses on all nodes. To query this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** mode. In **DBUSER** mode, only on-CPU flame graph data can be collected. In **DISABLE** mode, this function is unavailable.

  The parameters are described as follows.

Parameter	Description	Type	Range
duration	Stack collection duration, in milliseconds. If a floating-point number is entered, the first digit after the decimal point is rounded off.	integer	50~3000
freq	(Optional) Stack collection frequency, in Hz. The default value is **100**. If a floating-point number is entered, the first digit after the decimal point is rounded off.	integer	10–1000

  Return type: text.

  Example:

  Set the frequency to 100 Hz and collect the on-CPU and off-CPU flame graph data on all nodes of the host where the current node is located for 1000 ms.

```
gaussdb=# SELECT gs_perf_start_all(1000,100);
                                        gs_perf_start_all
-------------------------------------------------------------------------------------------------------------
 Successfully collected perf flamegraph data, the result filepaths are :                                   +
 $GAUSSLOG/gs_flamegraph/{nodename}/flamegraph-manual-
oncpu-2025-03-03_105122.html.gz;                    +
 $GAUSSLOG/gs_flamegraph/{nodename}/flamegraph-manual-
offcpu-2025-03-03_105122.html.gz;                   +
 Please visit gs_flamegraph directory(all nodes of current machine) to get results for individual nodes.+
 Record 664 cpu clock samples.                                                                           +
 Record 370 sched switch samples.                                                                         +

(1 row)
```

- gs_perf_list()

  Description: Views the collected on-CPU and off-CPU flame graph data saved on the current node. The multi-core and multi-thread flame graph records are not displayed. This function is available in **ROOT** or **DBUSER** mode and unavailable in **DISABLE** mode.

  Parameter: nan.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: SETOF record

The following table describes the fields returned by the function.

Name	Type	Description
nodename	text	Name of a collection node.
filename	text	Name of a file to be collected and flushed to disk.
collect_type	text	Collection type.

Example:

Query the flame graph generation records saved on the current node.

```
gaussdb=# SELECT * FROM gs_perf_list() limit 10;
 nodename |                          filename                          | collect_type
----------+------------------------------------------------------------+--------------
 cn_5001  | flamegraph-auto-offcpu-2025-03-02_175022.html.gz       | auto
 cn_5001  | flamegraph-cpuspiking-offcpu-2025-03-02_140825.html.gz | cpuspiking
 cn_5001  | flamegraph-auto-offcpu-2025-03-03_063522.html.gz       | auto
 cn_5001  | flamegraph-auto-offcpu-2025-03-03_102523.html.gz       | auto
 cn_5001  | flamegraph-auto-offcpu-2025-03-02_173523.html.gz       | auto
 cn_5001  | flamegraph-manual-offcpu-2025-03-03_025410.html.gz     | manual
 cn_5001  | flamegraph-manual-oncpu-2025-03-03_003428.html.gz      | manual
 cn_5001  | flamegraph-manual-oncpu-2025-03-02_181653.html.gz      | manual
 cn_5001  | flamegraph-manual-oncpu-2025-03-03_031815.html.gz      | manual
 cn_5001  | flamegraph-manual-oncpu-2025-03-03_001013.html.gz      | manual
(10 rows)
```

- gs_perf_query([filename text])

  Description: Queries the historical on-CPU and off-CPU flame graph stack call data. The function collects the function call stacks of each thread, sums up the function running time, and returns the collection result. To query this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** or **DBUSER** mode and unavailable in **DISABLE** mode.

  The parameters are described as follows.

Parameter	Description	Type
filename	Name of a historically collected file. This parameter is optional. The default value is the latest manually collected flame graph.	text

  Return type: SETOF record

  The following table describes the fields returned by the function.

Name	Type	Description
backtrace	text	Stack name (in a tree structure).
period	bigint	Execution time of a stack.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
level	integer	Level of the stack call tree where a stack is located.
sequence	integer	Sequence in the stack call tree after sorting.
thread_name	text	Name of the thread where a stack is located.
overhead	float	Percentage of the stack execution time.

Example:

Query the collected stack information.

```
gaussdb=# SELECT * FROM gs_perf_query() WHERE overhead > 2 AND level < 10;
                                backtrace                                |   period    | level | sequence |
thread_name  | overhead
-------------------------------------------------------------------------------------+-------------+-------+----------
+---------------+----------
 root                                                                    +| 74140000000 |     0 |        1 | root          |
100
                                                                        |             |       |          |               |
    worker                                                                +| 69930000000 |     1 |        2 | worker
|    94.32
                                                                        |             |       |          |               |
    |── start_thread                                                      +| 67620000000 |     2 |        3 | worker
|    91.21
                                                                        |             |       |          |               |
    |   └── ThreadStarterFunc                                             +| 67620000000 |     3 |        4 |
worker       |    91.21
                                                                        |             |       |          |               |
    |       └── internal_thread_func                                      +| 67620000000 |     4 |        5 |
worker       |    91.21
                                                                        |             |       |          |               |
    |         └── int gauss_db_thread_main                                +| 67620000000 |     5 |        6 |
worker       |    91.21
                                                                        |             |       |          |               |
     |            └── backend_run                                         +| 67620000000 |     6 |        7 |
worker       |    91.21
                                                                        |             |       |          |               |
    |              |── PostgresMain                                       +| 67520000000 |     7 |        8 | worker
|    91.07
                                                                        |             |       |          |               |
    |              |  |── exec_simple_query                               +| 64800000000 |     8 |        9 |
worker       |     87.4
                                                                        |             |       |          |               |
    |              |  |  |── OpFusion::opfusion_process                    +| 30130000000 |     9 |       10 |
worker       |    40.64
                                                                        |             |       |          |               |
    |              |  |  |── pg_analyze_and_rewrite                        +| 11290000000 |     9 |     1405 |
worker       |    15.23
                                                                        |             |       |          |               |
    |              |  |  |── pg_plan_queries                               +|  9550000000 |     9 |     2660 |
worker       |    12.88
                                                                        |             |       |          |               |
    |              |  |  |── PortalRun                                     +|  4680000000 |     9 |     4310 | worker
|     6.31
                                                                        |             |       |          |               |
    |              |  |  |── finish_xact_command                           +|  3120000000 |     9 |     4923 |
worker       |     4.21
                                                                        |             |       |          |               |
    |              |  |  |── pg_parse_query                                +|  1520000000 |     9 |     5262 |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
worker       |     2.05
                                                                |           |      |        |           |
    |              |  |  |── OpFusion::opfusion_factory           +|  1500000000 |     9 |     5374 |
worker       |     2.02
                                                                |           |      |        |           |
    txnsnapworker                                                 +|  1640000000 |     1 |     6770 |
txnsnapworker |     2.21
                                                                |           |      |        |           |
    |── start_thread                                              +|  1620000000 |     2 |     6771 |
txnsnapworker |     2.19
                                                                |           |      |        |           |
    |   └── ThreadStarterFunc                                     +|  1620000000 |     3 |     6772 |
txnsnapworker |     2.19
                                                                |           |      |        |           |
    |       └── internal_thread_func                              +|  1620000000 |     4 |     6773 |
txnsnapworker |     2.19
                                                                |           |      |        |           |
    |          └── int gauss_db_thread_main                       +|  1620000000 |     5 |     6774 |
txnsnapworker |     2.19
                                                                |           |      |        |           |
    |             └── txn_snap_cap_worker_main                    +|  1620000000 |     6 |     6775 |
txnsnapworker |     2.19
                                                                |           |      |        |           |
    |               └── PostgresInitializer::InitTxnSnapWorker    +|  1620000000 |     7 |     6776 |
txnsnapworker |     2.19
                                                                |           |      |        |           |
    |                 |── PostgresInitializer::SetDatabase        +|  1600000000 |     8 |     6777 |
txnsnapworker |     2.16
                                                                |           |      |        |           |
    |                 |   └── PostgresInitializer::SetDatabaseByName+|  1600000000 |     9 |     6778 |
txnsnapworker |     2.16
```

- gs_perf_query_general([filename text])

  Description: Queries the percentage of functions called by stacks in the historical on-CPU and off-CPU flame graphs. The sum of the percentages may not be 1 due to calculation precision. To query this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** or **DBUSER** mode and unavailable in **DISABLE** mode.

  The parameters are described as follows.

Parameter	Description	Type
filename	Name of a historically collected file. This parameter is optional. The default value is the latest manually collected flame graph.	text

  Return type: SETOF record

  The following table describes the fields returned by the function.

Name	Type	Description
function_name	text	Stack function name.
percentage	float	Percentage of stack function calls.

  Example:

  Query the percentage of flame graph function calls generated in the latest manual collection.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT * FROM gs_perf_query_general();
         function_name          | percentage
--------------------------------+------------
 pthread_mutex_lock             |      28.57
 do_select                      |       7.17
 undopage::init_undo_zone_lock |       7.14
 unknown                        |       7.14
 undo::recycle_main             |       7.14
 undo::is_valid_zone_id         |       7.14
 __vdso_gettimeofday            |       7.14
 hrtimer_start_range_ns         |       7.14
 undo::get_zone_id              |       7.14
 __pthread_mutex_unlock         |       7.14
 gs_stat_encap_status_info      |       7.14
(11 rows)
```

- gs_perf_query_detail([filename text])

  Description: Queries the stack function call percentage and call stack details of the historical on-CPU and off-CPU flame graphs. The sum of the percentages may not be 1 due to calculation precision. To query this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** or **DBUSER** mode and unavailable in **DISABLE** mode.

  The parameters are described as follows.

Parameter	Description	Type
filename	Name of a historically collected file. This parameter is optional. The default value is the latest manually collected flame graph.	text

  Return type: SETOF record

  The following table describes the fields returned by the function.

Name	Type	Description
function_name	text	Stack function name.
percentage	float	Percentage of stack function calls.
function_callstack	text	Stack function call stack.
function_callstack_percentage	float	Percentage of stack function call stacks.

  Example:

  Query the percentage and details of flame graph calls generated in the last manual collection.

```
testdb3=# SELECT * FROM gs_perf_query_detail();
     function_name       | percentage
|
        function_callstack
                                                
```

```
------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------
-------------------------+------------------------------
 pthread_mutex_lock      |         30 | TrackStmtWorker->unknown->internal_thread_func->int
gauss_db_other_feature_thread_main->pthread_mutex_lock

|                        30
 stmt_flush_main         |         20 | TrackStmtWorker->unknown->internal_thread_func->int
gauss_db_other_feature_thread_main->stmt_flush_main

|                        20
 finish_task_switch      |         20 | AntiRecycler->unknown->internal_thread_func->int
gauss_db_storage_thread_main->anti_recycler_main->anti_tab_recycle_loop->__select ->[enter kernel]-
>system_call_fastpath->SyS_select->core_
sys_select->do_select->poll_schedule_timeout->schedule_hrtimeout_range-
>schedule_hrtimeout_range_clock->schedule->__schedule->finish_task_switch |                          10
 finish_task_switch      |         20 | vlogwriter->unknown->internal_thread_func->int
gauss_db_storage_thread_main->verifylogwriter_main->__select ->[enter kernel]-
>system_call_fastpath->SyS_select->core_sys_select->do_select->
poll_schedule_timeout->schedule_hrtimeout_range->schedule_hrtimeout_range_clock->schedule-
>__schedule->finish_task_switch                       |                          10
 system_call_after_swapgs |         10 | WALwriter->semctl ->[enter kernel]->system_call_after_swapgs

|                        10
 lwlock_release          |         10 | AntiRecycler->unknown->internal_thread_func->int
gauss_db_storage_thread_main->anti_recycler_main->anti_tab_recycle_loop->lwlock_release

|                        10
 __select                |         10 | AntiRecycler->unknown->internal_thread_func->int
gauss_db_storage_thread_main->anti_recycler_main->anti_tab_recycle_loop->__select

|                        10
(7 rows)
```

- gs_perf_report()

  In the current version, the collection logic defaults to flushing to disk. Manual flushing is no longer required. It displays the address of the on-CPU file that is manually queried last time. To query this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** or **DBUSER** mode and unavailable in **DISABLE** mode.

  Parameter: nan.

  Return type: text.

  Example:

  Generate a flame graph file.

```
testdb3=# SELECT * FROM gs_perf_report();
                gs_perf_report
-------------------------------------------------
 flamegraph-manual-oncpu-2025-01-24_140232.html.gz
(1 row)
```

- gs_perf_clean()

  Description: The collection logic defaults to flushing to disk in the current version. After collection, the memory context is no longer occupied. Therefore, you do not need to manually clear the memory data. In the current version, a message is displayed, indicating that the function is deprecated. In later versions, the function is directly removed. To query this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** or **DBUSER** mode and unavailable in **DISABLE** mode.

  Parameter: nan.

  Return type: text.

  Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Clear data generated by perf.

```
gaussdb=# SELECT * FROM gs_perf_clean();
                                                gs_perf_clean
-------------------------------------------------------------------------------------------------------------------------------
----------
 gs_perf_clean is a duplicated function. now you don't need clean perf memory. this function will be removed in the next version
(1 row)
```

- gs_perf_start_shaking_collect(scope text, unique_querys textarray, duration int, max_collect_times int, ringbuffer_size int)

  Description: Enables performance jitter monitoring and returns whether the operation is successful. After performance jitter monitoring is enabled, the existing performance jitter flame graph files are cleared. If the gs_perf_start_shaking_collect function is repeatedly executed and the previous performance jitter monitoring is not complete, only the **unique_querys** parameter is reset and the existing performance jitter flame graph file is cleared at the next time, which does not affect the behavior of gs_perf_service. When a slow SQL statement is executed and belongs to the input parameter **unique_querys**, the performance jitter flame graph is generated. If the slow SQL statement is executed on the CN, the performance jitter flame graphs of all nodes are collected. Otherwise, the performance jitter flame graph of the local node is collected. If the performance jitter monitoring fails to be enabled on a node, the system attempts to disable the performance jitter monitoring on the node. To query this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** mode and unavailable in **DBUSER** or **DISABLE** mode.

  Table 1-245 gs_perf_start_shaking_collect parameters

Parameter	Type	Description	Range
scope	text	**local** indicates that the monitoring of the local node is started. **all** indicates that the performance jitter monitoring of all nodes is enabled. **all** can be used only on distributed CNs. This parameter is required. If it is set to NULL, the value is **local**.	**local** or **all**.
unique_querys	textarray	List of monitored unique SQL IDs. If a slow SQL statement is in the list, the performance jitter exists. This parameter is required. If it is set to NULL, the value is an empty list.	The list length ranges from 0 to 32.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description	Range
duration	int	Maximum continuous monitoring duration of performance jitter, in minutes. This parameter is required. If it is set to NULL, the value is **60**.	5-6000
max_collect_times	int	Maximum number of performance jitter snapshots that can be saved from a performance jitter flame graph. If the number of snapshots exceeds the value, the perf data collection stops. This parameter is required. If it is set to NULL, the value is **3**.	1-10
ringbuffer_size	int	Size of the ring buffer for storing performance perf, in MB. This parameter is required. If it is set to NULL, the value is **8**.	1-32

The following table describes return values.

Table 1-246 Return values of gs_perf_start_shaking_collect

Name	Type	Description
nodename	text	Node name.
message	text	If the return value is "Successfully entered CPU_SHAKE mode.", the performance jitter monitoring is enabled successfully. If the return value is "Current is in CPU_SHAKE mode.", the monitoring is enabled repeatedly. If the return value is "Unexpected Result: start_shaking_collect error, try to close shaking collect success. Details: *Detailed error information*", the monitoring fails to be enabled on the node and is successfully stopped. If the return value is "Unexpected Result: start_shaking_collect error, try to close shaking collect failed. Details: *Detailed error information*", the monitoring fails to be enabled on the node and fails to be stopped. This problem usually occurs when the node is disconnected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

Enable performance jitter monitoring for all nodes. Set the unique SQL ID list to be monitored to {4124148646,2053082188}, monitoring duration to 60 minutes, maximum number of performance jitter snapshots that can be saved to 4, and ring buffer size to 8 MB.

```
gaussdb=# SELECT * FROM gs_perf_start_shaking_collect('all','{4124148646,2053082188}',60,4,8);
 nodename |                message
----------+---------------------------------------
 cn_5001  | Current is in CPU_SHAKE mode.
 cn_5002  | Current is in CPU_SHAKE mode.
 cn_5003  | Current is in CPU_SHAKE mode.
 dn_6001  | Successfully entered CPU_SHAKE mode.
 dn_6004  | Successfully entered CPU_SHAKE mode.
 dn_6007  | Successfully entered CPU_SHAKE mode.
 dn_6002  | Current is in CPU_SHAKE mode.
 dn_6003  | Current is in CPU_SHAKE mode.
 dn_6005  | Current is in CPU_SHAKE mode.
 dn_6006  | Current is in CPU_SHAKE mode.
 dn_6009  | Current is in CPU_SHAKE mode.
 dn_6008  | Current is in CPU_SHAKE mode.
(12 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- gs_perf_start_shaking_collect(), gs_perf_do_shaking_collect(), gs_perf_close_shaking_collect(), and gs_perf_get_shaking_collect_status() can be executed only when gs_perf_service is running properly. In addition, gs_perf_service needs to apply for a ring buffer during collection. The size of the buffer is controlled by the number of CPU cores and the input parameter **ringbuffer_size** of gs_perf_start_shaking_collect().
- If you manually stop the execution of gs_perf_start_shaking_collect(), gs_perf_do_shaking_collect(), gs_perf_close_shaking_collect(), or gs_perf_get_shaking_collect_status() or the message return times out, only the frontend SQL statement is interrupted, and the perf_service process does not stop the corresponding task. During the execution of gs_perf_do_shaking_collect(), you can run **sudo kill -9 perf_service** *PID* to stop the perf_service process and forcibly interrupt the background task.
- Enabling and disabling performance jitter monitoring and saving performance jitter flame graphs are performed by physical machine. Therefore, when you perform these operations on a node, the monitoring status of other nodes on the same physical machine is affected. For example, after performance jitter monitoring is enabled on a node, other nodes on the same host also enter the monitoring status and the monitoring cannot be enabled again.
- When **scope** is set to **all**, the implementation of enabling and disabling performance jitter monitoring and saving performance jitter flame graphs is to traverse all nodes in sequence and perform the corresponding enabling, disabling, and saving operations on each node. Therefore, only one node on the same host returns a success message, and other nodes return failure messages indicating repeated execution.
- After a node in the cluster is faulty, if the system function is called and **scope** is set to **all**, corresponding operations are performed only on normal nodes, which are not affected by the faulty node. In the returned result, the **node_name** column of the faulty node is *node_host*:***node_port***.
- When a monitored SQL statement triggers a slow SQL statement and performance jitter data of all nodes are collected, the system traverses all nodes in sequence and collects the performance jitter data of the host where the nodes are located. The traversal sequence is not fixed. This ensures that the performance jitter data of one node on each physical machine is collected.
- After some shards are upgraded, you can run the new performance jitter flame graph function on them. The returned results of the upgraded shards are returned normally.
- After the collection process gs_perf_service is restarted, the performance jitter monitoring is disabled. You need to run the command again if you want to enable the performance jitter monitoring.
- After the gs_perf_start_shaking_collect() function is used to start performance jitter monitoring, the database takes about 1 second to initialize flame graph resources and then collects the performance jitter flame graph file.

- gs_perf_do_shaking_collect(scope text)

  Description: Flushes the multi-core and multi-thread flame graphs for performance jitter to disks, that is, saves the performance jitter snapshot and returns the result of whether the snapshot is saved successfully. This function depends on the gs_perf_start_shaking_collect() API. The performance jitter snapshot can be flushed to disks only after performance jitter monitoring is enabled. The minimum interval between two flushes is 1 minute. To prevent the background collection process from being hung, when the performance jitter flame graph is generated on a single node, if the background collection process gs_perf_service is not complete within 10 minutes or gs_perf_service exits abnormally during performance jitter collection, a timeout message is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

returned. To execute this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** mode and unavailable in **DBUSER** or **DISABLE** mode.

Table 1-247 gs_perf_do_shaking_collect parameters

Parameter	Type	Description	Range
scope	text	**local** indicates that the perf data of the local node is flushed to disks. **all** indicates that the perf data of all nodes is flushed to disks. **all** can be used only on distributed CNs. This parameter is required. If it is set to NULL, the value is **local**.	**all** or **local**.

The following table describes return values.

Table 1-248 Return values of gs_perf_do_shaking_collect

Name	Type	Description
nodename	text	Node name.
message	text	The return value "Successfully collect perf detail data ....." indicates that the performance jitter snapshot is saved successfully. The return value "Current is not in CPU_SHAKE mode." indicates that the monitoring status is not enabled, and the performance jitter snapshot fails to be saved. The return value "Skipping collection this time....." indicates that the performance jitter snapshot is repeatedly saved within 1 minute. The return value "Unexpected Result: do_shaking_collect error. Details: *Detailed error information*" indicates that the execution fails.

Example:

Flush the performance jitter snapshots to disks on all nodes. The saving success, snapshot path, and flushing time are returned.

```
gaussdb=# SELECT * FROM gs_perf_do_shaking_collect('all');
 nodename |
message

----------
+----------------------------------------------------------------------------------------------------------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-------------
------------------------------------------------------------
 cn_5001  | Skipping collection this time. Reason: The data has already been collected just now.
 cn_5002  | Skipping collection this time. Reason: The data has already been collected just now.
 cn_5003  | Skipping collection this time. Reason: The data has already been collected just now.
 dn_6001  | Successfully collect perf detail data. The file path is: $GAUSSLOG/gs_flamegraph/dn_6001/
shake_detail-2025-11-27_173849.lz4, collect time is 2025-11-27_173849
 dn_6004  | Successfully collect perf detail data. The file path is: $GAUSSLOG/gs_flamegraph/dn_6004/
shake_detail-2025-11-27_173850.lz4, collect time is 2025-11-27_173850
 dn_6007  | Skipping collection this time. Reason: The data has already been collected just now.
 dn_6003  | Successfully collect perf detail data. The file path is: $GAUSSLOG/gs_flamegraph/dn_6003/
shake_detail-2025-11-27_173850.lz4, collect time is 2025-11-27_173850
 dn_6002  | Skipping collection this time. Reason: The data has already been collected just now.
 dn_6005  | Skipping collection this time. Reason: The data has already been collected just now.
 dn_6006  | Skipping collection this time. Reason: The data has already been collected just now.
 dn_6009  | Skipping collection this time. Reason: The data has already been collected just now.
 dn_6008  | Skipping collection this time. Reason: The data has already been collected just now.
(12 rows)
```

- gs_perf_close_shaking_collect(scope text)

  Description: Disables performance jitter monitoring and returns whether the operation is successful. This function depends on the gs_perf_start_shaking_collect() API. The monitoring must be started before being disabled. Otherwise, a message is returned, indicating that the monitoring is not in progress. To query this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** mode and unavailable in **DBUSER** or **DISABLE** mode.

  Table 1-249 gs_perf_close_shaking_collect parameters

Parameter	Type	Description	Range
scope	text	**local** indicates that the monitoring of the current node is stopped. **all** indicates that the monitoring of all nodes is stopped. **all** can be used only on distributed CNs. This parameter is required. If it is set to NULL, the value is **local**.	**local** or **all**.

  The following table describes return values.

  Table 1-250 Return values of gs_perf_close_shaking_collect

Name	Type	Description
nodename	text	Node name.

Name	Type	Description
message	text	The return value "Successfully exit CPU_SHAKE mode." indicates that the performance jitter monitoring is disabled successfully. The return value "Current is not in CPU_SHAKE mode." indicates that the monitoring status is not enabled and fails to be disabled. The return value "Unexpected Result: close_shaking_collect error. Detail: *Detailed error information*" indicates that the disabling fails.

Example:

```
gaussdb=# SELECT * FROM gs_perf_close_shaking_collect('all');
 nodename |              message
----------+-----------------------------------
 cn_5001  | Current is not in CPU_SHAKE mode.
 cn_5002  | Current is not in CPU_SHAKE mode.
 cn_5003  | Current is not in CPU_SHAKE mode.
 dn_6001  | Successfully exit CPU_SHAKE mode.
 dn_6004  | Current is not in CPU_SHAKE mode.
 dn_6007  | Current is not in CPU_SHAKE mode.
 dn_6002  | Current is not in CPU_SHAKE mode.
 dn_6003  | Current is not in CPU_SHAKE mode.
 dn_6005  | Successfully exit CPU_SHAKE mode.
 dn_6006  | Current is not in CPU_SHAKE mode.
 dn_6009  | Successfully exit CPU_SHAKE mode.
 dn_6008  | Current is not in CPU_SHAKE mode.
(12 rows)
```

- gs_perf_get_shaking_collect_result(scope text)

  Description: Views the flame graph collection history file of the latest performance jitter monitoring. Each time performance jitter monitoring is enabled, the performance jitter snapshot collection history is cleared. Only results of normal nodes are returned. To query this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** mode and unavailable in **DBUSER** or **DISABLE** mode.

  Table 1-251 gs_perf_get_shaking_collect_result parameters

Parameter	Type	Description	Range
scope	text	**local** indicates that the performance jitter flame graph snapshot history of the current node is queried. **all** indicates that the performance jitter flame graph snapshot history of all nodes is queried. **all** can be used only on distributed CNs. This parameter is required. If it is set to NULL, the value is **local**.	**local** or **all**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The following table describes return values.

Table 1-252 Return values of gs_perf_get_shaking_collect_result

Name	Type	Description
nodename	text	Node name.
num	text	Collection sequence number.
filename	text	Name of the performance jitter snapshot file.
shake_time	text	Time when the performance jitter snapshot is flushed to disks.

Example:

Views the flame graph collection history of the latest performance jitter monitoring.

```
gaussdb=# SELECT * FROM gs_perf_get_shaking_collect_result('all');
 nodename | num |               filename               |    shake_time
----------+-----+--------------------------------------+------------------
 dn_6001  | 1   | shake_detail-2025-11-28_160835.lz4 | 2025-11-28_160835
 dn_6004  | 1   | shake_detail-2025-11-28_160835.lz4 | 2025-11-28_160835
 dn_6003  | 1   | shake_detail-2025-11-28_160835.lz4 | 2025-11-28_160835
(3 rows)
```

- gs_perf_get_shaking_collect_status(scope text)

  Description: Views the performance jitter monitoring status of a node. To query this function, you must have the SYSADMIN or MONADMIN permission. This function is available in **ROOT** mode and unavailable in **DBUSER** or **DISABLE** mode.

  Table 1-253 gs_perf_get_shaking_collect_status parameters

Parameter	Type	Description	Range
scope	text	**local** indicates that the collection status of the host where the current node is located is queried. **all** indicates that the collection status of the hosts where all nodes are located is queried. **all** can be used only on distributed CNs. This parameter is required. If you set it to **NULL**, the value is **all** when the execution node is a CN. Otherwise, the value is **local**.	**local** or **all**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-254 Return values of gs_perf_get_shaking_collect_status

Name	Type	Description
nodename	text	Node name.
status	text	Specifies whether performance jitter monitoring has been enabled on the host where the node is located. • **NOT IN SHAKE MODE**: The monitoring is not enabled. • **IN SHAKE MODE**: The monitoring is enabled. • **NO PERF SERVICE**: The collection process does not respond. • **Unexpected Result: get_shaking_connect_status error. Detail:** *Detailed error information*: The query fails.

Example:

Check whether performance jitter monitoring is enabled on the host where the node is located.

```
gaussdb=# SELECT * FROM gs_perf_get_shaking_collect_status('all');
 nodename |    status
----------+---------------
 cn_5001  | IN SHAKE MODE
 cn_5002  | IN SHAKE MODE
 cn_5003  | IN SHAKE MODE
 dn_6001  | IN SHAKE MODE
 dn_6004  | IN SHAKE MODE
 dn_6007  | IN SHAKE MODE
 dn_6002  | IN SHAKE MODE
 dn_6003  | IN SHAKE MODE
 dn_6005  | IN SHAKE MODE
 dn_6006  | IN SHAKE MODE
 dn_6009  | IN SHAKE MODE
 dn_6008  | IN SHAKE MODE
(12 rows)
```

- gs_tpworker_execstmt_stat()

  Description: Displays the runtime information of a statement. If the SYSADMIN or MONADMIN user runs the statement, the information about all the statements that are being executed is displayed. Common users can query only the information about the SQL statements executed by themselves.

  Return type: SETOF record

Name	Type	Description
db_oid	oid	OID of the database that the user session connects to in the backend.
db_name	name	Name of the database that the user session connects to in the backend.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
threadpool_worker	varchar	NUMA group to which a thread belongs and thread ID. The format is *numagroup_threadid*.
thread_id	bigint	Thread ID.
session_id	bigint	Session ID.
query_id	bigint	ID of the SQL statement that is being executed.
query_text	text	Content of the SQL statement that is being executed.
unique_sql_id	bigint	Unique ID generated by the SQL statement.
client_hostname	text	Host name of the connected client, as reported by a reverse DNS lookup of **client_addr**. This column will be non-null only for IP connections and only when **log_hostname** is enabled.
client_app_name	text	Name of the client app.
stmt_slow_time_threshold	int	Preset timeout interval for marking an SQL statement as a slow SQL statement, in milliseconds.
stmt_start_time	timestamp with time zone	Time when the statement starts to be executed for a stored procedure, function, or package, the first query time is queried and does not change with the running of statements in the stored procedure.
stmt_elapse_time	bigint	Time elapsed since the query starts.
stmt_control_status	varchar	Current statement status. • **Waiting**: The session access is successful but it is not executed by the thread. • **Running**: The current statement is executed properly. • **Control**: The current statement enters the resource control phase.
stmt_control_rule	text	Slow SQL control rule corresponding to the current language.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
stmt_control_iostat	text	IOPS value and maximum IOPS of the current statement. The format is *curVal*|*maxVal*.
stmt_control_memstat	text	This field is reserved and is not supported currently.
stmt_control_cpustat	text	This field is reserved and is not supported currently.
stmt_control_netstat	text	This field is reserved and is not supported currently.

- gs_tpworker_execslot_stat()

  Description: Displays the thread running information. If the SYSADMIN or MONADMIN user runs the command, information about all threads is displayed. Common users can query only information about the threads where the SQL statements executed by themselves are located.

  Return type: SETOF record

Name	Type	Description
numagroup	int	NUMA group to which the current thread belongs.
worker_id	int	Thread ID of the current thread.
worker_bind_type	text	Thread binding mode. The value can be **numabind**, **cpubind**, **allbind**, or **nobind**.
worker_cpu_affinity	text	Affinity between threads and CPU cores, that is, the range of CPU cores that can be scheduled by threads.
worker_status	varchar	Current thread status. • **Waiting**: The session access is successful but it is not executed by the thread. • **Running**: The current statement is executed properly. • **Control**: The current statement enters the resource control phase.
served_query_id	bigint	ID of the SQL statement that is being executed.
served_query_text	text	Content of the SQL statement that is being executed.

- gs_session_all_settings(sessionid bigint)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Queries the full GUC parameter settings of the session corresponding to the session ID on the local node. To execute this function, you must have the SYSADMIN or MONADMIN permission.

Input parameter description: **sessionid** indicates the session ID.

Return type: SETOF record

The following table describes return fields.

Name	Type	Description
name	text	Parameter name.
setting	text	Current parameter value.
unit	text	Implicit unit of a parameter.

Example:

```
gaussdb=# CREATE USER testuser WITH PASSWORD '********' sysadmin;
gaussdb=# SET SESSION AUTHORIZATION testuser PASSWORD '********';
gaussdb=> SELECT sessionid FROM pg_stat_activity WHERE usename = 'testuser';
 sessionid
-----------
    788861
(1 row)

gaussdb=> SELECT * FROM gs_session_all_settings(788861) WHERE name = 'work_mem';
   name   | setting | unit
----------+---------+------
 work_mem | 131072  | kB
(1 row)

gaussdb=> RESET SESSION AUTHORIZATION;
gaussdb=# DROP USER testuser;
```

- gs_session_all_settings()

  Description: Queries full GUC parameter settings of all sessions on the local node. To execute this function, you must have the SYSADMIN or MONADMIN permission.

  Return type: SETOF record

Name	Type	Description
sessionid	bigint	Session ID.
pid	bigint	Backend thread ID.
name	text	Parameter name.
setting	text	Current parameter value.
unit	text	Implicit unit of a parameter.

  Example:

```
gaussdb=# SELECT * FROM gs_session_all_settings() WHERE name = 'work_mem';
   sessionid    |       pid       |   name   | setting | unit
----------------+-----------------+----------+---------+------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 140550214145792 |          96974 | work_mem | 65536   | kB
 140550214145792 |          96971 | work_mem | 65536   | kB
 140549731735296 | 140549731735296 | work_mem | 65536   | kB
 140549764413184 | 140549764413184 | work_mem | 65536   | kB
(4 rows)
```

- gs_local_wal_preparse_statistics()

  Description: Queries the latest startup of the log pre-parsing thread on the local node as well as the pre-parsing logs. Only the user with the SYSADMIN permission can execute this function. The log pre-parsing thread is typically started by the CM component when the HA replication link of the standby DN is disconnected. If the log pre-parsing thread has been started, you can call this function to query the latest pre-parsing status. If the thread has not been started, this function returns the default value. In the DR scenario, the DN and CN of the standby cluster automatically start the log pre-parsing thread. Note that the CN only starts the log pre-parsing thread in the standby cluster.

  Return type: SETOF record

Name	Type	Description
preparser_term	text	Maximum term value obtained from the latest pre-parsing log.
preparser_start_time	timestamptz	Time when the latest pre-parsing is started.
preparser_end_time	timestamptz	End time of the latest pre-parsing.
preparser_start_location	text	Start position of the latest pre-parsing log.
preparser_end_location	text	End position of the latest pre-parsing log.
preparser_total_bytes	int8	Number of latest pre-parsed logs, in bytes.
preparser_speed	int8	Latest pre-parsing speed, in bytes/ms.
is_valid	bool	Specifies whether the latest pre-parsing result can be used for leader selection.

The following is an example of the query result after the log pre-parsing thread is started:

```
gaussdb=# SELECT * FROM gs_local_wal_preparse_statistics();
 preparser_term |     preparser_start_time      |      preparser_end_time       | preparser_start_location |
preparser_end_location | preparser_total_bytes | preparser_speed | is_valid
----------------+-------------------------------+-------------------------------+--------------------------
+------------------------+-----------------------+-----------------+----------
 3107           | 2023-02-01 17:04:23.367946+08 | 2023-02-01 17:04:25.354434+08 | 00000003/
C3EEA660       | 00000004/0BE60738      |            1207394520 |      1207394520 | f
(1 row)
```

The following is an example of the query result when the log pre-parsing thread has not been started:

```
gaussdb=# SELECT * FROM gs_local_wal_preparse_statistics();
 preparser_term |  preparser_start_time  |   preparser_end_time   | preparser_start_location |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
preparser_end_location | preparser_total_bytes | preparser_speed | is_valid
-----------------+------------------------+------------------------+--------------------------
+-----------------------+----------------------+-----------------+----------
 0               | 2000-01-01 08:00:00+08 | 2000-01-01 08:00:00+08 | 00000000/00000000        |
00000000/00000000      |                    0 |               0 | f
(1 row)
```

- gs_hot_standby_space_info()

  Description: Queries the total number and total size of files in the **standby_read/base_page**, **standby_read/block_info_meta** and **standby_read/lsn_info_meta** folders.

  Return type: SETOF record

Name	Type	Description
base_page_file_num	xid	Total number of base_page_files.
base_page_total_size	xid	Total size of the base_page_files.
lsn_info_meta_file_num	xid	Total number of lsn_info_meta_files.
lsn_info_meta_total_size	xid	Total size of the lsn_info_meta_files.
block_info_meta_file_num	xid	Total number of block_info_meta_files.
block_info_meta_total_size	xid	Total size of the block_info_meta_files.

  Example:

```
gaussdb=# SELECT * FROM gs_hot_standby_space_info();
 base_page_file_num | base_page_total_size | lsn_info_meta_file_num | lsn_info_meta_total_size |
block_info_meta_file_num | block_info_meta_total_size
--------------------+----------------------+------------------------+--------------------------
+--------------------------+---------------------------
                 6 |               163840 |                      6 |                     3136 |                       16
|                   147456
(1 row)
```

- exrto_file_read_stat()

  Description: Queries the number of disk access times and total access latency of new **base page files**, **lsn info meta files**, and **block info meta files** read on a standby node. Only a CN of a standby cluster for DR or a standby DN can be connected for query. In other cases, the query result is **0**.

  Return type: SETOF record

Name	Type	Description
lsn_info_page_disk_read_counter	int8	Number of disk access times of **lsn info meta files**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
lsn_info_page_disk_read_dur	int8	Total latency of **lsn info meta file** access to disks.
blk_info_meta_disk_read_counter	int8	Number of disk access times of **block info meta files**.
blk_info_meta_disk_read_dur	int8	Total latency of **block info meta file** access to disks.
base_page_read_disk_counter	int8	Number of disk access times of **base page files**.
base_page_read_disk_dur	int8	Total latency of **base page file** access to disks.

Example:

```
gaussdb=# SELECT * FROM exrto_file_read_stat();
 lsn_info_page_disk_read_counter | lsn_info_page_disk_read_dur | blk_info_meta_disk_read_counter |
blk_info_meta_disk_read_dur | base_page_read_disk_counter | base_page_read_disk_dur
---------------------------------+-----------------------------+---------------------------------
+-----------------------------+-----------------------------+-------------------------
                          14987 |                       92313 |                           23879 |                      129811
|                           0 |                           0
(1 row)
```

- gs_exrto_recycle_info()

  Description: Queries the resource recycling location, including the recycling LSN of each thread, global recycling LSN, and the earliest snapshot LSN of a query thread. Only a CN of a standby cluster for DR or a standby DN can be connected for query. In other cases, the query result is **0**.

  Return type: SETOF record

Name	Type	Description
page_redo_worker_thread_id	text	Reclamation LSN location of redo thread. **thread_id** indicates the redo thread ID.
global_recycle_lsn	text	LSN of global reclamation location.
exrto_snapshot_oldest_lsn	text	Earliest snapshot LSN of a query thread.

  Example:

```
gaussdb=# SELECT * FROM gs_exrto_recycle_info();
            thread_id             | recycle_lsn
----------------------------------+-------------
 page_redo_worker_140148895381248 | 0/7B4552E0
 page_redo_worker_140148872312576 | 0/7B4535B8
 global_recycle_lsn               | 0/7B4535B8
 exrto_snapshot_oldest_lsn        | 0/8488E6D0
(4 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_stat_get_db_conflict_all(oid)

  Description: Queries the number of sent replay conflict signals of different types.

  Parameter: The input parameter is the OID of the database.

  Return type: SETOF record

Name	Type	Description
conflict_all	int8	Number of sent replay conflict signals.
conflict_tablespace	int8	Number of sent replay conflict signals of the tablespace type.
conflict_lock	int8	Number of sent replay conflict signals of the **lock** type.
conflict_snapshot	int8	Number of sent replay conflict signals of the snapshot type.
conflict_bufferpin	int8	Number of sent replay conflict signals of the bufferpin type.
conflict_startup_deadlock	int8	Number of sent replay conflict signals of the **startup_deadlock** type.
conflict_truncate	int8	Number of sent replay conflict signals of the truncate type.
conflict_standby_query_timeout	int8	Number of sent replay conflict signals of the standby_query_timeout type.
conflict_force_recycle	int8	Number of sent replay conflict signals of the force_recycle type.

  Example:

```
gaussdb=# SELECT * FROM gs_stat_get_db_conflict_all(12738);
 conflict_all | conflict_tablespace | conflict_lock | conflict_snapshot | conflict_bufferpin |
conflict_startup_deadlock | conflict_truncate | conflict_standby_query_timeout | conflict_force_recycle
--------------+---------------------+---------------+-------------------+--------------------
+---------------------------+-------------------+-------------------------------+------------------------
            0 |                 0 |             0 |                 0 |                 0 |                         0 |                 0
|                         0 |                   0
(1 row)
```

- gs_redo_stat_info([int])

  Description: Queries statistics about WAL replay, including the cache hit ratio of the replayer thread, the number of files deleted by replaying DDL logs, the physical reads generated when the replayer thread reads data pages, and the wait event information when the replayer thread reads WALs. The query must be executed by connecting to the standby DN.

  Return type: SETOF record

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Name	Type	Description
Input parameter	operation	int	Optional. If no value is entered, statistics are queried. • **1**: queries statistics. • **2**: resets cache hit ratio statistics.
Output parameter	buffer_hit_rate	float8	Cache hit ratio of the replayer thread.
Output parameter	ddl_unlink_nrels_count	int8	Number of files deleted by replaying DDL logs.
Output parameter	read_buffer_io_counter	int8	Number of times that wait events for physical reads are triggered when the replayer thread reads data pages.
Output parameter	read_buffer_io_total_dur	int8	Total duration of wait events for physical reads when the replayer thread reads data pages.
Output parameter	read_buffer_io_avg_dur	int8	Average duration of wait events for physical reads when the replayer thread reads data pages.
Output parameter	read_buffer_io_min_dur	int8	Minimum duration of wait events for physical reads when the replayer thread reads data pages.
Output parameter	read_buffer_io_max_dur	int8	Maximum duration of wait events for physical reads when the replayer thread reads data pages.
Output parameter	read_wal_buf_counter	int8	Number of times that wait events are triggered when the replayer thread reads WALs.
Output parameter	read_wal_buf_total_dur	int8	Total duration of wait events when the replayer thread reads WALs.
Output parameter	read_wal_buf_avg_dur	int8	Average duration of wait events when the replayer thread reads WALs.
Output parameter	read_wal_buf_min_dur	int8	Minimum duration of wait events when the replayer thread reads WALs.
Output parameter	read_wal_buf_max_dur	int8	Maximum duration of wait events when the replayer thread reads WALs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT * FROM gs_redo_stat_info();
-[ RECORD 1 ]------------+----------
buffer_hit_rate          | 70.5707
ddl_unlink_nrels_count   | 3
read_buffer_io_counter   | 1732
read_buffer_io_total_dur | 2850806
read_buffer_io_avg_dur   | 1645
read_buffer_io_min_dur   | 3
read_buffer_io_max_dur   | 981639
read_wal_buf_counter     | 9779
read_wal_buf_total_dur   | 193612470
read_wal_buf_avg_dur     | 19798
read_wal_buf_min_dur     | 3
read_wal_buf_max_dur     | 1914777
```

- gs_recovery_conflict_waitevent_info()

  Description: Queries wait event information about the function that processes redo conflicts. You need to connect to a CN of a standby cluster for DR or a standby DN for query.

  Return type: SETOF record

Name	Type	Description
conflict_lock_counter	int8	Number of times that lock redo conflicts are triggered.
conflict_lock_total_dur	int8	Total duration for processing lock redo conflicts.
conflict_lock_avg_dur	int8	Average duration for processing lock redo conflicts.
conflict_lock_min_dur	int8	Minimum duration for processing lock redo conflicts.
conflict_lock_max_dur	int8	Maximum duration for processing lock redo conflicts.
conflict_snapshot_counter	int8	Number of times that snapshot redo conflicts are triggered.
conflict_snapshot_total_dur	int8	Total duration for processing snapshot redo conflicts.
conflict_snapshot_avg_dur	int8	Average duration for processing snapshot redo conflicts.
conflict_snapshot_min_dur	int8	Minimum duration for processing snapshot redo conflicts.
conflict_snapshot_max_dur	int8	Maximum duration for processing snapshot redo conflicts.
conflict_tablespace_counter	int8	Number of times that tablespace redo conflicts are triggered.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
conflict_tablespace_total_dur	int8	Total duration for processing tablespace redo conflicts.
conflict_tablespace_avg_dur	int8	Average duration for processing tablespace redo conflicts.
conflict_tablespace_min_dur	int8	Minimum duration for processing tablespace redo conflicts.
conflict_tablespace_max_dur	int8	Maximum duration for processing tablespace redo conflicts.
conflict_database_counter	int8	Number of times that database redo conflicts are triggered.
conflict_database_total_dur	int8	Total duration for processing database redo conflicts.
conflict_database_avg_dur	int8	Average duration for processing database redo conflicts.
conflict_database_min_dur	int8	Minimum duration for processing database redo conflicts.
conflict_database_max_dur	int8	Maximum duration for processing database redo conflicts.
conflict_truncate_counter	int8	Number of times that TRUNCATE redo conflicts are triggered.
conflict_truncate_total_dur	int8	Total duration for processing TRUNCATE redo conflicts.
conflict_truncate_avg_dur	int8	Average duration for processing TRUNCATE redo conflicts.
conflict_truncate_min_dur	int8	Minimum duration for processing TRUNCATE redo conflicts.
conflict_truncate_max_dur	int8	Maximum duration for processing TRUNCATE redo conflicts.
conflict_standby_query_timeout_counter	int8	Number of times that standby_query_timeout redo conflicts are triggered.
conflict_standby_query_timeout_total_dur	int8	Total duration for processing standby_query_timeout redo conflicts.
conflict_standby_query_timeout_avg_dur	int8	Average duration for processing standby_query_timeout redo conflicts.
conflict_standby_query_timeout_min_dur	int8	Minimum duration for processing standby_query_timeout redo conflicts.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
conflict_standby_query_timeout_max_dur	int8	Maximum duration for processing standby_query_timeout redo conflicts.
conflict_force_recycle_counter	int8	Number of times that force_recycle redo conflicts are triggered.
conflict_force_recycle_total_dur	int8	Total duration for processing force_recycle redo conflicts.
conflict_force_recycle_avg_dur	int8	Average duration for processing force_recycle redo conflicts.
conflict_force_recycle_min_dur	int8	Minimum duration for processing force_recycle redo conflicts.
conflict_force_recycle_max_dur	int8	Maximum duration for processing force_recycle redo conflicts.
conflict_bufferpin_counter	int8	Number of times that buffer pin redo conflicts are triggered.
conflict_bufferpin_total_dur	int8	Total duration for processing buffer pin redo conflicts.
conflict_bufferpin_avg_dur	int8	Average duration for processing buffer pin redo conflicts.
conflict_bufferpin_min_dur	int8	Minimum duration for processing buffer pin redo conflicts.
conflict_bufferpin_max_dur	int8	Maximum duration for processing buffer pin redo conflicts.

Example:

```
gaussdb=# SELECT * FROM gs_recovery_conflict_waitevent_info();
-[ RECORD 1 ]----------------------------+------
conflict_lock_counter                   | 0
conflict_lock_total_dur                 | 0
conflict_lock_avg_dur                   | 0
conflict_lock_min_dur                   | 0
conflict_lock_max_dur                   | 0
conflict_snapshot_counter               | 0
conflict_snapshot_total_dur             | 0
conflict_snapshot_avg_dur               | 0
conflict_snapshot_min_dur               | 0
conflict_snapshot_max_dur               | 0
conflict_tablespace_counter             | 0
conflict_tablespace_total_dur           | 0
conflict_tablespace_avg_dur             | 0
conflict_tablespace_min_dur             | 0
conflict_tablespace_max_dur             | 0
conflict_database_counter               | 0
conflict_database_total_dur             | 0
conflict_database_avg_dur               | 0
conflict_database_min_dur               | 0
conflict_database_max_dur               | 0
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
conflict_truncate_counter               | 6
conflict_truncate_total_dur             | 35872
conflict_truncate_avg_dur               | 5978
conflict_truncate_min_dur               | 5130
conflict_truncate_max_dur               | 7459
conflict_standby_query_timeout_counter  | 0
conflict_standby_query_timeout_total_dur | 0
conflict_standby_query_timeout_avg_dur  | 0
conflict_standby_query_timeout_min_dur  | 0
conflict_standby_query_timeout_max_dur  | 0
conflict_force_recycle_counter          | 0
conflict_force_recycle_total_dur        | 0
conflict_force_recycle_avg_dur          | 0
conflict_force_recycle_min_dur          | 0
conflict_force_recycle_max_dur          | 0
conflict_bufferpin_counter              | 0
conflict_bufferpin_total_dur            | 0
conflict_bufferpin_avg_dur              | 0
conflict_bufferpin_min_dur              | 0
conflict_bufferpin_max_dur              | 0
```

- gs_display_delay_ddl_info()

  Description: Views information about files that are delayed for deletion on the standby node.

  Return type: SETOF record

Name	Type	Description
type	INT4	Indicates that the deleted object is a table or database.
lsn	TEXT	Marks the location of a particular log file.
tablespace	INT4	Indicates the physical space for storing tables and indexes in a database.
database	INT4	Indicates the physical storage location of a database.
relation	INT4	Indicates the object in a database, which can be the physical location of a table, view, or index.
bucketid	INT4	Specifies the bucket to which the relationship object belongs.
opt	INT4	Indicates the attribute of a compressed table.
forknum	INT4	Specifies a suffix name for a subject name. You can find a unique physical file based on the subject name and suffix name.

  Example:

```
gaussdb=# SELECT * FROM gs_display_delay_ddl_info();
 type | lsn | tablespace | database | relation | bucketid | opt | forknum
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
------+-----+------------+----------+----------+----------+-----+---------
(0 rows)
```

- gs_plsql_statements(sessionid bigint, pkg id oid, function id oid)

  Description: Returns the memory information about a stored procedure.

  Return type: SETOF record

  The following table describes the fields returned by the function.

Name	Type	Description
pid	bigint	Background thread ID.
sessionid	bigint	Session ID.
global_sessionid	text	Global session ID.
databaseid	oid	Database ID.
pkg_id	oid	ID of the package to which the stored procedure belongs. If the stored procedure does not belong to a package, the value is **0**.
func_id	oid	Stored procedure ID.
func_signature	text	Stored procedure name (name+fn_collation +search_path).
plsql_compiled_totalsize	bigint	Total memory size of the stored procedure compilation product.
plsql_plan_totalsize	bigint	Total memory size of the plan within the stored procedure.

Example:

```
gaussdb=#
CREATE OR REPLACE PROCEDURE proc_assign AS
    a int := 0;
BEGIN
    a = a + 1;
    RAISE INFO '%', a;
END;
/
gaussdb=# CALL proc_assign();
INFO:  1
 proc_assign
-------------


(1 row)

gaussdb=# SELECT * FROM pg_catalog.gs_plsql_statements(pg_current_sessid());
    pid    | sessionid | global_sessionid | databaseid | pkg_id | func_id |
func_signature
            | plsql_compiled_totalsize | plsql_plan_totalsize
-----------+-----------+------------------+------------+--------+---------
+-------------------------------------------------------------------------
------------+--------------------------+----------------------
 536868608 |      5052 | 0:0#0            |      13108 |      0 |   16854 | name: proc_assign(), fn_collation: 0,
search_path: addUser: false, schemas: 2200 |                    8192 |                65536
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Function ID to be filtered --
gaussdb=# SELECT * FROM pg_catalog.gs_plsql_statements(NULL, NULL, 16854);
    pid    | sessionid | global_sessionid | databaseid | pkg_id | func_id |
func_signature
           | plsql_compiled_totalsize | plsql_plan_totalsize
-----------+-----------+------------------+------------+--------+---------
+-------------------------------------------------------------------------
------------+-------------------------+----------------------
 536868608 |      5052 | 0:0#0            |      13108 |      0 |   16854 | name: proc_assign(), fn_collation: 0,
search_path: addUser: false, schemas: 2200 |                    8192 |                65536
(1 row)

gaussdb=# DROP PROCEDURE proc_assign;
```

- gs_global_plsql_statements(sessionid bigint, pkg id oid, function id oid)

  Description: Returns the memory information about a stored procedure.

  Return type: SETOF record

  The following table describes the fields returned by the function.

Name	Type	Description
node_name	text	Name of the current node.
pid	bigint	Background thread ID.
sessionid	bigint	Session ID.
global_sessionid	text	Global session ID.
databaseid	oid	Database ID.
pkg_id	oid	ID of the package to which the stored procedure belongs. If the stored procedure does not belong to a package, the value is **0**.
func_id	oid	Stored procedure ID.
func_signature	text	Stored procedure name (name +fn_collation+search_path).
plsql_compiled_totalsize	bigint	Total memory size of the stored procedure compilation product.
plsql_plan_totalsize	bigint	Total memory size of the plan within the stored procedure.

Example:

```
gaussdb=#
CREATE OR REPLACE PROCEDURE proc_assign AS
    a int := 0;
BEGIN
    a = a + 1;
    RAISE INFO '%', a;
END;
/
gaussdb=# CALL proc_assign();
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
INFO:  1
 proc_assign
-------------

(1 row)

gaussdb=# SELECT * FROM pg_catalog.gs_global_plsql_statements(pg_current_sessid());
  node_name   |    pid     |    sessionid    | global_sessionid | databaseid | pkg_id | func_id
|
func_signature                                          | plsql_compiled_totalsize | plsql_plan_totalsize
--------------+------------+-----------------+------------------+------------+--------+---------
+-------------------------------------------------
-------------------------------------------------------------------+--------------------------+----------------------
 coordinator1 | 1171777280 | 139767997523712 | 0:0#0            |      14023 |      0 |   16387 | name:
proc_assign(), fn_collation: 0, search_path: addUser: false, schemas: 2200
|                  24576 |               64512
 coordinator1 | 1171777280 | 139767997523712 | 0:0#0            |      14023 |      0 |   13509 | name:
gs_global_plsql_statements(bigint,oid,oid), fn_collation: 0, search_path: addUser: false, schemas: 2200
|                 122880 |              216064
(2 rows)

gaussdb=# DROP PROCEDURE proc_assign;
```

- gs_plsql_statements_details(sessionid bigint, pkg id oid, function id oid)

  Description: Returns the memory information about the plan cache statements of a stored procedure.

  NOTE

  When the function is queried, GaussDB determines whether to allow access based on the permissions of **current_user**. If **current_user** is an initial user or a system administrator, they can query information about all sessions. If **current_user** is a common user, they can only view their own session information (**session_user** refers to the information about the current user).

  Return type: SETOF record

  The following table describes the fields returned by the function.

Name	Type	Description
pid	bigint	Background thread ID.
sessionid	bigint	Session ID.
global_sessionid	text	Global session ID.
databaseid	oid	Database ID.
pkg_id	oid	ID of the package to which the stored procedure belongs. If the stored procedure does not belong to a package, the value is **0**.
func_id	oid	Stored procedure ID.
func_signature	text	Stored procedure name (name+fn_collation +search_path).
lineno	bigint	Line number of the statement in the stored procedure.
query	text	Plan cache statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
usedsize	bigint	Memory size used by plan cache statements.
totalsize	bigint	Total memory size of plan cache statements.

Example:

```
gaussdb=#
CREATE OR REPLACE PROCEDURE proc_assign AS
    a int := 0;
BEGIN
    a = a + 1;
    RAISE INFO '%', a;
END;
/
gaussdb=# CALL proc_assign();
INFO:  1
 proc_assign
-------------


(1 row)

gaussdb=# SELECT * FROM pg_catalog.gs_plsql_statements_details(pg_current_sessid());
    pid     | sessionid | global_sessionid | databaseid | pkg_id | func_id |
func_signature
            | lineno |     query      | usedsize | totalsize
------------+-----------+------------------+------------+--------+---------
+-------------------------------------------------------------------------
------------+--------+--------------+----------+-----------
 536868608 |      5251 | 0:0#0            |      13108 |      0 |   16854 | name: proc_assign(), fn_collation: 0,
search_path: addUser: false, schemas: 2200 |      3 | SELECT a + 1 |    15064 |     25600
 536868608 |      5251 | 0:0#0            |      13108 |      0 |   16854 | name: proc_assign(), fn_collation: 0,
search_path: addUser: false, schemas: 2200 |      4 | SELECT a     |    12320 |     17408
 536868608 |      5251 | 0:0#0            |      13108 |      0 |   16854 | name: proc_assign(), fn_collation: 0,
search_path: addUser: false, schemas: 2200 |      1 | SELECT 0     |    12088 |     21504
(3 rows)

-- Function ID to be filtered --
gaussdb=# SELECT * FROM pg_catalog.gs_plsql_statements(NULL, NULL, 16854);
    pid     | sessionid | global_sessionid | databaseid | pkg_id | func_id |
func_signature
            | lineno |     query      | usedsize | totalsize
------------+-----------+------------------+------------+--------+---------
+-------------------------------------------------------------------------
------------+--------+--------------+----------+-----------
 536868608 |      5251 | 0:0#0            |      13108 |      0 |   16854 | name: proc_assign(), fn_collation: 0,
search_path: addUser: false, schemas: 2200 |      3 | SELECT a + 1 |    15064 |     25600
 536868608 |      5251 | 0:0#0            |      13108 |      0 |   16854 | name: proc_assign(), fn_collation: 0,
search_path: addUser: false, schemas: 2200 |      4 | SELECT a     |    12320 |     17408
 536868608 |      5251 | 0:0#0            |      13108 |      0 |   16854 | name: proc_assign(), fn_collation: 0,
search_path: addUser: false, schemas: 2200 |      1 | SELECT 0     |    12088 |     21504
(3 rows)

gaussdb=# DROP PROCEDURE proc_assign;
```

- gs_global_plsql_statements_details(sessionid bigint, pkg id oid, function id oid)

  Description: Returns the memory information about the plan cache statements of a stored procedure.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

When the function is queried, GaussDB determines whether to allow access based on the permissions of **current_user**. If **current_user** is an initial user or a system administrator, they can query information about all sessions. If **current_user** is a common user, they can only view their own session information (**session_user** refers to the information about the current user).

Return type: SETOF record

The following table describes the fields returned by the function.

Name	Type	Description
node_name	text	Name of the current node.
pid	bigint	Background thread ID.
sessionid	bigint	Session ID.
global_sessionid	text	Global session ID.
databaseid	oid	Database ID.
pkg_id	oid	ID of the package to which the stored procedure belongs. If the stored procedure does not belong to a package, the value is **0**.
func_id	oid	Stored procedure ID.
func_signature	text	Stored procedure name (name +fn_collation+search_path).
lineno	bigint	Line number of the statement in the stored procedure.
query	text	Plan cache statement.
usedsize	bigint	Memory size used by plan cache statements.
totalsize	bigint	Total memory size of plan cache statements.

Example:

```
gaussdb=#
CREATE OR REPLACE PROCEDURE proc_assign AS
    a int := 0;
BEGIN
    a = a + 1;
    RAISE INFO '%', a;
END;
/
gaussdb=# CALL proc_assign();
INFO:  1
 proc_assign
-------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)

gaussdb=# SELECT * FROM pg_catalog.gs_global_plsql_statements_details(pg_current_sessid());
  node_name   |    pid     |    sessionid    | global_sessionid | databaseid | pkg_id | func_id
|
    func_signature                                          | lineno
|
                                query                                                
```

```
gaussdb=# DROP PROCEDURE proc_assign;
```

Functions for Collecting Statistics in Partitioned Tables

- gs_stat_get_partition_stats(oid)
  Description: Obtains the statistics of a specific partition.
  Return type: record.
- gs_stat_get_xact_partition_stats(oid)
  Description: Obtains transaction statistics of a specific partition.
  Return type: record.
- gs_stat_get_all_partitions_stats()
  Description: Obtains the statistics of all partitions.
  Return type: SETOF record
- gs_stat_get_xact_all_partitions_stats()
  Description: Obtains transaction statistics of all partitions.
  Return type: SETOF record
- gs_statio_get_all_partitions_stats()
  Description: Obtains the I/O statistics of all partitions.
  Return type: SETOF record
  Examples of the preceding five functions

CAUTION

Statistics are reported asynchronously during execution. Based on UDP, delay and packet loss may occur during background thread processing. The following example is for reference only.

Querying out-of-transaction statistics:

```
gaussdb=# CREATE TABLE part_tab1
gaussdb-# (
gaussdb(#     a int, b int
gaussdb(# )
gaussdb-# PARTITION BY RANGE(b)
gaussdb-# (
gaussdb(#     PARTITION P1 VALUES LESS THAN(10),
gaussdb(#     PARTITION P2 VALUES LESS THAN(20),
gaussdb(#     PARTITION P3 VALUES LESS THAN(MAXVALUE)
gaussdb(# );
CREATE TABLE
gaussdb=# CREATE TABLE subpart_tab1
gaussdb-# (
gaussdb(#     month_code VARCHAR2 ( 30 ) NOT NULL ,
gaussdb(#     dept_code  VARCHAR2 ( 30 ) NOT NULL ,
gaussdb(#     user_no    VARCHAR2 ( 30 ) NOT NULL ,
gaussdb(#     sales_amt  int
gaussdb(# )
gaussdb-# PARTITION BY RANGE (month_code) SUBPARTITION BY RANGE (dept_code)
gaussdb-# (
gaussdb(#   PARTITION p_201901 VALUES LESS THAN( '201903' )
gaussdb(#   (
gaussdb(#     SUBPARTITION p_201901_a VALUES LESS THAN( '2' ),
gaussdb(#     SUBPARTITION p_201901_b VALUES LESS THAN( '3' )
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb(#   ),
gaussdb(#   PARTITION p_201902 VALUES LESS THAN( '201904' )
gaussdb(#   (
gaussdb(#     SUBPARTITION p_201902_a VALUES LESS THAN( '2' ),
gaussdb(#     SUBPARTITION p_201902_b VALUES LESS THAN( '3' )
gaussdb(#   )
gaussdb(# );
CREATE TABLE
gaussdb=# CREATE INDEX index_part_tab1 ON part_tab1(b) LOCAL
gaussdb-# (
gaussdb(# PARTITION b_index1,
gaussdb(# PARTITION b_index2,
gaussdb(# PARTITION b_index3
gaussdb(# );
CREATE INDEX
gaussdb=# CREATE INDEX idx_user_no ON subpart_tab1(user_no) LOCAL;
CREATE INDEX
gaussdb=# INSERT INTO part_tab1 VALUES(1, 1);
INSERT 0 1
gaussdb=# INSERT INTO part_tab1 VALUES(1, 11);
INSERT 0 1
gaussdb=# INSERT INTO part_tab1 VALUES(1, 21);
INSERT 0 1
gaussdb=# UPDATE part_tab1 SET a = 2 WHERE b = 1;
UPDATE 1
gaussdb=# UPDATE part_tab1 SET a = 3 WHERE b = 11;
UPDATE 1
gaussdb=# UPDATE /*+ indexscan(part_tab1) */ part_tab1 SET a = 4 WHERE b = 21;
UPDATE 1
gaussdb=# DELETE FROM part_tab1;
DELETE 3
gaussdb=# ANALYZE part_tab1;
ANALYZE
gaussdb=# VACUUM part_tab1;
VACUUM
gaussdb=# INSERT INTO subpart_tab1 VALUES('201902', '1', '1', 1);
INSERT 0 1
gaussdb=# INSERT INTO subpart_tab1 VALUES('201902', '2', '2', 1);
INSERT 0 1
gaussdb=# INSERT INTO subpart_tab1 VALUES('201903', '1', '3', 1);
INSERT 0 1
gaussdb=# INSERT INTO subpart_tab1 VALUES('201903', '2', '4', 1);
INSERT 0 1
gaussdb=# UPDATE subpart_tab1 SET sales_amt = 2 WHERE user_no='1';
UPDATE 1
gaussdb=# UPDATE subpart_tab1 SET sales_amt = 3 WHERE user_no='2';
UPDATE 1
gaussdb=# UPDATE subpart_tab1 SET sales_amt = 4 WHERE user_no='3';
UPDATE 1
gaussdb=# UPDATE /*+ indexscan(subpart_tab1) */ subpart_tab1 SET sales_amt = 5 WHERE
user_no='4';
UPDATE 1
gaussdb=# DELETE FROM subpart_tab1;
DELETE 4
gaussdb=# ANALYZE subpart_tab1;
ANALYZE
gaussdb=# VACUUM subpart_tab1;
VACUUM
gaussdb=# SELECT * FROM gs_stat_all_partitions;
 partition_oid | schemaname |   relname    | partition_name | sub_partition_name | seq_scan |
seq_tup_read | idx_scan | idx_tup_fetch | n_tup_ins | n_tup_upd | n_tup_del | n_tup_hot_upd | n_live_tup
|
 n_dead_tup |         last_vacuum          |    last_autovacuum     |         last_analyze          |
last_autoanalyze    | vacuum_count | autovacuum_count | analyze_count | autoanalyze_count
---------------+------------+--------------+----------------+--------------------+----------+--------------
+----------+---------------+-----------+-----------+-----------+---------------+------------+
------------+-------------------------------+-----------------------+-------------------------------
+-----------------------+--------------+------------------+---------------+------------------
        16964 | public     | subpart_tab1 | p_201902       | p_201902_b         |       5 |            1 |       4
```

```
|            1 |         1 |         1 |         1 |            1 |          0 |
         1 | 2023-05-15 20:36:45.293965+08 | 2000-01-01 08:00:00+08 | 2023-05-15
20:36:44.688861+08 | 2000-01-01 08:00:00+08 |            1 |             0 |            1 |             0
         16963 | public     | subpart_tab1 | p_201902       | p_201902_a         |       5 |           1 |       4
|           0 |         1 |         1 |         1 |            1 |          0 |
         1 | 2023-05-15 20:36:45.291022+08 | 2000-01-01 08:00:00+08 | 2023-05-15
20:36:44.688843+08 | 2000-01-01 08:00:00+08 |            1 |             0 |            1 |             0
         16961 | public     | subpart_tab1 | p_201901       | p_201901_b         |       5 |           1 |       4
|           0 |         1 |         1 |         1 |            1 |          0 |
         1 | 2023-05-15 20:36:45.288037+08 | 2000-01-01 08:00:00+08 | 2023-05-15
20:36:44.688829+08 | 2000-01-01 08:00:00+08 |            1 |             0 |            1 |             0
         16960 | public     | subpart_tab1 | p_201901       | p_201901_a         |       5 |           1 |       4
|           0 |         1 |         1 |         1 |            1 |          0 |
         1 | 2023-05-15 20:36:45.285311+08 | 2000-01-01 08:00:00+08 | 2023-05-15
20:36:44.688802+08 | 2000-01-01 08:00:00+08 |            1 |             0 |            1 |             0
         16954 | public     | part_tab1    | p3             |                    |       2 |           1 |       1 |           1
|           1 |         1 |         1 |            1 |          0 |
         1 | 2023-05-15 20:36:29.490636+08 | 2000-01-01 08:00:00+08 | 2023-05-15
20:36:28.540115+08 | 2000-01-01 08:00:00+08 |            1 |             0 |            1 |             0
         16953 | public     | part_tab1    | p2             |                    |       4 |           1 |       1 |           0
|           1 |         1 |         1 |            1 |          0 |
         1 | 2023-05-15 20:36:29.487914+08 | 2000-01-01 08:00:00+08 | 2023-05-15
20:36:28.540098+08 | 2000-01-01 08:00:00+08 |            1 |             0 |            1 |             0
         16952 | public     | part_tab1    | p1             |                    |       5 |           1 |       1 |           0
|           1 |         1 |         1 |            1 |          0 |
         1 | 2023-05-15 20:36:29.48536+08  | 2000-01-01 08:00:00+08 | 2023-05-15 20:36:28.540071+08
| 2000-01-01 08:00:00+08 |            1 |               0 |             1 |               0
(7 rows)

gaussdb=# SELECT * FROM gs_statio_all_partitions;
 partition_oid | schemaname |    relname     | partition_name | sub_partition_name | heap_blks_read |
heap_blks_hit | idx_blks_read | idx_blks_hit | toast_blks_read | toast_blks_hit | tidx_blks_read | t
idx_blks_hit
---------------+------------+--------------+----------------+--------------------+----------------+---------------
+---------------+--------------+-----------------+----------------+----------------+--
-------------
         16964 | public     | subpart_tab1 | p_201902       | p_201902_b         |              4 |             8
|             2 |           21 |                |                |                |

         16963 | public     | subpart_tab1 | p_201902       | p_201902_a         |              4 |             8
|             2 |           21 |                |                |                |

         16961 | public     | subpart_tab1 | p_201901       | p_201901_b         |              4 |             8
|             2 |           21 |                |                |                |

         16960 | public     | subpart_tab1 | p_201901       | p_201901_a         |              4 |             8
|             2 |           21 |                |                |                |

         16954 | public     | part_tab1    | p3             |                    |              4 |             8 |             2
|           15 |                |                |                |

         16953 | public     | part_tab1    | p2             |                    |              4 |             8 |             2
|           15 |                |                |                |

         16952 | public     | part_tab1    | p1             |                    |              4 |             8 |             2
|           15 |                |                |                |

(7 rows)

gaussdb=# SELECT * FROM gs_stat_get_partition_stats(16952);
 partition_oid | seq_scan | seq_tup_read | idx_scan | idx_tup_fetch | n_tup_ins | n_tup_upd | n_tup_del |
n_tup_hot_upd | n_live_tup | n_dead_tup |          last_vacuum          |     last_autovacuum
  |         last_analyze          |     last_autoanalyze     | vacuum_count | autovacuum_count |
analyze_count | autoanalyze_count |    last_data_changed     | heap_blks_read | heap_blks_hit |
idx_blks_re
ad | idx_blks_hit | tup_fetch | block_fetch
---------------+----------+--------------+----------+---------------+-----------+-----------+-----------
+---------------+------------+------------+-----------------------------+---------------------
--+-------------------------------+-------------------------+--------------+------------------+---------------
```

```
+-------------------+------------------------+----------------+---------------+------------
---+--------------+-----------+-------------
        16952 |         5 |            1 |         1 |            0 |         1 |         1 |         1 |            1 |          0
|          1 | 2023-05-15 20:36:29.48536+08 | 2000-01-01 08:00:00+0
8 | 2023-05-15 20:36:28.540071+08 | 2000-01-01 08:00:00+08 |             1 |               0 |             1
|                0 | 2000-01-01 08:00:00+08 |              4 |             8 |
 2 |          21 |         0 |          12
(1 row)
```

Querying statistics within a transaction

```
gaussdb=# BEGIN;
BEGIN
gaussdb=# INSERT INTO part_tab1 VALUES(1, 1);
INSERT 0 1
gaussdb=# INSERT INTO part_tab1 VALUES(1, 11);
INSERT 0 1
gaussdb=# INSERT INTO part_tab1 VALUES(1, 21);
INSERT 0 1
gaussdb=# UPDATE part_tab1 SET a = 2 WHERE b = 1;
UPDATE 1
gaussdb=# UPDATE part_tab1 SET a = 3 WHERE b = 11;
UPDATE 1
gaussdb=# UPDATE /*+ indexscan(part_tab1) */ part_tab1 SET a = 4 WHERE b = 21;
UPDATE 1
gaussdb=# DELETE FROM part_tab1;
DELETE 3
gaussdb=# INSERT INTO subpart_tab1 VALUES('201902', '1', '1', 1);
INSERT 0 1
gaussdb=# INSERT INTO subpart_tab1 VALUES('201902', '2', '2', 1);
INSERT 0 1
gaussdb=# INSERT INTO subpart_tab1 VALUES('201903', '1', '3', 1);
INSERT 0 1
gaussdb=# INSERT INTO subpart_tab1 VALUES('201903', '2', '4', 1);
INSERT 0 1
gaussdb=# UPDATE subpart_tab1 SET sales_amt = 2 WHERE user_no='1';
UPDATE 1
gaussdb=# UPDATE subpart_tab1 SET sales_amt = 3 WHERE user_no='2';
UPDATE 1
gaussdb=# UPDATE subpart_tab1 SET sales_amt = 4 WHERE user_no='3';
UPDATE 1
gaussdb=# UPDATE /*+ indexscan(subpart_tab1) */ subpart_tab1 SET sales_amt = 5 WHERE
user_no='4';
UPDATE 1
gaussdb=# DELETE FROM subpart_tab1;
DELETE 4
gaussdb=# SELECT * FROM gs_stat_xact_all_partitions;
 partition_oid | schemaname |    relname    | partition_name | sub_partition_name | seq_scan |
seq_tup_read | idx_scan | idx_tup_fetch | n_tup_ins | n_tup_upd | n_tup_del | n_tup_hot_upd
---------------+------------+---------------+----------------+--------------------+----------+--------------
+----------+---------------+-----------+-----------+-----------+---------------
        16964 | public     | subpart_tab1 | p_201902       | p_201902_b         |        4 |          4 |        1
|           2 |         1 |         1 |         1 |           1
        16963 | public     | subpart_tab1 | p_201902       | p_201902_a         |        4 |          4 |        1
|           0 |         1 |         1 |         1 |           1
        16961 | public     | subpart_tab1 | p_201901       | p_201901_b         |        4 |          4 |        1
|           0 |         1 |         1 |         1 |           1
        16960 | public     | subpart_tab1 | p_201901       | p_201901_a         |        4 |          4 |        1
|           0 |         1 |         1 |         1 |           1
        16954 | public     | part_tab1    | p3             |                    |        1 |          1 |        1 |          2
|         1 |         1 |         1 |           1
        16953 | public     | part_tab1    | p2             |                    |        3 |          2 |        0 |          0
|         1 |         1 |         1 |           1
        16952 | public     | part_tab1    | p1             |                    |        4 |          2 |        0 |          0
|         1 |         1 |         1 |           1
(7 rows)

gaussdb=# SELECT * FROM gs_stat_get_xact_partition_stats(16952);
 partition_oid | seq_scan | seq_tup_read | idx_scan | idx_tup_fetch | n_tup_ins | n_tup_upd | n_tup_del |
n_tup_hot_upd | tup_fetch
---------------+----------+--------------+----------+---------------+-----------+-----------+-----------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
+---------------+-----------
       16952 |       4 |           2 |       0 |           0 |       1 |       1 |       1 |           1 |       0
(1 row)

gaussdb=# DROP TABLE part_tab1;
gaussdb=# DROP TABLE subpart_tab1;
```

- gs_stat_get_partition_analyze_count(oid)

  Description: Specifies the number of times that a user starts analysis on a partition.

  Return type: bigint.

- gs_stat_get_partition_autoanalyze_count(oid)

  Description: Specifies the number of times that the autovacuum daemon thread starts analysis in a partition.

  Return type: bigint.

- gs_stat_get_partition_autovacuum_count(oid)

  Description: Specifies the number of times that the autovacuum daemon thread starts vacuum in a partition.

  Return type: bigint.

- gs_stat_get_partition_last_analyze_time(oid)

  Description: Specifies the last time when a partition starts to be analyzed manually or by the autovacuum thread.

  Return type: timestamptz.

- gs_stat_get_partition_last_autoanalyze_time(oid)

  Description: Specifies the time when the last analysis initiated by the autovacuum daemon thread in a partition.

  Return type: timestamptz.

- gs_stat_get_partition_last_autovacuum_time(oid)

  Description: Specifies the time of the last vacuum initiated by the autovacuum daemon thread in a partition.

  Return type: timestamptz.

- gs_stat_get_partition_last_data_changed_time(oid)

  Description: Specifies the last time of a modification in a partition, such as insert, update, delete, and truncate. Currently, this parameter is not supported.

  Return type: timestamptz.

- gs_stat_get_partition_last_vacuum_time(oid)

  Description: Specifies the most recent time when the user manually vacuumed a table or when the autovacuum thread was started to vacuum a partition.

  Return type: timestamptz.

- gs_stat_get_partition_numscans(oid)

  Description: Specifies the number of rows read by sequential scan in a partition.

  Return type: bigint.

- gs_stat_get_partition_tuples_returned(oid)

  Description: Specifies the number of rows read by sequential scan in a partition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: bigint.

- gs_stat_get_partition_tuples_fetched(oid)

  Description: Specifies the number of rows fetched by bitmap scans in a partition.

  Return type: bigint.

- gs_stat_get_partition_vacuum_count(oid)

  Description: Specifies the number of times that a user starts vacuum in a partition.

  Return type: bigint.

- gs_stat_get_xact_partition_tuples_fetched(oid)

  Description: Specifies the number of tuple rows scanned in a transaction.

  Return type: bigint.

- gs_stat_get_xact_partition_numscans(oid)

  Description: Specifies the number of sequential scans performed on a partition in the current transaction.

  Return type: bigint.

- gs_stat_get_xact_partition_tuples_returned(oid)

  Description: Specifies the number of rows read through sequential scans in a partition in the current transaction.

  Return type: bigint.

- gs_stat_get_partition_blocks_fetched(oid)

  Description: Specifies the number of disk block fetch requests for a partition.

  Return type: bigint.

- gs_stat_get_partition_blocks_hit(oid)

  Description: Specifies the number of disk block requests found in cache for a partition.

  Return type: bigint.

- pg_stat_get_partition_tuples_inserted(oid)

  Description: Specifies the number of rows inserted into the corresponding table partition.

  Return type: bigint.

- pg_stat_get_partition_tuples_updated(oid)

  Description: Specifies the number of rows that have been updated in the corresponding table partition.

  Return type: bigint.

- pg_stat_get_partition_tuples_deleted(oid)

  Description: Specifies the number of rows deleted from the corresponding table partition.

  Return type: bigint.

- pg_stat_get_partition_tuples_changed(oid)

  Description: Specifies the total number of inserted, updated, and deleted rows after a table partition was last analyzed or autoanalyzed.

  Return type: bigint.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_stat_get_partition_live_tuples(oid)

  Description: Specifies the number of live rows in a partitioned table.

  Return type: bigint.

- pg_stat_get_partition_dead_tuples(oid)

  Description: Specifies the number of dead rows in a partitioned table. It indicates the number of inactive row pointers in Ustore tables.

  Return type: bigint.

- pg_stat_get_xact_partition_tuples_inserted(oid)

  Description: Specifies the number of inserted tuples in the active sub-transactions related to a table partition.

  Return type: bigint.

- pg_stat_get_xact_partition_tuples_deleted(oid)

  Description: Specifies the number of deleted tuples in the active sub-transactions related to a table partition.

  Return type: bigint.

- pg_stat_get_xact_partition_tuples_hot_updated(oid)

  Description: Specifies the number of hot updated tuples in the active sub-transactions related to a table partition.

  Return type: bigint.

- pg_stat_get_xact_partition_tuples_updated(oid)

  Description: Specifies the number of updated tuples in the active sub-transactions related to a table partition.

  Return type: bigint.

- pg_stat_get_partition_tuples_hot_updated(oid)

  Description: Returns statistics on the number of hot updated tuples in a partition with a specified partition ID.

  Parameter: oid.

  Return type: bigint.

- gs_stat_partition_analyze_vacuum(oid)

  Description: Obtains vacuum/analyze statistics about a specific partition for cache acceleration.

  Parameter: partoid.

  Return type: record.

- gs_stat_get_parttabentries()

  Description: Obtains the vacuum/analyze statistics about partitioned tables on other nodes. This is an internal function. You are advised not to use it directly.

  Return type: record.

- gs_wlm_respool_cpu_info()

  Description: Displays the limit and usage of CPU resources in a resource pool.

  Return type: SETOF record

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
respool_name	name	Name of the resource pool.
control_group	name	Cgroup name.
cpu_affinity	name	Value of cores bound to the CPU.
cpu_usage	integer	CPU usage of a resource pool. If CPU control is not performed for the resource pool, the value of this parameter is **0**.

NOTE

If the CN and DN are deployed together, the CN and DN share the same CPU resources. Therefore, the **cpu_usage** values of the CN and DN are the same. If the CN and DN are independently deployed, the **cpu_usage** values of the CN and DN are displayed separately.

Example:

```
gaussdb=# SELECT * FROM GS_WLM_RESPOOL_CPU_INFO();
 respool_name |     control_group      | cpu_affinity | cpu_usage
--------------+------------------------+--------------+-----------
 respool_cpu_2 | respool_cpu_2:Medium | 0-95          |         78
 default_pool  | DefaultClass:Medium  | 0-32          |         65
(2 rows)
```

- gs_wlm_respool_connection_info()

  Description: Displays the limit and usage of the number of connections in ta resource pool.

  Return type: SETOF record

Name	Type	Description
respool_name	name	Name of the resource pool.
max_connections	name	Maximum number of connections to a resource pool.
curr_connections	integer	Number of existing connections in the current resource pool.

Example:

```
gaussdb=# SELECT * FROM GS_WLM_RESPOOL_CONNECTION_INFO();
 respool_name  | max_connections | curr_connections
---------------+-----------------+------------------
 respool1      |              -1 |                0
 default_pool  |              -1 |                1
(2 rows)
```

- gs_wlm_respool_memory_info()

  Description: Displays the limit and usage of memory resources in a resource pool.

  Return type: SETOF record

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
respool_name	name	Name of the resource pool.
max_dynamic_memory	integer	Maximum dynamic memory that can be used.
current_dynamic_memory	integer	Used dynamic memory.
max_shared_memory	integer	Maximum shared memory that can be used.
current_shared_memory	integer	Shared memory that has been used.
shared_memory_hits_percent	integer	This function is unavailable and is reserved for future evolution. This column is reserved and the value is **0**.

NOTE

When the dynamic memory usage of the resource pool exceeds the maximum value, the returned value of the GUC parameter **current_dynamic_memory** may be greater than the value of **max_dynamic_memory**. This is normal because no memory is allocated.

Example:

```
gaussdb=# SELECT * FROM GS_WLM_RESPOOL_MEMORY_INFO();
 respool_name   | max_dynamic_memory | current_dynamic_memory | max_shared_memory |
current_shared_memory | shared_memory_hits_percent
----------------+--------------------+------------------------+-------------------+------------------------
+----------------------------
 default_pool   | -1                 | 3383kB                 | -1                | 3848kB                 |
0
 resource_pool_a | 30720kB           | 0kB                    | -1                | 0kB
|                        0
(2 rows)
```

- gs_wlm_respool_concurrency_info()

  Description: Displays the limit and usage of concurrent resources in a resource pool.

  Return type: SETOF record

Name	Type	Description
respool_name	name	Name of the resource pool.
max_concurrency	integer	Maximum number of concurrent queries allowed by the resource pool.
running_concurrency	integer	Number of concurrent tasks that are being executed in the current resource pool.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
waiting_concurrency	integer	Number of concurrent tasks that are waiting in the current resource pool. If the number of concurrent requests is not limited in the resource pool, the value of this parameter is **0**.

Example:

```
gaussdb=# SELECT * FROM GS_WLM_RESPOOL_CONCURRENCY_INFO();
 respool_name   | max_concurrency | running_concurrency | waiting_concurrency
----------------+-----------------+---------------------+---------------------
 default_pool   |              -1 |                   1 |                   0
 resource_pool_a |             -1 |                   0 |                   0
(2 rows)
```

- gs_wlm_respool_io_info()

  Description: Displays the limit and usage of I/O resources in a resource pool.

  Return type: SETOF record

Name	Type	Description
respool_name	name	Name of the resource pool.
io_limits	integer	Maximum number of I/Os triggered per second (unit: IOPS). The value **0** indicates there is no limit.
io_priority	text	Priority of I/O resource control. It takes effect when the I/O usage reaches 90%. **None** indicates there is no control.
current_iops	integer	Number of times that the current I/O has been triggered (unit: IOPS). ● The value is related to the I/O statistics algorithm and occasionally exceeds the upper limit, which is normal. ● If I/O control is not performed for the resource pool, the value of this parameter is **0**.

NOTE

During I/O control, the unit of I/O operations is defined by the GUC parameter **io_control_unit**. The value of **io_control_unit** is one IOPS.

Example:

```
gaussdb=# SELECT * FROM GS_WLM_RESPOOL_IO_INFO();
 respool_name   | io_limits | io_priority | current_iops
----------------+-----------+-------------+--------------
 default_pool   |         0 | None        |            0
 resource_pool_a |        0 | Low         |            0
(2 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_wlm_user_space_info()

  Description: Displays the storage space usage of a user.

  Return type: SETOF record

Name	Type	Description
user_name	name	Username.
max_permanent_space	bigint	Maximum permanent storage space that can be used by a user, in bytes.
current_permanent_space	bigint	Permanent storage space used by the current user, in bytes.
max_temp_space	bigint	Maximum temporary storage space that can be used by a user, in bytes.
current_temp_space	bigint	Temporary storage space used by the current user, in bytes.
max_spill_space	bigint	Maximum operator-level data flushing storage space that can be used by a user, in bytes.
current_spill_space	bigint	Operator flushing storage space used by the current user, in bytes.

  Example:

```
gaussdb=# SELECT * FROM GS_WLM_USER_SPACE_INFO();
          user_name          | max_permanent_space | current_permanent_space | max_temp_space |
current_temp_space | max_spill_space | current_spill_space
-----------------------------+---------------------+-------------------------+----------------+--------------------
+-----------------+--------------------
 xy                          |                  -1 |                    2464 |             -1 |                  0 |              -1
|                  0
(1 rows)
```

- gs_wlm_session_io_info()

  Description: Displays the I/O usage of a session.

  Return type: SETOF record

Name	Type	Description
session_id	integer	Session ID.
io_limits	integer	Maximum number of I/Os triggered per second (unit: IOPS). The value **0** indicates there is no limit.
io_priority	text	Priority of I/O resource control. It takes effect when the I/O usage reaches 90%. **None** indicates there is no control.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
current_iops	integer	Number of times that the current I/O has been triggered (unit: IOPS). The current I/O statistics occasionally exceed the upper limit, which is related to the I/O statistics algorithm and is normal.
wait_time	integer	Total waiting time after the current I/O exceeds the upper limit.

NOTE

During I/O control, the unit of I/O operations is defined by the GUC parameter **io_control_unit**. The value of **io_control_unit** is one IOPS.

Example:

```
gaussdb=# SELECT * FROM GS_WLM_SESSION_IO_INFO();
   session_id    | io_limits | io_priority | current_iops | wait_time
-----------------+-----------+-------------+--------------+-----------
 139976325986048 |        10 | None        |            0 |      2709
(1 row)
```

- gs_wlm_session_memory_info()

  Description: Displays the memory usage of a session.

  Return type: SETOF record

Name	Type	Description
session_id	integer	Session ID.
sess_used_dynamic_memory	bigint	Used dynamic memory, in bytes.
sess_max_dynamic_memory	bigint	Maximum dynamic memory that can be used, in bytes.

  Example:

```
gaussdb=# SELECT * FROM GS_WLM_SESSION_MEMORY_INFO();
    sessid      | sess_used_dynamic_memory | sess_max_dynamic_memory
----------------+--------------------------+-------------------------
 139976325986048 |                  4326056 |                      -1
 139976402532096 |                  4452664 |                      -1
(2 rows)
```

- gs_emergency_operation_records()

  Description: Displays the usage of the UDS O&M channel on the current node.

  Return type: SETOF record

Name	Type	Description
record_time	timestamptz	Time when the O&M channel is used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
action	text	Operations performed through the O&M channel.
source	text	Source of the O&M channel. The value can be **gsql** or **cm**.
detail	text	Operation result.

- gs_memory_profiling(option integer)

  Description: Generates the jeprof report of the current process of GaussDB. **option** specifies whether to enable the memory collection function. The return result is a string based on the actual input.

  Parameters: **option**: specifies the operation option, which is of the integer type.

  - **0**:Memory collection is disabled. The function returns the "memory profiling closed" message.
  - **1**: Memory collection is enabled. The function returns the "memory profiling opened" message.
  - **2**: Memory collection is enabled. In this case, memory calls are collected and a jeprof report is generated. The report is stored in the **mem_log** directory and named in the format of **jeprof-***YY-mm-dd_HHMMSS*. The function returns the "dump succeeded" message.
  - Others: The message "Invalid input, please input 0(disable), 1(enable) or 2(dump)." is displayed.

  Return type: text.

  NOTE

  - Only users with the SYSADMIN permission can use this function.
  - Enabling memory collection affects performance. Therefore, you are advised to enable this function only in non-performance-sensitive scenarios where memory problems (for example, memory leakage) need to be analyzed. You can run the following command to generate a jeprof report in the PDF format:
    jeprof --show_bytes *gaussdbbin path* -pdf *Path of the generated jeprof report* > *Expected output file*.pdf

- gs_jemalloc_info_details()

  Description: Collects statistics on key fields of jemalloc and generates a jemalloc statistics report. The statistics report is stored in the **mem_log** directory and named in the format of **memory_stat_report-***YY-mm-dd_HHMMSS***.log**.

  Return type: SETOF record

  Table 1-255 gs_jemalloc_info_details columns

Name	Type	Description
nodename	text	Node name
memorytype	text	Column

Name	Type	Description
memorymbytes	integer	Virtual memory size of the corresponding field (unit: MB)

NOTE

Only users with the SYSADMIN permission can use this function.

1.6.30 Trigger Functions

- pg_get_triggerdef(oid)

  Description: Obtains the definition information of a trigger.

  Parameter: OID of the trigger to be queried

  Return type: text

  Example:

```
-- Create the tri_insert table.
gaussdb=# CREATE TABLE tri_insert (a int, b int);
CREATE TABLE
-- Create the trigger_func function.
gaussdb=# CREATE FUNCTION trigger_func() RETURNS trigger LANGUAGE plpgsql AS '
 BEGIN
 RAISE NOTICE ''trigger_func(%) called: action = %, when = %, level = %'', TG_ARGV[0], TG_OP,
TG_WHEN, TG_LEVEL;
 RETURN NULL;
 END;';
CREATE FUNCTION
-- Create the before_ins_stmt_trig trigger.
gaussdb=# CREATE TRIGGER before_ins_stmt_trig BEFORE INSERT ON tri_insert FOR EACH row
EXECUTE PROCEDURE trigger_func('before_ins_stmt');
CREATE TRIGGER
-- Create the after_ins_when_trig trigger.
gaussdb=# CREATE TRIGGER after_ins_when_trig AFTER INSERT ON tri_insert FOR EACH ROW WHEN
(new.a IS NOT NULL) EXECUTE PROCEDURE trigger_func('after_ins_when');
CREATE TRIGGER
-- View the trigger definition information of the tri_insert table.
gaussdb=# SELECT pg_get_triggerdef(oid) FROM pg_trigger WHERE tgrelid = 'tri_insert'::regclass;

pg_get_triggerdef
----------------------------------------------------------------------------------------------------------------------------
---------------------------------
 CREATE TRIGGER after_ins_when_trig AFTER INSERT ON tri_insert FOR EACH ROW WHEN ((new.a IS
NOT NULL)) EXECUTE PROCEDURE trigger_func('after_ins_when')
 CREATE TRIGGER before_ins_stmt_trig BEFORE INSERT ON tri_insert FOR EACH ROW EXECUTE
PROCEDURE trigger_func('before_ins_stmt')
(2 rows)
```

- pg_get_triggerdef(oid, boolean)

  Description: Obtains the definition information of a trigger.

  Parameter: OID of the trigger to be queried and whether it is displayed in pretty mode

  NOTE

  Boolean parameters take effect only when the WHEN condition is specified during trigger creation.

  Return type: text

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
-- View the trigger definition information of the tri_insert table in non-pretty mode.
gaussdb=# SELECT pg_get_triggerdef(oid, false) FROM pg_trigger WHERE tgrelid = 'tri_insert'::regclass;

pg_get_triggerdef
-------------------------------------------------------------------------------------------------------------------------------------------------------------
 CREATE TRIGGER after_ins_when_trig AFTER INSERT ON tri_insert FOR EACH ROW WHEN ((new.a IS NOT NULL)) EXECUTE PROCEDURE trigger_func('after_ins_when')
 CREATE TRIGGER before_ins_stmt_trig BEFORE INSERT ON tri_insert FOR EACH ROW EXECUTE PROCEDURE trigger_func('before_ins_stmt')
(2 rows)

-- View the trigger definition information of the tri_insert table in pretty mode.
gaussdb=# SELECT pg_get_triggerdef(oid, true) FROM pg_trigger WHERE tgrelid = 'tri_insert'::regclass;

pg_get_triggerdef
-------------------------------------------------------------------------------------------------------------------------------------------------------------
 CREATE TRIGGER after_ins_when_trig AFTER INSERT ON tri_insert FOR EACH ROW WHEN (new.a IS NOT NULL) EXECUTE PROCEDURE trigger_func('after_ins_when')
 CREATE TRIGGER before_ins_stmt_trig BEFORE INSERT ON tri_insert FOR EACH ROW EXECUTE PROCEDURE trigger_func('before_ins_stmt')
(2 rows)

-- Clear the tri_insert table.
gaussdb=# DROP TABLE tri_insert CASCADE;
DROP TABLE
-- Clear the trigger_func function.
gaussdb=# DROP FUNCTION trigger_func;
DROP FUNCTION
```

- suppress_redundant_updates_trigger()

  Description: Suppresses update operations that do not change data. This function is called before data update. If the updated data does not change, the update operation does not take effect.

  Return type: trigger.

  > NOTE
  >
  > This function is a built-in trigger execution function. It can be called only when a trigger is defined and takes effect only in a BEFORE UPDATE trigger.

  Example:

```
-- Create the tri_update table.
gaussdb=# CREATE TABLE tri_update (a int, b int);
CREATE TABLE
-- Insert data (1,1).
gaussdb=# INSERT INTO tri_update VALUES(1,1);
INSERT 0 1
-- Create a BEFORE UPDATE trigger. The trigger function is suppress_redundant_updates_trigger.
gaussdb=# CREATE TRIGGER before_update_trig BEFORE UPDATE ON tri_update FOR EACH ROW EXECUTE PROCEDURE suppress_redundant_updates_trigger();
WARNING:  Trigger function with non-plpgsql type is not recommended.
DETAIL:  Non-plpgsql trigger function are not shippable by default.
HINT:  Unshippable trigger may lead to bad performance.
CREATE TRIGGER
-- The data in the table remains unchanged, that is, (1,1). The update operation is not performed.
gaussdb=# UPDATE tri_update SET b=1;
UPDATE 0
-- Drop the tri_update table.
gaussdb=# DROP TABLE tri_update CASCADE;
DROP TABLE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.6.31 Hash Functions

- bucketabstime (value, flag)

  Description: Hashes the value in the abstime format and finds the corresponding hash bucket.

  Parameter: **value** indicates the value to be converted, which is of the abstime type. **flag** is of the int type, indicating the data distribution mode. The value **0** indicates hash distribution.

  Return type: int32

  Example:

```
gaussdb=# SELECT bucketabstime('2011-10-01 10:10:10.112',1);
 bucketabstime
---------------
         13954
(1 row)
```

- bucketbool (value, flag)

  Description: Hashes the value in the boolean format and finds the corresponding hash bucket.

  Parameter: **value** indicates the value to be converted, which is of the Boolean type. **flag** is of the int type, indicating the data distribution mode. The value **0** indicates hash distribution.

  Return type: int32

  Example:

```
gaussdb=# SELECT bucketbool(true,1);
 bucketbool
------------
          1
(1 row)
gaussdb=# SELECT bucketbool(false,1);
 bucketbool
------------
          0
(1 row)
```

- bucketbpchar(value, flag)

  Description: Hashes the value in the bpchar format and finds the corresponding hash bucket.

  Parameter: **value** indicates the value to be converted, which is of the bpchar type. **flag** is of the int type, indicating the data distribution mode. The value **0** indicates hash distribution.

  Return type: int32

  Example:

```
gaussdb=# SELECT bucketbpchar('test',1);
 bucketbpchar
--------------
         9761
(1 row)
```

- bucketbytea (value, flag)

  Description: Hashes the value in the bytea format and finds the corresponding hash bucket.

  Parameter: **value** indicates the value to be converted, which is of the bytea type. **flag** is of the int type, indicating the data distribution mode. The value **0** indicates hash distribution.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: int32

Example:

```
gaussdb=# SELECT bucketbytea('test',1);
 bucketbytea
-------------
        9761
(1 row)
```

- bucketcash (value, flag)

  Description: Hashes the value in the money format and finds the corresponding hash bucket.

  Parameter: **value** indicates the value to be converted, which is of the money type. **flag** is of the int type, indicating the data distribution mode. The value **0** indicates hash distribution.

  Return type: int32

  Example:

  ```
  gaussdb=# SELECT bucketcash(10::money,1);
   bucketcash
  ------------
        8468
  (1 row)
  ```

- getbucket (value, flag)

  Description: Obtains the hash bucket from the distribution key.

  value indicates the value to be entered, which can be of the following types:

  "char", abstime, bigint, Boolean, bytea, character varying, character, date, double precision, int2vector, integer, interval, money, name, numeric, nvarchar2, oid, oidvector, raw, real, record, reltime, smalldatetime, smallint,text, time with time zone, time without time zone, timestamp with time zone, timestamp without time zone, tinyint, uuid

  flag is of the integer type, indicating the data distribution mode.

  Return type: integer

  Example:

  ```
  gaussdb=# SELECT getbucket(10,'H');
   getbucket
  -----------
      14535
  (1 row)

  gaussdb=# SELECT getbucket(11,'H');
   getbucket
  -----------
      13449
  (1 row)

  gaussdb=# SELECT getbucket(11,'R');
   getbucket
  -----------
      13449
  (1 row)

  gaussdb=# SELECT getbucket(12,'R');
   getbucket
  -----------
       9412
  (1 row)
  ```

- ora_hash(expression,[seed])

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Calculates the hash value of a given expression. **expression**: The value can be a character string, time, row ID, or number. The hash value is calculated based on the expression. **seed**: an int8 value that can return different results for the same input value. This parameter is optional and is used to calculate the hash value with a random number.

Return type: hash value of the int8 type.

Example:

```
gaussdb=# SELECT ora_hash(123);
 ora_hash
------------
 4089882933
(1 row)
gaussdb=# SELECT ora_hash('123');
 ora_hash
------------
 2034089965
(1 row)
gaussdb=# SELECT ora_hash('sample');
 ora_hash
------------
 1573005290
(1 row)
gaussdb=# SELECT ora_hash(to_date('2012-1-2','yyyy-mm-dd'));
 ora_hash
------------
 1171473495
(1 row)
gaussdb=# SELECT ora_hash(123,234);
 ora_hash
------------
 -9089505052966355682
(1 row)
gaussdb=# SELECT ora_hash('123',234);
 ora_hash
------------
 5742589019960764616
(1 row)
gaussdb=# SELECT ora_hash('sample',234);
 ora_hash
------------
 -1747984408055821656
(1 row)
gaussdb=# SELECT ora_hash(to_date('2012-1-2','yyyy-mm-dd'),234);
 ora_hash
------------
 -3306025179710572679
(1 row)
gaussdb=# SELECT ora_hash('AAAAAAAAAP//AAAAAAAAAAAAAAAA'::rowid);
  ora_hash
------------
 2241979125
(1 row)
```

NOTE

This function is valid only when **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s2**.

- hash_array(anyarray)

  Description: Hashes an array, obtains the result of an array element using the hash function, and returns the combination result.

  Parameter: data of the anyarray type.

  Return type: integer

  Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT hash_array(ARRAY[[1,2,3],[1,2,3]]);
 hash_array
------------
 -382888479
(1 row)
```

- hash_numeric(numeric)

  Description: Calculates the hash value of numeric data.

  Parameter: data of the numeric type.

  Return type: integer

  Example:

  ```
  gaussdb=# SELECT hash_numeric(30);
   hash_numeric
  --------------
    -282860963
  (1 row)
  ```

- hash_range(anyrange)

  Description: Calculates the hash value of a range.

  Parameter: data of the anyrange type.

  Return type: integer

  Example:

  ```
  gaussdb=# SELECT hash_range(numrange(1.1,2.2));
   hash_range
  ------------
    683508754
  (1 row)
  ```

- hashbpchar(character)

  Description: Calculates the hash value of bpchar.

  Parameter: data of the character type.

  Return type: integer

  Example:

  ```
  gaussdb=# SELECT hashbpchar('hello');
   hashbpchar
  -------------
   -1870292951
  (1 row)
  ```

- hashchar(char)

  Description: Converts char and Boolean data into hash values.

  Parameter: data of the char or bool type.

  Return type: integer

  Example:

  ```
  gaussdb=# SELECT hashbpchar('hello');
   hashbpchar
  -------------
   -1870292951
  (1 row)

  gaussdb=# SELECT hashchar('true');
    hashchar
  ------------
   1686226652
  (1 row)
  ```

- hashenum(anyenum)

  Description: Converts enumerated values to hash values.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter: data of the anyenum type.

Return type: integer

Example:

```
gaussdb=# CREATE TYPE b1 AS ENUM('good', 'bad', 'ugly');
CREATE TYPE
gaussdb=# call hashenum('good'::b1);
  hashenum
------------
 1821213359
(1 row)
gaussdb=# DROP TYPE b1;
```

- hashfloat4(real)

  Description: Converts float4 values to hash values.

  Parameter: data of the real type.

  Return type: integer

  Example:

  ```
  gaussdb=# SELECT hashfloat4(12.1234);
   hashfloat4
  ------------
   1398514061
  (1 row)
  ```

- hashfloat8(double precision)

  Description: Converts float8 values to hash values.

  Parameter: data of the double precision type.

  Return type: integer

  Example:

  ```
  gaussdb=# SELECT hashfloat8(123456.1234);
   hashfloat8
  ------------
   1673665593
  (1 row)
  ```

- hashinet(inet)

  Description: Converts inet or cidr values to hash values.

  Parameter: data of the inet type.

  Return type: integer

  Example:

  ```
  gaussdb=# SELECT hashinet('127.0.0.1'::inet);
    hashinet
  -------------
   -1435793109
  (1 row)
  ```

- hashint1(tinyint)

  Description: Converts INT1 values to hash values.

  Parameter: data of the tinyint type.

  Return type: uint32

  Example:

  ```
  gaussdb=# SELECT hashint1(20);
    hashint1
  -------------
   -2014641093
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- hashint2(smallint)
  Description: Converts INT2 values to hash values.
  Parameter: data of the smallint type.
  Return type: uint32
  Example:

```
gaussdb=# SELECT hashint2(20000);
  hashint2
------------
 -863179081
(1 row)
```

- bucketchar()
  Description: Calculates the hash value of the input parameter.
  Parameter: char, integer
  Return type: integer
- bucketdate()
  Description: Calculates the hash value of the input parameter.
  Parameter: date, integer
  Return type: integer
- bucketfloat4()
  Description: Calculates the hash value of the input parameter.
  Parameter: real, integer
  Return type: integer
- bucketfloat8()
  Description: Calculates the hash value of the input parameter.
  Parameters: double precision, integer
  Return type: integer
- bucketint1()
  Description: Calculates the hash value of the input parameter.
  Parameter: tinyint, integer
  Return type: integer
- bucketint2()
  Description: Calculates the hash value of the input parameter.
  Parameter: smallint, integer
  Return type: integer
- bucketint2vector()
  Description: Calculates the hash value of the input parameter.
  Parameter: int2vector, integer
  Return type: integer
- bucketint4()
  Description: Calculates the hash value of the input parameter.
  Parameter: integer, integer
  Return type: integer

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- bucketint8()
  Description: Calculates the hash value of the input parameter.
  Parameter: bigint, integer
  Return type: integer
- bucketinterval()
  Description: Calculates the hash value of the input parameter.
  Parameter: interval, integer
  Return type: integer
- bucketname()
  Description: Calculates the hash value of the input parameter.
  Parameter: name, integer
  Return type: integer
- bucketnumeric()
  Description: Calculates the hash value of the input parameter.
  Parameter: numeric, integer
  Return type: integer
- bucketnvarchar2()
  Description: Calculates the hash value of the input parameter.
  Parameter: nvarchar2, integer
  Return type: integer
- bucketoid()
  Description: Calculates the hash value of the input parameter.
  Parameters: oid, integer
  Return type: integer
- bucketoidvector()
  Description: Calculates the hash value of the input parameter.
  Parameter: oidvector, integer
  Return type: integer
- bucketraw()
  Description: Calculates the hash value of the input parameter.
  Parameter: raw, integer
  Return type: integer
- bucketreltime()
  Description: Calculates the hash value of the input parameter.
  Parameter: reltime, integer
  Return type: integer
- bucketsmalldatetime()
  Description: Calculates the hash value of the input parameter.
  Parameter: smalldatetime, integer
  Return type: integer

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- buckettext()
  Description: Calculates the hash value of the input parameter.
  Parameter: text, integer
  Return type: integer
- buckettime()
  Description: Calculates the hash value of the input parameter.
  Parameter: time without time zone, integer
  Return type: integer
- buckettimestamp()
  Description: Calculates the hash value of the input parameter.
  Parameter: timestamp without time zone, integer
  Return type: integer
- buckettimestamptz()
  Description: Calculates the hash value of the input parameter.
  Parameter: timestamp with time zone, integer
  Return type: integer
- buckettimetz()
  Description: Calculates the hash value of the input parameter.
  Parameter: time with time zone, integer
  Return type: integer
- bucketuuid()
  Description: Calculates the hash value of the input parameter.
  Parameter: uuid, integer
  Return type: integer
- bucketvarchar()
  Description: Calculates the hash value of the input parameter.
  Parameter: character varying, integer
  Return type: integer

1.6.32 Prompt Message Function

- report_application_error()
  Description: This function can be used to throw errors during PL execution.
  Return type: void

  Table 1-256 report_application_error parameters

Parameter	Type	Description	Required
log	text	Content of an error message.	Yes

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description	Required
code	int4	Error code corresponding to an error message. The value ranges from –20999 to –20000.	No

Example:

```
gaussdb=# CREATE OR REPLACE FUNCTION GET_RESULT_UNKNOWN(
gaussdb(#     IN context_id int, /*context_id*/
gaussdb(#     IN col_type text /*col_type*/
gaussdb(# )RETURNS INTEGER
gaussdb-# AS $$
gaussdb$# BEGIN
gaussdb$#     if col_type is NULL then
gaussdb$#         PG_CATALOG.REPORT_APPLICATION_ERROR('invalid input for the third parameter
col_type should not be null');
gaussdb$#     end if;
gaussdb$#     PG_CATALOG.REPORT_APPLICATION_ERROR('UnSupport data type for
column_value(context: '||context_id||', '||PG_CATALOG.QUOTE_LITERAL(col_type)||')');
gaussdb$#     return -1;
gaussdb$# END;
gaussdb$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
gaussdb=# CALL GET_RESULT_UNKNOWN(NULL, NULL);
ERROR:  invalid input for the third parameter col_type should not be null
CONTEXT:  SQL statement "CALL pg_catalog.report_application_error('invalid input for the third
parameter col_type should not be null')"
PL/pgSQL function get_result_unknown(integer,text) line 4 at PERFORM
```

1.6.33 Fault Injection System Function

gs_fault_inject(int64, text, text, text, text, text)

Description: Specifies a function that cannot be called. The WARNING message **unsupported fault injection** is reported when this function is called, which does not affect or change the database.

Parameter: fault injection of the int64 type (**0**: CLOG extended page; **1**: CLOG page reading; **2**: forcible deadlock)

- If the first input parameter of text is set to **2** and the second input parameter of text is set to **1**, the second input parameter deadlock occurs. Other input parameters are not deadlocked. When the first input parameter is **0** or **1**, the second input parameter indicates the number of the start page from which the CLOG starts to be extended or read.
- The third input parameter of text indicates the number of extended or read pages when the first input parameter is **0** or **1**.
- The fourth to sixth input parameters of text are reserved.

Return type: int64

1.6.34 Redistribution Functions

The following functions are system functions used by gs_redis during redistribution (Do not call them unless absolutely necessary.):

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- pg_get_redis_rel_end_ctid(text, name, int, int)
- pg_get_redis_rel_start_ctid(text, name, int, int)
- pg_enable_redis_proc_cancelable()
- pg_disable_redis_proc_cancelable()
- pg_tupleid_get_blocknum(tid)
- pg_tupleid_get_offset(tid)
- pg_tupleid_get_ctid_to_bigint(ctid)

1.6.35 Distribution Key Recommendation Functions

Distribution key recommendation is used to recommend distribution keys and distribution modes in a distributed database. The purpose is to reduce the labor cost of selecting distribution keys during service migration or rollout.

- sqladvisor.init(char, boolean, boolean, boolean, int, int)

Description: Initializes parameters.

Return type: Boolean

Table 1-257 Parameter description of init

Parameter	Type	Description	Required
kind	char	Recommendation type. Currently, this parameter can only be set to **D**.	Yes
isUseCost	boolean	Specifies whether optimizers are used. If data is available, optimizers are used.	Yes
isUseCollect	boolean	Specifies whether the analysis is started from the collected load. The default value is **false**.	No
isConstraintPrimaryKey	boolean	Specifies whether primary key constraints are retained. The default value is **true**.	No
sqlCount	int	Number of collected SQL statements. The default value is **10000**. The value ranges from 1 to 100000.	No
maxMemory	int	Maximum memory occupied by distribution key recommendation. The default value is **1024**. The value ranges from 1 to 10240, in MB.	No

- sqladvisor.set_weight_params(real, real, real)

Description: Sets the weight of different components in heuristic rules. A default parameter is set when the init function is called. This function does not need to be called during analysis.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: Boolean

Table 1-258 Parameter description of set_weight_params

Parameter	Type	Description	Required
joinWeight	real	Weight of JOIN. The value ranges from 0 to 1000.	Yes
groupbyWeight	real	Weight of GROUP BY. The value ranges from 0 to 1000.	Yes
qualWeight	real	Weight of predicate. The value ranges from 0 to 1000.	Yes

NOTE

This function is optional. When the init function is executed, the default weights of JOIN, GROUP BY, and predicate are preset to 1.0, 0.1, and 0.05, respectively.

- sqladvisor.set_cost_params(bigint, boolean, text)

Description: Parameter that can be set in the Whtif cost model.

Return type: Boolean

Table 1-259 Parameter description of set_cost_params

Parameter	Type	Description	Required
maxTime	bigint	Maximum recommendation duration, in minutes. If the value is less than or equal to **0**, the duration is not limited by default.	Yes
isTotalSQL	boolean	Specifies whether all SQL statements are used for calculation. The value **true** indicates that all SQL statements are used for calculation. The value **false** indicates that SQL statements whose cost is too high or too low are filtered out based on the percentile.	Yes
compressLevel	text	Search space size of the recommendation algorithm. The options are **low**, **med**, and **high**.	Yes

NOTE

- This function is optional. When the init function is executed, **maxTime** is preset to **–1**, **isTotalSQL** is preset to **true**, and **compressLevel** is preset to **high**.
- A lower compression level indicates longer time, and it is more likely that a better result can be achieved.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- sqladvisor.assign_table_type(text)

Description: Specifies a table as a replication table.

Parameter: table name.

Return type: Boolean

NOTE

The specified replication table must be used before **analyze_query** and **analyze_workload** are called.

- sqladvisor.analyze_query(text, int)

Description: Imports SQL statements to be recommended and analyzes the components of the statements.

Return type: Boolean

Table 1-260 Parameter description of analyze_query

Parameter	Type	Description	Required
query	text	SQL statement.	Yes
frequency	int	Frequency of a statement in the load. The default value is **1**. The value ranges from 1 to 2147483647.	No

NOTE

- If the value of the **query** parameter contains special characters, such as single quotation marks ('), you can use single quotation marks (') to escape the special characters.
- This function is not supported in semi-online mode.

- sqladvisor.analyze_workload()

Description: Analyzes the load information collected online.

Return type: Boolean

- sqladvisor.get_analyzed_result(text)

Description: Obtains beneficial components extracted from the current table.

Parameter: text

Return type: record

The following table describes return columns.

Name	Type	Description
schema_name	text	Schema name.

Name	Type	Description
table_name	text	Table name.
col_name	text	Column name.
operator	text	Operator type.
count	int	Number of times that an operator is used.

- sqladvisor.run()

Description: Performs calculation and analysis based on the specified schema and input SQL statements.

Return type: Boolean

- sqladvisor.get_distribution_key()

Description: Obtains the recommendation result.

NOTE

The analysis result is saved in a session. If the session disconnects, the result will be lost.

Return type: record

The following table describes return columns.

Name	Type	Description
db_name	text	Database name.
schema_name	text	Schema name.
table_name	text	Table name.
distribution_type	text	Recommended distribution type.
distribution_key	text	Recommended distribution key.
start_time	timestamp	Recommended start time.
end_time	timestamp	Recommended end time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
cost_improve	text	Cost increase brought by the recommendation result.
comment	text	Comment.

- sqladvisor.clean()

Description: Clears all the memory in the recommendation process of a session.

Return type: Boolean

- sqladvisor.start_collect_workload(int, int)

Description: Starts online load collection.

Return type: Boolean

Table 1-261 Parameter description of start_collect_workload

Parameter	Type	Description	Required
sqlCount	int	Maximum number of SQL statements for online load collection. The value ranges from 1 to 100000 and the default value is **10000**.	Yes
maxMemory	int	Maximum memory occupied by online load collection. The default value is **1024**. The value ranges from 1 to 10240, in MB.	Yes

NOTICE

- The online collection function can be called only by the system administrator.
- The load of only one database can be collected at a time.
- Currently, only common SQL statements as well as DML and DQL statements in stored procedures are supported.

- sqladvisor.end_collect_workload()

Description: Disables online load collection.

Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- The online collection function can be disabled only by the system administrator.

- sqladvisor.clean_workload()

Description: Clears the memory in the load.

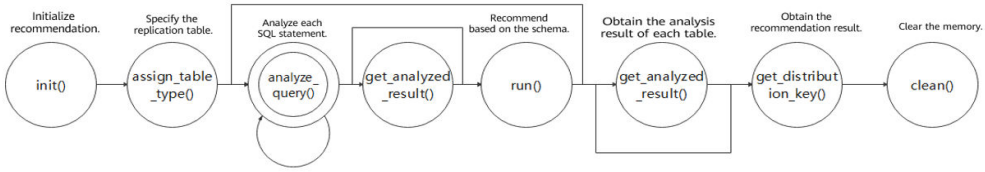
Return type: Boolean

NOTICE

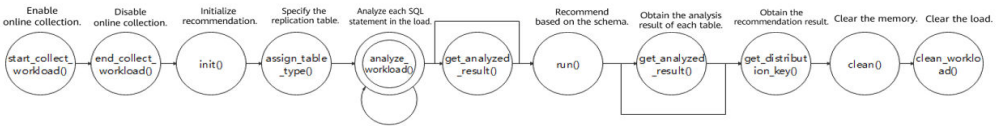
- The function of clearing memory in the load can be called only by the system administrator.
- You must manually execute the cleanup function.

Suggestions

- The following figure shows the call state machine in heuristic or what-if cost recommendation mode.



- The following figure shows the call state machine in semi-online recommendation mode.



1.6.36 Internal Functions

The following functions in GaussDB are used only for internal logic function implementation. You are advised not to use them because unexpected results may occur.

- Selectivity calculation functions

areajoinsel	areasel	arraycontjoinsel	arraycontsel	contjoinsel	contsel	eqjoinsel
eqsel	iclikejoinsel	iclikesel	icnlikejoinsel	icnlikesel	icregexeqjoinsel	icregexeqsel
icregexnejoinsel	icregexnesel	likejoinsel	likesel	neqjoinsel	neqsel	nlikejoinsel

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

nlikesel	positionjoinsel	positionsel	regexeqjoinsel	regexeqsel	regexnejoinsel	regexnesel
scalargtjoinsel	scalargtsel	scalarltjoinsel	scalarltsel	tsmatchjoinsel	tsmatchsel	-

- Statistics collection functions

array_typanalyze	range_typanalyze	ts_typanalyze	local_rto_stat	remote_rto_stat	gs_plan_trace_delete	gs_plan_trace_watch_sqlid
gs_plan_trace_show_sqlids	standby_statement_history_internal	-	-	-	-	-

- Internal functions for sorting

bpchar_sortsupport	bytea_sortsupport	date_sortsupport	numeric_sortsupport	timestamp_sortsupport	-	-

- Internal type processing functions

abstimerecv	euc_jis_2004_to_utf8	int2recv	line_recv	oidvectorrecv_extend	tidrecv	utf8_to_koi8u
anyarray_recv	euc_jp_to_mic	int2vectorrecv	lseg_recv	path_recv	time_recv	utf8_to_shift_jis_2004
array_recv	euc_jp_to_sjis	int4recv	macaddr_recv	pg_node_tree_recv	time_transform	utf8_to_sjis
ascii_to_mic	euc_jp_to_utf8	int8recv	mic_to_ascii	point_recv	timestamp_recv	utf8_to_uhc
ascii_to_utf8	euc_kr_to_mic	internal_out	mic_to_big5	poly_recv	timestamp_transform	utf8_to_win
big5_to_euc_tw	euc_kr_to_utf8	interval_recv	mic_to_euc_cn	pound_nexttoken	timestamptz_recv	uuid_recv
big5_to_mic	euc_tw_to_big5	interval_transform	mic_to_euc_jp	prsd_nexttoken	timetz_recv	varbit_recv

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

big5_to_utf8	euc_tw_to_mic	iso_to_koi8r	mic_to_euc_kr	range_recv	tintervalrecv	varbit_transform
bit_recv	euc_tw_to_utf8	iso_to_mic	mic_to_euc_tw	rawrecv	tsqueryrecv	varchar_transform
boolrecv	float4recv	iso_to_win1251	mic_to_iso	record_recv	tsvectorrecv	varcharrecv
box_recv	float8recv	iso_to_win866	mic_to_koi8r	regclassrecv	txid_snapshot_recv	void_recv
bpcharrecv	gb18030_to_utf8	iso8859_1_to_utf8	mic_to_latin1	regconfigrecv	uhc_to_utf8	win_to_utf8
btoidsortsupport	gbk_to_utf8	iso8859_to_utf8	mic_to_latin2	regdictionaryrecv	unknownrecv	win1250_to_latin2
byteavecv	-	johab_to_utf8	mic_to_latin3	regoperatorrecv	utf8_to_ascii	win1250_to_mic
byteawithoutorderwithequalcolrecv	gtsvector_compress	json_recv	mic_to_latin4	regoperrecv	utf8_to_big5	win1251_to_iso
cash_recv	gtsvector_consistent	koi8r_to_iso	mic_to_sjis	regprocedurerecv	utf8_to_euc_cn	win1251_to_koi8r
charrecv	gtsvector_decompress	koi8r_to_mic	mic_to_win1250	regprocrecv	utf8_to_euc_jis_2004	win1251_to_mic
cidr_recv	gtsvector_penalty	koi8r_to_utf8	mic_to_win1251	regtyperecv	utf8_to_euc_jp	win1251_to_win866
cidrecv	gtsvector_picksplit	koi8r_to_win1251	mic_to_win866	reltimerecv	utf8_to_euc_kr	win866_to_iso
circle_recv	gtsvector_same	koi8r_to_win866	namerecv	shift_jis_2004_to_euc_jis_2004	utf8_to_euc_tw	win866_to_koi8r
cstring_recv	gtsvector_union	koi8u_to_utf8	ngram_nexttoken	shift_jis_2004_to_utf8	utf8_to_gb18030	win866_to_mic

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

date_recv	hll_recv	latin1_to_mic	numeric_recv	sjis_to_euc_jp	utf8_to_gbk	win866_to_win1251
domain_recv	hll_trans_recv	latin2_to_mic	numeric_transform	sjis_to_mic	utf8_to_iso8859	xidrecv
euc_cn_to_mic	-	latin2_to_win1250	nvarchar2recv	sjis_to_utf8	utf8_to_iso8859_1	xidrecv4
euc_cn_to_utf8	inet_recv	latin3_to_mic	oidrecv	smalldatetime_recv	utf8_to_johab	xml_recv
euc_jis_2004_to_shift_jis_2004	int1recv	latin4_to_mic	oidvectorrecv	textrecv	utf8_to_koi8r	-
numeric_bool	int2vectorin_extend	int2vectorout_extend	int2vectorrecv_extend	int2vectorsend_extend	int8_accum	large_seq_rollback_ntree
large_seq_upgrade_ntree	int16eq	int16ge	int16gt	int16in	int16le	int16lt
int16mi	int16mul	int16ne	int16out	int16pl	int16recv	int16send
int16_bool	i16toi1	anyset_in	anyset_out	btint2setcmp	btint4setcmp	btint8setcmp
btsetcmp	btsetint2cmp	btsetint4cmp	btsetint8cmp	btsetsortsupport	float4	float8
hashsetint	hashsettext	int2	int2seteq	int2setge	int2setgt	int2setle
int2setlt	int2setne	int4	int4seteq	int4setge	int4setgt	int4setle
int4setlt	int4setne	int8	int8seteq	int8setge	int8setgt	int8setle
int8setlt	int8setne	set	set_in	set_out	set_recv	set_send
seteq	setge	setgt	setint2eq	setint2ge	setint2gt	setint2le
setint2lt	setint2ne	setint4eq	setint4ge	setint4gt	setint4le	setint4lt

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

setint4ne	setint8eq	setint8ge	setint8gt	setint8le	setint8lt	setint8ne
setle	setlt	setne	settexteq	settextge	settextgt	settextle
settextlt	settextne	settobpchar	settonumber	settonvarchar2	settotext	settovarchar
textseteq	textsetge	textsetgt	textsetle	textsetlt	textsetne	gb18030_2022_to_utf8
utf8_to_gb18030_2022	array_to_nesttable	array_to_indexby_int_table	nesttable_to_array	indexbytableint_to_array	array_to_nesttable	array_to_indexby_int_table
zhs16gbk_to_utf8	utf8_to_zhs16gbk	zhs16gbk_to_gb18030	gb18030_to_zhs16gbk	zhs16gbk_to_gb18030_2022	gb18030_2022_to_zhs16gbk	-

- Internal functions for aggregation operations

array_agg_finalfn	array_agg_transfn	bytea_string_agg_finalfn	bytea_string_agg_transfn	date_list_agg_noarg2_transfn	date_list_agg_transfn	float4_list_agg_noarg2_transfn
float4_list_agg_transfn	float8_list_agg_noarg2_transfn	float8_list_agg_transfn	int2_list_agg_noarg2_transfn	int2_list_agg_transfn	int4_list_agg_noarg2_transfn	int4_list_agg_transfn
int8_list_agg_noarg2_transfn	int8_list_agg_transfn	interval_list_agg_noarg2_transfn	interval_list_agg_transfn	list_agg_finalfn	list_agg_noarg2_transfn	list_agg_transfn
median	median_float8_finalfn	median_interval_finalfn	median_transfn	mode_final	numeric_list_agg_noarg2_transfn	numeric_list_agg_transfn
ordered_set_transition	percentile_cont_float8_final	percentile_cont_interval_final	string_agg_finalfn	string_agg_transfn	timestamp_list_agg_noarg2_transfn	timestamp_list_agg_transfn

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

timestamptz_list_agg_noarg2_transfn	timestamptz_list_agg_transfn	checksumtext_agg_transfn	json_agg_transfn	json_agg_finalfn	json_object_agg_transfn	json_object_agg_finalfn
datea_list_agg_transfn	datea_list_agg_noarg2_transfn	int1_list_agg_noarg2_transfn	int1_list_agg_transfn	smalldatetime_list_agg_noarg2_transfn	smalldatetime_list_agg_transfn	rowid_list_agg_noarg2_transfn
rowid_list_agg_transfn	-	-	-	-	-	-

- Hash internal functions

hashbeginscan	hashbuild	hashbuildempty	hashbulkdelete	hashcostestimate	hashendscan	hashgetbitmap
hashgettuple	hashinsert	hashmarkpos	hashmerge	hashrescan	hashrestrpos	hashvacuumcleanup
hashvarlena	jsonb_hash	datea_hash	rowid_hash	-	-	-

- B-tree index functions

cbtreebuild	cbtreecanreturn	cbtreecostestimate	cbtreegetbitmap	cbtreegettuple	btbeginscan	btbuild
btbuildempty	btbulkdelete	btcanreturn	btcostestimate	btendscan	btfloat4sortsupport	btfloat8sortsupport
btgetbitmap	btgettuple	btinsert	btint2sortsupport	btint4sortsupport	btint8sortsupport	btmarkpos
btmerge	btnamesortsupport	btrescan	btrestrpos	bttextsortsupport	btvacuumcleanup	cbtreeoptions
btvarstrequalimag	-	-	-	-	-	-

- UGIN index functions

ugininsert	uginbeginscan	ugingetbitmap	uginrescan	uginendscan	uginbuild	uginbuildempty

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

uginbulkdelete	uginvacuumcleanup	ugincostestimate	uginoptions	ugin_extract_value_trgm	ugin_extract_query_trgm	ugin_trgm_consistent
ugin_trgm_triconsistent	-	-	-	-	-	-

- GIN index functions

gin_cmp_prefix	gin_extract_tsquery	gin_tsquery_consistent	gin_tsquery_triconsistent	ginarrayconsistent	ginarrayextract	ginarraytriconsistent
ginbeginscan	ginbuild	ginbuildempty	ginbulkdelete	gincostestimate	ginendscan	gingetbitmap
gininsert	ginmarkpos	ginmerge	ginqueryarrayextract	ginrescan	ginrestrpos	ginvacuumcleanup
cginbuild	cgingetbitmap	gin_consistent_jsonb_hash	gin_triconsistent_jsonb_hash	gin_extract_jsonb_hash	gin_extract_jsonb_query_hash	gin_triconsistent_jsonb
gin_compare_jsonb	gin_extract_jsonb	gin_extract_jsonb_query	gin_consistent_jsonb	-	-	-

- GiST index functions

gist_box_compress	gist_box_consistent	gist_box_decompress	gist_box_penalty	gist_box_picksplit	gist_box_same	gist_box_union
gist_circle_compress	gist_circle_consistent	gist_point_compress	gist_point_consistent	gist_point_distance	gist_poly_compress	gist_poly_consistent
gistbeginscan	gistbuild	gistbuildempty	gistbulkdelete	gistcostestimate	gistendscan	gistgetbitmap
gistinsert	gistmarkpos	gistmerge	gistrescan	gistrestrpos	gistvacuumcleanup	range_gist_compress
range_gist_decompress	range_gist_penalty	range_gist_picksplit	range_gist_same	range_gist_union	spg_kd_choose	spg_kd_config

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

spg_kd_picksplit	spg_quad_choose	spg_quad_config	spg_quad_inner_consistent	spg_quad_leaf_consistent	spg_quad_picksplit	spg_text_choose
spg_text_inner_consistent	spg_text_leaf_consistent	spg_text_picksplit	spgbeginscan	spgbuild	spgbuildempty	spgbulkdelete
spgcostestimate	spgendscan	spggetbitmap	spggettuple	spginsert	spgmarkpos	spgmerge
spgrestrpos	spgvacuumcleanup	-	-	-	-	-

- Psort index functions

psortbuild	psortcanreturn	psortcostestimate	psortgetbitmap	psortgettuple	-	-

- UB-tree index functions

ubtbeginscan	ubtbuild	ubtbuildempty	ubtbulkdelete	ubtcanreturn	ubtcostestimate	ubtendscan
ubtgetbitmap	ubtgettuple	ubtinsert	ubtmarkpos	ubtmerge	ubtoptions	ubtrescan
ubtrestrpos	ubtvacuumcleanup	-	-	-	-	-

- BM25 index functions (not supported in the current version)

gs_bm25beginscan	gs_bm25build	gs_bm25buildempty	gs_bm25bulkdelete	gs_bm25insert	gs_bm25costestimate	gs_bm25endscan
gs_bm25options	gs_bm25gettuple	gs_bm25vacuumcleanup	gs_bm25rescan	-	-	-

- PL/pgSQL internal function

plpgsql_inline_handler	-	-	-	-	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Foreign table-related internal functions

dist_fdw_handler	roach_handler	streaming_fdw_handler	dist_fdw_validator	file_fdw_handler	file_fdw_validator	log_fdw_handler
gc_fdw_handler	gc_fdw_validator	dblink_fdw_handler	dblink_fdw_validator	-	-	-

- Internal function related to data skew optimization

distributed_count	-	-	-	-	-	-

- Internal functions related to table statistics

pgxc_get_stat_dirty_tables	pgxc_stat_dirty_tables	get_global_stat_all_tables	get_summary_stat_all_tables	-	-	-

- Functions for reading data remotely
  - gs_read_block_from_remote is used to read the pages of a non-segment-page table file. By default, only the initial user can view the data. Other users can use the data only after being granted permissions.
  - gs_read_segment_block_from_remote is used to read the pages of a segment-page table file. By default, only the initial user can view the data. Other users can use the data only after being granted permissions.
- Functions for reading files remotely
  - gs_read_file_size_from_remote is used to read the size of a specified file. Before using the gs_repair_file function to repair a file, you need to obtain the size of the file from the remote end to verify the missing file information and repair the missing files one by one. By default, only the initial user can view the data. Other users can use the data only after being granted permissions.
  - gs_read_file_from_remote is used to read a specified file. After obtaining the file size by using the gs_read_file_size_from_remote function, gs_repair_file reads the remote file segment by segment using this function. By default, only the initial user can view the data. Other users can use the data only after being granted permissions.
- Auxiliary functions for incrementally rebuilding other standby or cascaded standby DNs using the standby DN.
  - gs_standby_incremental_filemap_create is used to create a temporary filemap file for incremental rebuilding on the standby DN. The filemap file is used to store the path and size of the data to be transmitted during the incremental rebuilding. This API can be called only when the initial user is used and **application** is **gs_rewind**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - gs_standby_incremental_filemap_insert is used to insert file information into a specified temporary filemap file. The file information includes the file path, transfer start point, length of data to be transmitted at a time, and rebuild flag bit. This API can be called only when the initial user is used and **application** is **gs_rewind**.
  - gs_standby_incremental_filemap_execute is used to obtain file information stored in a specified temporary filemap file and delete the specified filemap file for data transmission during incremental standby DN rebuilding. This API can be called only when the initial user is used and **application** is **gs_rewind**.

- Functions for obtaining remote information in segment-page mode

remote_gs_seg_get_datafiles	remote_gs_seg_get_extents	remote_gs_seg_get_segment	remote_gs_seg_get_spc_extents	remote_gs_seg_spc_segments	remote_segment_space_info	-

- Ledger database function

get_dn_hist_relhash	-	-	-	-	-	-

- Other functions

to_datea	sysdate_a	get_char_max_length	to_datea_internal	gs_get_part_relfilenode_id	gs_check_internal	-

- View-related reference functions

adm_hist_sqlstat_func	adm_hist_sqlstat_idlog_func	adm_hist_sqltext_func	-	-	-	-

- gs_txn_snapshot system catalog maintenance function

  gs_insert_delete_txn_snapshot is used to maintain the gs_txn_snapshot system catalog of each node in distributed GTM-Lite mode. Only system administrators can call gs_insert_delete_txn_snapshot.

- XMLType function

isschemavalid (unavailable)	-	-	-	-	-	-

- Functions for creating and deleting indexes online.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

gs_mark_indisvalid	gs_mark_indisvalid_false	-	-	-	-	-

- pivot functions

pivot_anyelement	pivot_bit	pivot_bool	pivot_bpchar	pivot_bytea	pivot_date	pivot_float4
pivot_float8	pivot_hll	pivot_int1	pivot_int2	pivot_int4	pivot_int8	pivot_interval
pivot_json	pivot_money	pivot_numeric	pivot_oid	pivot_smalldatetime	pivot_tdigestdata	pivot_text
pivot_tid	pivot_time	pivot_datea	pivot_timestamp	pivot_timestamptz	pivot_timetz	pivot_xml
pivot_char	pivot_varchar	pivot_nvarchar2	pivot_raw	pivot_clob	pivot_abstime	pivot_reltime
pivot_blob	pivot_rowid	pivot_varbit	-	-	-	-

- subtype functions

subtype_in	subtype_recv	-	-	-	-	-

- nesttable functions

nesttable_in	nesttable_out	nesttable_count	nesttable_delete	nesttable_eq	nesttable_except	nesttable_except_distinct
nesttable_exists	nesttable_extend	nesttable_first	nesttable_intersect	nesttable_intersect_distinct	nesttable_last	nesttable_limit
nesttable_ne	nesttable_next	nesttable_prior	nesttable_to_array	nesttable_trim	nesttable_union	nesttable_union_distinct

- indexbytable functions

indexbytable_count	indexbytable_in	indexbytable_out	indexbytableint_delete	indexbytableint_exists	indexbytableint_first	indexbytableint_last

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

indexbytableint_next	indexbytableint_prior	indexbytableint_to_array	indexbytablevarchar_delete	indexbytablevarchar_exists	indexbytablevarchar_first	indexbytablevarchar_last
indexbytablevarchar_next	indexbytablevarchar_prior	-	-	-	-	-

- Message queue functions

mq_clear	mq_get_queue	mq_query	mq_register_pull	mq_send_message	mq_sync	mq_sync_ccn
mq_transaction_processing	mq_unregister	-	-	-	-	-

- Sorting calculation functions

tdigest_in	tdigest_out	calculate_quantile_of	calculate_value_at	tdigest_merge	tdigest_merge_to_one	tdigest_mergep

- Internal functions for GsIVFFLAT indexes

ivfflatbeginscan	ivfflatbuild	ivfflatbuildempty	ivfflatbulkdelete	ivfflatinsert	ivfflatcostestimate	ivfflatendscan
ivfflatoptions	ivfflatgettuple	ivfflatvacuumcleanup	ivfflatrescan	-	-	-

- Internal functions for GsDiskANN indexes

diskannbeginscan	diskannbuild	diskannbuildempty	diskannbulkdelete	diskanninsert	diskannccostestimate	diskannendscan
diskannoptions	diskanngettuple	diskannvacuumcleanup	diskannrescan	-	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.6.37 Sensitive Data Discovery Function

- gs_sensitive_data_discovery(scan_target text, scan_classifier text)

  Description: Scans target data and returns statistical scanning results.

  Parameters:

  scan_target: specifies the object to be scanned. The value must be the name of a schema, table, or column, and you must include its parent names. For example, to scan a column, specify *schema_name*.*table_name*.*column_name*.

  scan_classifier: specifies the classifier to be used. Five classifiers can be specified: **email**, **creditcard**, **phonenumber**, **chinesename**, and **encryptedcontent**. Multiple classifiers are separated by commas (,). The value **all** indicates all classifiers are selected.

  Return type: record

  For details, see "Feature Usage Guidelines > Sensitive Data Discovery" in *Developer Guide*.

- gs_sensitive_data_discovery_detail(scan_target text, scan_classifier text)

  Description: Scans the target data and returns the detailed scanning result.

  scan_target: specifies the object to be scanned. The value must be the name of a schema, table, or column, and you must include its parent names. For example, to scan a column, specify *schema_name*.*table_name*.*column_name*.

  scan_classifier: specifies the classifier to be used. Five classifiers can be specified: **email**, **creditcard**, **phonenumber**, **chinesename**, and **encryptedcontent**. Multiple classifiers are separated by commas (,). The value **all** indicates all classifiers are selected.

  Return type: record

  For details, see "Feature Usage Guidelines > Sensitive Data Discovery" in *Developer Guide*.

1.6.38 Dynamic Data Masking Functions

- creditcardmasking(col text, letter char default 'x')

  Description: Replaces the digits before the last four bits following the col string with letters.

  Parameter: Character string to be replaced or character string used for replacement

  Return type: text.

- basicemailmasking(col text, letter char default 'x')

  Description: Replaces the characters before the first at sign (@) in the col string with letters.

  Parameter: Character string to be replaced or character string used for replacement

  Return type: text.

- fullemailmasking(col text, letter char default 'x')

  Description: Replaces the characters (except first **@**) before the last period (.) in the col string with letters.

  Parameter: Character string to be replaced or character string used for replacement

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: text.

- alldigitsmasking(col text, letter char default '0')

  Description: Replaces the digits in the col string with letters.

  Parameter: Character string to be replaced or character string used for replacement

  Return type: text.

- shufflemasking(col text)

  Description: Sorts the characters in the col string out of order.

  Parameter: Character string to be replaced or character string used for replacement

  Return type: text.

- randommasking(col text)

  Description: Randomizes the characters in the col string.

  Parameter: Character string to be replaced or character string used for replacement

  Return type: text.

- **Optional:** regexpmasking(col text, reg text, replace_text text, pos INTEGER default 0, reg_len INTEGER default -1)

  Description: Replaces the col string with a regular expression.

  Parameters: Character string to be replaced, regular expression, replacement start position, and replacement length.

  Return type: text.

1.6.39 Hierarchical Recursion Query Functions

The following functions can be used in a hierarchical recursion query statement to return information about the connection path.

- sys_connect_by_path(col, separator)

  Description: Returns the connection path from the root node to the current row. This function applies only to hierarchical recursion queries.

  The **col** parameter indicates the name of the column displayed in the path. Only columns of the CHAR, VARCHAR, NVARCHAR2, TEXT, ROWID, INT1, INT2, INT4, INT8, FLOAT4, FLOAT8, and NUMERIC type are supported. The **separator** parameter indicates the separator between path nodes. Currently, the col input of the function does not support expression input. If the column content is the same as the separator, the function can run properly and no error is reported.

  Return type: TEXT

  Example:

```
-- Create a table and insert data.
gaussdb=# CREATE TABLE connect_by_table(id int,pid int,name text);
CREATE TABLE
gaussdb=# INSERT INTO connect_by_table VALUES(1,0,'a'),(2,1,'b'),(3,2,'c'),(4,1,'d');
INSERT 0 4
-- Query.
gaussdb=# SELECT *, sys_connect_by_path(name, '-') FROM connect_by_table START WITH id = 1
CONNECT BY prior id = pid ORDER BY pid, id;
 id | pid | name | sys_connect_by_path
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
----+-----+------+---------------------
 1 |   0 | a    | -a
 2 |   1 | b    | -a-b
 4 |   1 | d    | -a-d
 3 |   2 | c    | -a-b-c
(4 rows)
-- Restore the environment.
gaussdb=# DROP TABLE connect_by_table;
DROP TABLE
```

- connect_by_root(col)

  Description: Returns the value of a column in the top-most parent row of the current row. This function applies only to hierarchical recursion queries.

  The **col** parameter indicates the name of an output column. Only columns of the following types are supported: INT8, INT1, INT2, OID, INT4, BOOL, CHAR, NAME, FLOAT4, FLOAT8, ABSTIME, RELTIME, DATE, CASH, TIME, TIMESTAMP, TIMESTAMPTZ, SMALLDATETIME, UUID, INTERVAL, TIMETZ, INT2VECTOR, CLOB, NVARCHAR2, VARCHAR, TEXT, ROWID, VECTOR, BPCHAR, RAW, BYTEA, NUMERIC, XID, CID, and TID. The whitelist restriction cannot be bypassed by forcibly converting the type to the preceding types, for example, connect_by_root(col::text) (where **col** is not of the type in the whitelist) is not supported.

  Return type: data type of the specified column **col**.

  Example:

```
-- Create a table and insert data.
gaussdb=# CREATE TABLE connect_by_table(id int,pid int,name text);
CREATE TABLE
gaussdb=# INSERT INTO connect_by_table VALUES(1,0,'a'),(2,1,'b'),(3,2,'c'),(4,1,'d');
INSERT 0 4
-- Query.
gaussdb=# SELECT *, connect_by_root(name) FROM connect_by_table START WITH id = 1 CONNECT
BY prior id = pid ORDER BY pid, id;
 id | pid | name | connect_by_root
----+-----+------+-----------------
 1 |   0 | a    | a
 2 |   1 | b    | a
 4 |   1 | d    | a
 3 |   2 | c    | a
(4 rows)
-- Restore the environment.
gaussdb=# DROP TABLE connect_by_table;
DROP TABLE
```

1.6.40 Hotkey Feature Functions

- gs_stat_get_hotkeys_info()

  Description: Obtains hotkey information queried on the local node.

  Return type: Tuple

  Example:

```
gaussdb=# SELECT * FROM  gs_stat_get_hotkeys_info() order by count, hash_value;
 database_name | schema_name |     table_name     | key_value | hash_value | count
---------------+-------------+--------------------+-----------+------------+-------
 regression    | public      | hotkey_single_col | {22}      | 1858004829 |     2
 regression    | public      | hotkey_single_col | {11}      | 2011968649 |     2
(2 rows)
```

- gs_stat_clean_hotkeys()

  Description: Clears hotkey cache and resets hotkey status information.

  Return type: bool, and the return value is always **true**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT * FROM  gs_stat_clean_hotkeys();
 gs_stat_clean_hotkeys
-----------------------
 t
(1 row)
```

1.6.41 Global SysCache Functions

- gs_gsc_table_detail(database_id default NULL, rel_id default NULL)

  Description: Queries the table metadata in the global system cache in a database. The user who calls this function must have the **SYSADMIN** permission.

  Parameter: Specifies the database and table to be queried in the global system cache. The default value **NULL** or value **–1** of **database_id** indicates all databases. The value **0** indicates a shared table. Other values indicate a specified database and a shared table. **rel_id** indicates the OID of a specified table. The default value **NULL** or value **–1** indicates all tables. Other values indicate a specified table. If **database_id** does not exist, an error is reported. If **rel_id** does not exist, the query result is empty.

  Return type: tuple

  ```
  gaussdb=# SELECT * FROM  gs_gsc_table_detail(-1) limit 1;
  database_oid | database_name | reloid |          relname          | relnamespace | reltype | reloftype |
  relowner | relam | relfilenode | reltablespace | relhasindex | relisshared | relkind | relnatts | relhasoids |
  relhaspkey | parttype | tdhasuids | attnames  | extinfo
  --------------+---------------+--------+--------------------------+--------------+---------+-----------+----------
  +-------+-------------+---------------+-------------+-------------+---------+----------+------------+------------
  +----------+-----------+-----------+---------
  0 |               |   2676 | pg_authid_rolname_index |           11 |       0 |         0 |       10 |   403 |           0
  |          1664 | f           | t           | i       |         1 | f          | f          | n       | f         | 'rolname' |
  (1 row)
  ```

- gs_gsc_catalog_detail(database_id default NULL, rel_id default NULL)

  Description: Queries the system catalog row information in the global system cache in a database. The user who calls this function must have the **SYSADMIN** permission.

  Parameter: Specifies the database and table to be queried in the global system cache. The default value **NULL** or value **–1** of **database_id** indicates all databases. The value **0** indicates a shared table. Other values indicate a specified database and a shared table. **rel_id** indicates the ID of a specified table, including all system catalogs in the system cache. The default value **NULL** or value **–1** indicates all tables. Other values indicate a specified table. If **database_id** does not exist, an error is reported. If **rel_id** does not exist, the result is empty.

  Return type: tuple

  Example:

  ```
  -- Obtain the OID of a specific database through pg_database, for example, running SELECT oid, * FROM pg_database;.
  -- In the returned tuple, find the value of the OID column based on the datname column, and then run the following query command. In the example, the obtained OID is 16574.
  gaussdb=# SELECT * FROM  gs_gsc_catalog_detail(16574, 1260);
   database_id | database_name | rel_id | rel_name  | cache_id |  self  |  ctid  | infomask | infomask2 |
  hash_value | refcount
  -------------+---------------+--------+-----------+----------+--------+--------+----------+-----------+------------
  +----------
            0 |               |   1260 | pg_authid |       10 | (0, 9) | (0, 9) |    10507 |        26 |  531311568 |
  10
  ```

```
        0 |          |  1260 | pg_authid |      11 | (0, 4) | (0, 4) |    2313 |      26 |  365368336 |       1
        0 |          |  1260 | pg_authid |      11 | (0, 9) | (0, 9) |   10507 |      26 | 3911517328 |
10
        0 |          |  1260 | pg_authid |      11 | (0, 7) | (0, 7) |    2313 |      26 | 1317799983 |
1
        0 |          |  1260 | pg_authid |      11 | (0, 5) | (0, 5) |    2313 |      26 | 3664347448 |
1
        0 |          |  1260 | pg_authid |      11 | (0, 1) | (0, 1) |    2313 |      26 |  276477273 |       1
        0 |          |  1260 | pg_authid |      11 | (0, 3) | (0, 3) |    2313 |      26 | 2465837659 |
1
        0 |          |  1260 | pg_authid |      11 | (0, 8) | (0, 8) |    2313 |      26 | 3205288035 |
1
        0 |          |  1260 | pg_authid |      11 | (0, 6) | (0, 6) |    2313 |      26 |  131811687 |       1
        0 |          |  1260 | pg_authid |      11 | (0, 2) | (0, 2) |    2313 |      26 | 1226484587 |
1
(10 rows)
```

- gs_gsc_clean(database_id default NULL)

  Description: Clears the global system cache. Note that data in use will not be cleared. The user who calls this function must have the **SYSADMIN** permission.

  Parameter: Specifies the database whose global system cache needs to be cleared. The default value **NULL** or value **–1** indicates that the global system cache of all databases is forcibly cleared. The value **0** indicates that the global system cache of only the shared table is cleared. Other values indicate that the global system cache of a specified database and a specified shared table is cleared. If **database_id** does not exist, an error is reported.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT * FROM  gs_gsc_clean();
  gs_gsc_clean
  --------------
  t
  (1 row)
  ```

- gs_gsc_dbstat_info(database_id default NULL)

  Description: Obtains GSC memory statistics on the local node, including cache query, hit, loading, expiration, and occupied space information of tuples, relations, and partitions, database-level eviction information, thread reference information, and memory usage information. This function can be used to locate performance problems. For example, if the value of the hits or searches array is far less than 1, the value of **global_syscache_threshold** may be too small. As a result, the query hit ratio decreases. The user who calls this function must have the **SYSADMIN** permission.

  Parameter: Specifies the global system cache statistics of the database to be queried. Value **NULL** or **–1** indicates that all databases are queried. Value **0** indicates that only the shared table is queried. Other values indicate that a specified database and a specified shared table are queried. If an invalid value is entered, an error is reported, indicating that **database_id** does not exist.

  Return type: tuple

  Example:

  ```
  gaussdb=# SELECT * FROM  gs_gsc_dbstat_info();
  database_id | database_name | tup_searches | tup_hits | tup_miss | tup_count | tup_dead | tup_memory
  | rel_searches | rel_hits | rel_mis
  s | rel_count | rel_dead | rel_memory | part_searches | part_hits | part_miss | part_count | part_dead |
  part_memory | total_memory | swa
  pout_count | refcount
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-------------+---------------+--------------+----------+----------+-----------+----------+------------
+--------------+----------+--------
--+-----------+----------+------------+---------------+-----------+-----------+------------+-----------
+-------------+--------------+----
-----------+----------
          0 |               |         300 |      235 |       31 |        22 |        2 |       9752 |         598 |      108
|       1
8 |        18 |        0 |      77720 |            0 |         0 |         0 |         0 |         0 |         0 |       752912 |
         0 |         0
       16574 | testdb      |         3368 |     2289 |      329 |       273 |        0 |      92593 |        1113 |
524 |        4
8 |        48 |        0 |     340456 |            0 |         0 |         0 |         0 |         0 |         0 |      4124792 |
         0 |        10
(2 rows)
```

1.6.42 Data Damage Detection and Repair Functions

- The restrictions for repairing files or pages of the primary and standby nodes are as follows.

File Type	File/ Page	Primary/ Standby	Detection and Repair
Ordinary row-store tables (including Astore and Ustore), segment-page tables, and cross-page transparent compression tables, excluding indexes and compressed tables	File and page	Primary	Manual detection and repair.
Undo files (excluding undo meta)	Page	Primary	Manual detection and repair (excluding analyse verify).
init fork file for unlogged tables	File	Primary	Manual detection and repair.
Ordinary row-store tables (Astore and Ustore), ILM compressed tables, cross-page transparent compression tables, segment-page tables, hash bucket tables, indexes (B-tree and UB-tree), and undo files (excluding undo meta. Undo files support only when the CRC verification fails.)	Page	Standby	Automatic detection and repair during replay.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

CAUTION

The main standby node and cascaded standby node of the standby cluster can be repaired.

A critical section is used in the database to ensure the consistency of access to key resources. Errors cannot be thrown in the critical section. If a damaged page is accessed in the critical area, PANIC is triggered. Repair detection and automatic repair are not supported. Typical operations that access the critical area during execution include database write operations such as INSERT, DELETE, UPDATE, and DROP.

During the execution of AUTOVACUUM and ANALYZE, the remote access function is temporarily disabled to ensure the accuracy of page reading. As a result, the standby node may fail to restore the primary node.

- gs_verify_data_file(verify_segment bool)

  Description: Checks whether files in the current database of the current instance are lost. The verification only checks whether intermediate segments are lost in the main file of the data table. The default value is **false**, indicating that the segment-page table data file is not verified. When this parameter is set to **true**, only segment-page table files are verified; hash bucket files are not supported for verification. By default, only the initial user, users with the SYSADMIN attribute, and users with the OPRADMIN attribute in the O&M mode can view the information. Other users can view the information only after being granted permissions.

  The returned result is as follows:

  – Non-segment-page table: **rel_oid** and **rel_name** indicate the table OID and table name of the corresponding file, and **miss_file_path** indicates the relative path of the lost file.

  – Segment-page table: All tables are stored in the same file. Therefore, **rel_oid** and **rel_name** cannot display information about a specific table. For a segment-page table, if the first file is damaged, the subsequent files such as .1 and .2 are not checked. For example, if 3, 3.1, and 3.2 are damaged, only 3 damage can be detected. When the number of segment-page files is less than 5, the files that are not generated are also detected during function detection. For example, if there are only files 1 and 2, files 3, 4, and 5 are detected during segment-page file detection. In the following examples, the first is an example of checking a non-segment-page table, and the second is an example of checking a segment-page table.

  Parameters:

  – verify_segment

    Specifies the range of files to be checked. **false** indicates that non-segment-page tables are verified. **true** indicates that segment-page tables are verified.

    The value can be **true** or **false** (default value).

  Return type: record

  Example (The abnormal line is displayed only when an exception is detected. Otherwise, no line is displayed.):

  Verify a non-segment-page table:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT * FROM  gs_verify_data_file();
node_name         | rel_oid |  rel_name    |  miss_file_path
------------------+---------+--------------+------------------
dn_6001_6002_6003 |   16554 |     test     | base/16552/24745
```

Verify a segment-page table:

```
gaussdb=# SELECT * FROM  gs_verify_data_file(true);
     node_name     | rel_oid | rel_name | miss_file_path
-------------------+---------+----------+----------------
 dn_6001_6002_6003 |       0 | none     | base/16552/2
```

- gs_repair_file(tableoid Oid, path text, timeout int)

  Description: Repairs the file based on the input parameters. Only the primary DN with normal primary/standby connection is supported. Only the main file of the data table can be repaired. The parameters are set based on the OID and path returned by the gs_verify_data_file function. The value of table OID for a segment-page table ranges from 0 to 4294967295. (The internal verification determines whether a file is a segment-page table file based on the file path. The table OID is not used for a segment-page table file.) If the repair is successful, **true** is returned. If the repair fails, the failure cause is displayed. By default, only the initial user, users with the SYSADMIN permission, and users with the OPRADMIN permission in the O&M mode on the primary DN can view the information. Other users can view the information only after being granted permissions.

  CAUTION

  1. If a file on a DN is damaged, a verification error at the PANIC level is reported when the DN is promoted to primary. The DN cannot be promoted to primary, which is normal.
  2. If a file exists but its size is 0, the file will not be repaired. To repair the file, you need to delete the file whose size is 0 and then repair it.
  3. You can delete a file only after the file descriptor is automatically closed. You can manually restart the process or perform a primary/standby switchover.
  4. The restoration function requires that the paths of the files to be restored on the primary and standby nodes be the same. If the paths are different, the restoration fails. That is, if the RELATIVE LOCATION tablespace is used, the path is different because tblspcversiondir is different. As a result, the restoration fails.

  Parameters:

  – tableoid

    OID of the table corresponding to the file to be repaired. Set this parameter based on the **rel_oid** column in the list returned by the gs_verify_data_file function.

    Value range: OID ranging from 0 to 4294967295. Note: A negative value will be forcibly converted to a non-negative integer.

  – path

    Path of the file to be repaired. Set this parameter based on the **miss_file_path** column in the list returned by the gs_verify_data_file function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: a string

– timeout

Specifies the duration for waiting for the standby DN to replay. The repair file needs to wait for the standby DN to be put back to the corresponding location on the current primary DN. Set this parameter based on the replay duration of the standby DN.

Value range: 60s to 3600s.

Return type: Boolean

Example (Set **tableoid** and **path** based on the output of gs_verify_data_file):

Page-based storage:

```
gaussdb=# SELECT * FROM gs_repair_file(16554,'base/16552/24745',360);
gs_repair_file
----------------
t
```

Segment-page storage:

```
gaussdb=# SELECT * FROM gs_repair_file(16554,'base/16552/2',360);
gs_repair_file
----------------
t
```

- local_bad_block_info()

  Description: Displays the page damage of the instance. You can read the page from the disk and record the page CRC failure. By default, only the initial user, users with the SYSADMIN attribute, users with the MONADMIN attribute, users with the OPRADMIN attribute in the O&M mode, and monitor users can view the information. Other users can view the information only after being granted permissions. **file_path** indicates the relative path of a damaged file. The segment-page format is supported. **block_num** indicates the number of the page where the file is damaged. The page number starts from 0. **check_time** indicates the time when the page damage is detected. **repair_time** indicates the time when the page is repaired.

  Return type: record

  Example (Related entries are displayed only when there are damaged records. Otherwise, no log is displayed.):

```
gaussdb=# SELECT * FROM  local_bad_block_info();
node_name    | spc_node | db_node | rel_node| bucket_node | fork_num | block_num |    file_path     |
check_time            |   repair_time
-----------------+-------+--------+--------+--------------+----------+-----------+-----------------
+--------------------------+-------------------------------
dn_6001_6002_6003|  1663 |  16552 |  24745 |        -1    |    0    | 0        | base/16552/24745 |
2022-01-13 20:19:08.385004+08 | 2022-01-13 20:19:08.407314+08
```

- remote_bad_block_info()

  Description: Queries the page damage of other instances except the current instance when a query is performed on the CN. The recorded data is the same as that of the local_bad_block_info function executed on other instances. The execution result on the DN is empty. By default, only the initial user, users with the SYSADMIN attribute, users with the MONADMIN attribute, users with the OPRADMIN attribute in the O&M mode, and monitor users can view the information. Other users can view the information only after being granted permissions.

  Return type: record

- local_clear_bad_block_info()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Deletes data of repaired pages from local_bad_block_info, that is, information whose **repair_time** is not empty. By default, only the initial user, users with the SYSADMIN permission, users with the OPRADMIN attribute in the O&M mode, and monitor users can view the information. Other users can view the information only after being granted permissions.

Return type: Boolean

Example:

```
gaussdb=# SELECT * FROM  local_clear_bad_block_info();
result
--------
t
```

- remote_clear_bad_block_info()

  Description: Clears the data of the repaired pages of other instances except the current instance when this function is executed on the CN, that is, information whose **repair_time** is not empty. The execution result on the DN is empty. By default, only the initial user, users with the SYSADMIN permission, users with the OPRADMIN attribute in the O&M mode, and monitor users can view the information. Other users can view the information only after being granted permissions.

  Return type: record

- gs_inconsistency_info(reset bool)

  Description: Displays or resets data inconsistency information in the database. If the input parameter **reset** is set to **true**, data inconsistency information is reset. If the input parameter **reset** is set to **false**, data inconsistency information is displayed. By default, only the initial user, users with the SYSADMIN attribute, users with the MONADMIN attribute, users with the OPRADMIN attribute in the O&M mode, and monitor users can view the information. Other users can view the information only after being granted permissions. **inconsistency_type** indicates the inconsistency type, **node_name** indicates the node name, **spc_node** indicates the tablespace ID, **db_node** indicates the database node ID, **rel_node** indicates the table ID, **bucket_node** indicates the hash bucket ID, **backend** indicates the temporary table ID, **index_oid** indicates the index OID, **tuple_blkno** indicates the page number of the heap table where the tuple is located, **tuple_offset** indicates the offset of the heap table page where the tuple is located, **count** indicates the number of times the inconsistency information occurs, and **last_update_time** indicates the last time when the inconsistent information is updated.

  Return type: record

  When the input parameter **reset** is set to **false** and inconsistency information is detected, related items are displayed. In other cases, no line is displayed. Example:

```
gaussdb=# SELECT * FROM gs_inconsistency_info(false);
 inconsistency_type  | node_name | spc_node | db_node | rel_node | bucket_node | backend | index_oid
| tuple_blkno | tuple_offset | count |        last_update_time
---------------------+-----------+----------+---------+----------+-------------+---------+-----------+-------------
+--------------+-------+-------------------------------
 INCONSISTENCY_INDEX | sgnode    |     1663 |   12902 |    16722 |          -1 |      -1 |     16728 |
0 |            1 |     1 | 2024-11-20 16:18:56.85289-05
```

- gs_verify_and_tryrepair_page (path text, blocknum Oid, verify_mem bool, is_segment bool)

  Description: Verifies the page specified by the instance. By default, only the initial user, users with the SYSADMIN permission, and users with the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

OPRADMIN permission in the O&M mode on the primary DN can view the information. Other users can view the information only after being granted permissions. In the command output, **disk_page_res** indicates the verification result of the page on the disk, **mem_page_res** indicates the verification result of the page in the memory, and **is_repair** specifies whether the repair function is triggered during the verification. **t** indicates that the page is repaired, and **f** indicates that the page is not repaired.

NOTE

- If a page on a DN is damaged, a verification error at the PANIC level is reported when the DN is promoted to primary. The DN cannot be promoted to primary, which is normal.
- The repair triggered by this function can only repair pages in the memory. The repair takes effect only after the memory pages are flushed to disks and the physical pages are repaired.

Parameters:

- path

  Path of the damaged file. Set this parameter based on the **file_path** column in local_bad_block_info. To verify the undo pages of the Ustore table, enter the path of the undo pages to be verified.

  Value range: a string

- blocknum

  Page number of the damaged file. Set this parameter based on the **block_num** column in local_bad_block_info. If you want to verify the undo pages of the Ustore table, enter the block number of the undo pages to be verified.

  Value range: OID ranging from 0 to 4294967295. Note: A negative value will be forcibly converted to a non-negative integer.

- verify_mem

  Specifies whether to verify a specified page in the memory. If this parameter is set to **false**, only pages on the disk are verified. If this parameter is set to **true**, pages in the memory and on the disk are verified. If a page on the disk is damaged, the system verifies the basic information of the page in the memory and flushes the page to the disk to restore the page. If a page is not found in the memory during memory page verification, the page on the disk is read through the memory API. During this process, if the disk page is faulty, the remote read automatic repair function is triggered.

  Value range: The value is of a Boolean type and can be **true** or **false**.

- is_segment

  Determines whether the table is a segment-page table. Set this parameter based on the value of **bucket_node** in local_bad_block_info. If the value of **bucket_node** is **-1**, the table is not a segment-page table. In this case, set **is_segment** to **false**. If the value of **bucket_node** is not -1, set **is_segment** to **true**. Segment-page storage is supported.

  Value range: The value is of a Boolean type and can be **true** or **false**.

Return type: record

Examples (Set parameters based on the output of local_bad_block_info. Otherwise, an error is reported.):

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Page-based storage:

```
gaussdb=# SELECT * FROM gs_verify_and_tryrepair_page('base/16552/24745',0,false,false);
node_name        |       path       |  blocknum  |         disk_page_res         | mem_page_res | is_repair
-----------------+------------------+------------+-------------------------------+--------------+----------
dn_6001_6002_6003 | base/16552/24745 |     0      | page verification succeeded.|              | f
```

Segment-page storage:

```
gaussdb=# SELECT * FROM gs_verify_and_tryrepair_page('base/14365/1',4494,false,true);
node_name        |     path      | blocknum |         disk_page_res         | mem_page_res | is_repair
-----------------+---------------+----------+-------------------------------+--------------+-----------
dn_6001_6002_6003 | base/14365/1 |     4494 | page verification succeeded. |              | f
```

- gs_repair_page(path text, blocknum oid, is_segment bool, timeout int)

  Description: Restores the specified page of the instance. This function can be used only by the primary DN that is properly connected to the primary and standby DNs. By default, only the initial user, users with the SYSADMIN permission, and users with the OPRADMIN permission in the O&M mode on the primary DN can view the information. Other users can view the information only after being granted permissions.

  Note: If a page on a DN is damaged, a verification error at the PANIC level is reported when the DN is promoted to primary. The DN cannot be promoted to primary, which is normal.

  Parameters

  – path

    Path of the damaged page. Set this parameter based on the **file_path** column in local_bad_block_info or the **path** column in the gs_verify_and_tryrepair_page function.

    Value range: a string

  – blocknum

    Number of the damaged page. Set this parameter based on the **block_num** column in local_bad_block_info or the **blocknum** column in the gs_verify_and_tryrepair_page function.

    Value range: OID ranging from 0 to 4294967295. Note: A negative value will be forcibly converted to a non-negative integer.

  – is_segment

    Determines whether the table is a segment-page table. The value of this parameter is determined by the value of **bucket_node** in local_bad_block_info. If the value of **bucket_node** is **–1**, the table is not a segment-page table and **is_segment** is set to **false**. If the value of **bucket_node** is not **–1**, **is_segment** is set to **true**. Segment-page storage is supported.

    Value range: The value is of a Boolean type and can be **true** or **false**.

  – timeout

    Duration of waiting for standby DN replay. The page to be repaired needs to wait for the standby DN to be played back to the location of the current primary DN. Set this parameter based on the replay duration of the standby DN.

    Value range: 60s to 3600s.

  Return type: Boolean

  Examples (Set parameters based on the output of local_bad_block_info. Otherwise, an error is reported.):

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Page-based storage:

```
gaussdb=# SELECT * FROM gs_repair_page('base/16552/24745',0,false,60);
result
--------
t
```

Segment-page storage:

```
gaussdb=# SELECT * FROM gs_repair_page('base/16552/1',4494,true,60);
result
--------
t
```

- gs_seg_verify_datafile(IN tablespace_name name, IN database_name name, IN file_id integer, IN bucketnode integer, IN start_block_id bigint default 0, IN end_block_id bigint default UINT32MAX)

  Description: Verifies segment-page files 1 to 5, checks whether the main fork has page damage, and records the verification result in the local_bad_block_info() function. Only users with the SYSADMIN attribute and users with the OPRADMIN attribute in O&M mode can execute this function.

  Parameters

Name	Type	Description
tablespace_name	name	Tablespace name.
database_name	name	Database name.
file_id	integer	File name. The value ranges from 1 to 5.
bucketnode	integer	Bucket node of the table. • **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. Currently, only hash bucket tables and segment-page ordinary tables are supported.
start_block_id	bigint	Start value of the verification page range. The default value is **0**.
end_block_id	bigint	End value of the verification page range. The default value is **4294967295**.

  Return value: none

  Example:

```
gaussdb=# SELECT * FROM gs_seg_verify_datafile('seg_tblspc', 'postgres', 2, 1024);
WARNING:  page verification failed, calculated checksum 60994 but expected 11565, the block num is 4157
 gs_seg_verify_datafile
------------------------


(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_edit_page_bypath(path text, blocknum int64, offset int, data text, data_size int, read_backup bool, storage_type text)

  Description: Transfers the path, block number, offset, target data to be modified, and length of the target table file, modifies the target data to the corresponding fields on the page, and returns the path of the modified file to be flushed to the disk. The **read_backup** column determines the file reading mode, and the **storage_type** column indicates the file storage mode (for example, segment-page storage). To prevent incorrect modification, this function does not directly modify the original page but modifies the copied page and flushes the modified page to the specified path. Only the SYSADMIN or OPRADMIN in O&M mode can execute this function, and the function applies only to the primary node. In a distributed environment, this function must be executed on DNs.

  Return type: text.

  Table 1-262 gs_edit_page_bypath parameters

Category	Parameter	Type	Description
Input parameter	path	text	Physical file path of the file to be modified, which is related to the **read_backup** field. The value can be the relative path of the file in the database directory or the absolute path of files such as the backup file. If the target file does not exist or fails to be read, an error message is displayed. • If **read_backup** is **false**, the path format is *tablespace name*/*database oid*/*table relfilenode* (*physical file name*). For example, **base/16603/16394**. • If **read_backup** is **true**, **path** is a valid path. In this case, because other information about the input file cannot be obtained, you need to ensure that the input data is correct.
Input parameter	blocknum	bigint	Block number of the page to be repaired. Value range: 0 to *MaxBlockNumber*. Reads the page corresponding to the specified physical or logical block number based on the **read_backup** field. If the specified block number is out of range, an error message is returned.
Input parameter	offset	int	In-page offset of the field to be modified. Value range: 0 ≤ **offset** < *BLCKSZ*. If the specified value is less than **0** or greater than or equal to that of **BLCKSZ**, the system view is used to return the corresponding error information.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Input parameter	data	text	Type of the target value to be modified. Type: ● **'0x'**: hexadecimal. ● **'0b'**: binary. ● **'0s'**: character string. Others: If the value of the **data** parameter is not one of the preceding types, the value can only be a decimal numeric string.
Input parameter	data_size	int	Length of the written data, in bytes. Value range: 1 to 8. If the specified write length is less than 1 byte or greater than 8 bytes, or the sum of offset and data_size is greater than the value of **BLCKSZ**, the system view is used to return the corresponding error information.
Input parameter	read_backup	bool	Specifies whether to read pages from the backup directory. If this parameter is set to **false**, the target page is read based on the logical block number. Otherwise, the page is read based on the physical block number.
Input parameter	storage_type	text	Specifies the file storage mode. This parameter is optional. ● **'page'**: page mode. ● **'segment'**: segment-page mode. In page-based mode, only U-page and UB-tree data pages can be edited and modified. In segment-page mode, Astore data pages can be edited and modified. Tables with tablespaces are not supported. Other information about the input file cannot be obtained. Therefore, you need to ensure that the input data type is correct.
Output parameter	output_msg	text	If the modification is successful, the absolute path of the modified file is returned. The modified file is stored in the **gs_log/dump** directory. If the modification fails, a failure message is returned.

In the example, set parameters based on the parameter description and use the actual physical path.

Example 1: Overwrite the data whose value is **0X1FFF** at the offset of 16 bytes on page 0 in the **base/15808/25075** table.

```
gaussdb=# SELECT gs_edit_page_bypath('base/15808/25075',0,16,'0x1FFF', 2, false, 'page');
                gs_edit_page_bypath
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-----------------------------------------------------------------------
 /gs_log_dir/dump/1663_15808_25075_0.editpage
(1 rows)
```

Example 2: If the input parameter does not comply with the specifications, an error message is returned.

```
gaussdb=# SELECT gs_edit_page_bypath('base/15808/25075', 0,16,'@1231!', 8, false, 'page');
           gs_edit_page_bypath
-----------------------------------------
 Error: the parameter 'data' decode failed.
(1 row)
```

Example 3: When the data to be written is the same as the original value, an alarm is returned.

```
gaussdb=# SELECT gs_edit_page_bypath('/gs_log_dir/dump/1663_15808_25075_0.editpage',
0,16,'0x1FFF', 2, true, 'page');
                  gs_edit_page_bypath
-------------------------------------------------------------
 Warning: source buffer is consistent with target buffer.
(1 row)
```

- gs_repair_page_bypath(src_path text, src_blkno int64, dest_path text, dest_blkno int64, storage_type text)

  Description: Transfers the path and page number of the source file, and writes the page to the specified page number of the target file. You can repair pages of the standby node through the primary node or repair the pages of the primary node through the standby node. In addition, you can initialize bad blocks using this function. In a distributed environment, this function must be executed on DNs.

  a. Target pages are overwritten and synchronized to the standby node. In page storage mode, U-heap and UB-tree pages can be modified. The Undo Record page, Undo Slot page, compressed table, and Astore page will be supported later. In segment-page storage mode, Astore pages and hash bucket tables can be modified. System catalog files and data sections cannot be modified.
  b. With this function, you can overwrite target pages during the write operation. Before overwriting, the target page is backed up and flushed to a specified directory. The backup page can be rewritten back to the target page. If an ordinary table is modified on the primary node, a new WAL is generated and synchronized to the standby node. If an ordinary table is modified on the standby node, no WAL is recorded.
  c. The repair view applies only to the primary node or the standby node when the read on standby function is enabled. Only the SYSADMIN or OPRADMIN in O&M mode can use this function. All modifications will be recorded in database logs. In addition, you are advised to enable the audit logging function of system functions before using this function to record audit information.
  d. When the repair view is called on the standby node, the standby node can be repaired only by reading pages from the primary node.
  e. During the repair, the LSNs of the source and target pages must be the same, and the CRC check of the page must be passed. Otherwise, the repair fails. During this process, other repair methods may be triggered.
  f. The target page is verified. If the target page is correct, it cannot be overwritten. Currently, the U-heap and UB-tree pages can be verified.

  Return type: text.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

CAUTION

Calling this system function is a high-risk operation. Exercise caution when performing this operation.

Category	Parameter	Type	Description
Input parameter	src_path	text	Path of the source file. The following types of paths are supported: • Data files and index files: **gs_log/dump/1663_15808_25075_0.editpage**. • **src_path** is set to **'standby'** on the primary node. That is, pages are read from the standby node to repair the primary node. • **src_path** is set to **'primary'** on the standby node. That is, pages are read from the primary node to repair the standby node. On the standby node, **src_path** can only be set to **'primary'** and the read function must be enabled on the standby node before the repair. • **src_path** is set to **init_block** on the primary node to allow skipping bad blocks in extreme scenarios.
Input parameter	src_blkno	bigint	Physical block number of the source page. Value range: 0 to *MaxBlockNumber*.
Input parameter	dest_path	text	Relative path of the target file. For example, **base/15808/25075**.
Input parameter	dest_blkno	bigint	Logical block number of the target page. Value range: 0 to *MaxBlockNumber*.
Input parameter	storage_type	text	Specifies the target file storage mode. This parameter is optional. • **'page'**: page mode. • **'segment'**: segment-page mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	output_msg	text	If the overwrite operation is successful, the backup path of the target page is returned. If the overwrite operation fails, an error message is returned. The format of the flushed file name is *relfilepath_blocknum_timestamp.repairpage*.

Set parameters based on the preceding table and ensure that the physical file exists. If the input parameter is abnormal or the restoration fails, an error is reported.

Example 1: Enter a file in a specified path to overwrite the target file.

```
gaussdb=# SELECT * FROM gs_repair_page_bypath('gs_log/dump/1663_15991_16767_0.editpage', 0, 'base/15991/16767', 0, 'page');
                                        output_msg
-------------------------------------------------------------------------------------------------
/gs_log_dir/dump/1663_15991_16767_0_738039702421788.repairpage
(1 row)
```

Example 2: Read pages from the standby node to repair the primary node.

```
gaussdb=# SELECT * FROM gs_repair_page_bypath('standby', 0, 'base/15990/16768', 0, 'page');
                                        output_msg
-------------------------------------------------------------------------------------------------
/gs_log_dir/dump/1663_15990_16768_0_738040397197907.repairpage
(1 row)
```

Example 3: Read pages from the primary node to repair the standby node.

```
gaussdb=# SELECT * FROM gs_repair_page_bypath('primary', 0, 'base/15990/16768', 0, 'page');
                                        output_msg
-------------------------------------------------------------------------------------------------
 /gs_log_dir/dump/1663_15990_16768_0_738040506157799.repairpage
(1 row)
```

Example 4: Initialize the target page and skip bad blocks.

```
gaussdb=# SELECT * FROM gs_repair_page_bypath('init_block', 0, 'base/15990/16768', 0, 'page');
                                        output_msg
-------------------------------------------------------------------------------------------------
 /gs_log_dir/dump/1663_15990_16768_0_738040768010281.repairpage
(1 row)
```

- gs_edit_repair_page(path text, blocknum int64, area text, index int, name text, value uint64)

  Description: Modifies the XID-related columns on a page based on the input source file path and page number, and directly updates the page in the database. Calling this function to modify a page generates the corresponding Xlogs. Before the modification, the source page and parsed page of the original page are dumped to the log path, and the function returns the path of the dump file.

  The limitations on this function are as follows:

  – It can be called only on the primary DN.

  – Compressed pages cannot be repaired.

  – Segment-page pages cannot be repaired.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Only the initial user, users with the SYSADMIN attribute, and users with the OPRADMIN attribute in O&M mode can call this function.
- Only pages that pass the CRC can be repaired.

Return type: text.

NOTE

This system function is used to handle XID damage and is a high-risk operation. Exercise caution when using this function and do not call it frequently.

Table 1-263 gs_edit_repair_page parameters

Category	Parameter	Category	Description
Input parameter	path	text	Path of the source file. To obtain the relative path of a table file, call pg_relation_filepath(table_name text). To obtain the path of a partitioned table, query the pg_partition system catalog and call pg_partition_filepath(partition_oid). For example, **base/16603/16394**.
Input parameter	blocknum	bigint	Physical block number of the source page. The value ranges from 0 to *MaxBlockNumber*.
Input parameter	area	text	Target area of a page. The options are as follows: • **header**: page header. • **td**: td or ted part of the U-heap page. • **tuple**: page tuple.
Input parameter	index	integer	Index of the target area in the same type of area. The value starts from 0. The header area supports only the index 0.
Input parameter	name	text	Target columns to be repaired. • **header**: **pd_prune_xid**, **pd_xid_base**, and **pd_multi_base** • **td**: **xid** • **tuple(uheap)**: **xid** • **tuple(heap)**: **xmin** and **xmax** • **tuple(ubtree)**: **xmin** and **xmax**
Input parameter	value	bigint	Target value to be repaired.
Output parameter	output_msg	text	If the repair is successful, the dump file path is returned. If the repair fails, an error message is returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Pass the parameters based on the preceding table and ensure that the physical file exists. If the parameter is abnormal or the repair fails, the function reports an error.

Example 1: Modify **pd_prune_xid** in the header on page 0 in the **base/14816/16404** file.

```
gaussdb=# select * from gs_edit_repair_page('base/14816/16404', 0, 'header', 0, 'pd_prune_xid', 0);
                        output_msg
-------------------------------------------------------------------
/gs_log_dir/dump/1663_14816_16404_0_824366317606633.originpage+
/gs_log_dir/dump/1663_14816_16404_0.dump
(1 row)
```

Example 2: Modify **xid** of ted 0 in the **td** part on page 3 in the **base/14816/16404** file.

```
gaussdb=# select * from gs_edit_repair_page('base/14816/16404', 3, 'td', 0, 'xid', 196218);
                        output_msg
-------------------------------------------------------------------
/gs_log_dir/dump/1663_14816_16404_3_824368882816559.originpage+
/gs_log_dir/dump/1663_14816_16404_3.dump
(1 row)
```

Example 3: Modify **xmax** of tuple 16 in the **tuple** part on page 5 in the **base/14816/16387** file.

```
gaussdb=# select * from gs_edit_repair_page('base/14816/16387', 5, 'tuple', 16, 'xmax', 0);
                        output_msg
-------------------------------------------------------------------
/gs_log_dir/dump/1663_14816_16387_5_824368882816559.originpage+
/gs_log_dir/dump/1663_14816_16387_5.dump
(1 row)
```

- gs_repair_undo_byzone(zone_id int)

  Description: Transfers the ID of the undo zone to be repaired, repairs the metadata of the target undo zone, and returns the repair result details. If the undo zone is not repaired, no information is output.

  Return type: record

  Note: Currently, the function can be called only on the primary node. After the repair is successful, the repair will be synchronized to the standby node by recording Xlogs. The caller must be a SYSADMIN or an OPRADMIN in O&M mode. You are advised to enable the audit logging function before using the function to record audit information.

  CAUTION

  Calling this system function is a high-risk operation. Exercise caution when performing this operation.

  Table 1-264 gs_repair_undo_byzone parameters

Category	Parameter	Type	Description
Input parameter	zone_id	int	Undo zone ID. • **-1**: repairs the metadata of all undo zones. • **0** to **1048575**: repairs the metadata of the undo zone corresponding to the zone ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Output parameter	zone_id	int	Undo zone ID.
Output parameter	repair_detail	text	Repair result of the undo zone metadata corresponding to the zone ID. If the repair is successful, "rebuild undo meta succeed." is displayed. If the repair fails, "rebuild undo meta failed." as well as the failure cause is displayed.

Example 1: If the undo zone meta information corresponding to the entered **zone_id** is not damaged, no output is expected.

```
gaussdb=# SELECT * FROM gs_repair_undo_byzone(4);
 zone_id | repair_detail
---------+---------------
(0 rows)
```

Example 2: If the undo zone metadata corresponding to the entered **zone_id** is successfully restored, the system displays a message indicating that the restoration is successful.

```
gaussdb=# SELECT * FROM gs_repair_undo_byzone(78);
 zone_id | repair_detail
---------+---------------
      78 | rebuild undo meta succeed.
(1 row)
```

Example 3: If the undo zone metadata corresponding to the entered zone ID fails to be repaired, the detailed information about the repair failure is displayed.

```
gaussdb=# SELECT * FROM gs_repair_undo_byzone(0);
 zone_id | repair_detail
---------+---------------
       0 | rebuild undo meta failed. try lock undo zone_id failed.
(1 row)
```

NOTE

If the undo zone to be repaired is damaged and the zone ID is occupied by another active thread, calling this repair function will end the thread that occupies the zone ID and forcibly repair the damaged undo zone metadata.

- gs_verify_urq(index_oid oid, partindex_oid oid, blocknum bigint, queue_type text)

  Description: Verifies the correctness of the index recycling queue (potential queue/available queue/single page).

  Parameters: See **Table 1-265**.

  Return type: record

Table 1-265 gs_verify_urq parameters

Category	Parameter	Type	Description
Input parameter	index_oid	oid	UB-tree index OID. • Ordinary index: index OID. • Global index: GPI OID. • Local index: OID of the primary index.
Input parameter	partindex_oid	oid	UB-tree partitioned index OID. • Ordinary index: 0. • Global index: 0. • Local index: OID of the partitioned index (primary or secondary).
Input parameter	blocknum	bigint	Specifies the page number: • If the queue type is single page, the correctness of all tuples of **blocknum** on a single page is verified. The value range is [0,*Queue file size*/8192). • If the queue is empty or free, **blocknum** is an invalid value.
Input parameter	queue_type	text	Specifies the queue type: • **empty queue**: potential queue • **free queue**: available queue • **single page**: single-page queue
Output parameter	error_code	text	Error code
Output parameter	detail	text	Detailed error information and other key information.

When using the example, set parameters based on the parameter description and use the actual OID and **blocknum**. Otherwise, an error is reported.

Example 1:

```
gaussdb=# SELECT * FROM gs_verify_urq(16387, 0, 1, 'free queue');
 error_code | detail
------------+--------
(0 rows)
```

Example 2:

```
gaussdb=# SELECT * FROMgs_verify_urq(16387, 0, 1, 'empty queue');
     error_code       |
detail
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
----------------------
+-------------------------------------------------------------------------------------------------------------------
 VERIFY_URQ_PAGE_ERROR | invalid urq meta: oid 16387, blkno 1, head_blkno = 1, tail_blkno = 3,
nblocks_upper = 4294967295, nblocks_lower = 1; urq_blocks = 6, index_blocks = 12
(1 row)
```

NOTE

Currently, this API supports only Ustore index tables. If the verification of the index recycling queue is normal, the view does not display the error code and error details. Otherwise, the view displays the error code and error details. The error codes include "VERIFY_URQ_PAGE_ERROR", "VERIFY_URQ_LINK_ERROR", "VERIFY_URQ_HEAD_MISSED_ERROR", and "VERIFY_URQ_TAIL_MISSED_ERROR". If any of the preceding error codes is displayed, contact Huawei technical support to locate the fault.

- gs_urq_dump_stat(index_oid oid, partindex_oid oid)

  Description: Queries information about a specified index recycling queue.

  In the return result, **recentGlobalDataXmin** and **globalFrozenXid** are two oldestxmins used by the recycling queue to determine whether the index page can be recycled, **next_xid** is the XID of the next latest transaction, **urq_blocks** indicates the total number of pages in the recycling queue and information about valid pages in the free queue (available queue) and empty queue (potential queue). For PCR indexes, the output recentGlobalDataXmin is replaced by globalRecycleXid that affects the recycling of the PCR index pages.

  Parameters: See **Table 1-266**.

  Table 1-266 gs_urq_dump_stat parameters

Category	Parameter	Type	Description
Input parameter	index_oid	oid	UB-tree index OID. • Ordinary index: index OID. • Global index: GPI OID. • Local index: OID of the primary index.
Input parameter	partindex_oid	oid	UB-tree partitioned index OID. • Ordinary index: 0. • Global index: 0. • Local index: OID of the partitioned index (primary or secondary).
Output parameter	result	text	Detailed statistics about the index recycling queue.

  When using the example, set parameters based on the parameter description and use the actual OID. Otherwise, an error is reported.

  Example (RCR):

```
gaussdb=# SELECT * FROM gs_urq_dump_stat(16387, 0);
                                                result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-----------------------------------------------------------------------------------------------------------------------------------
 urq stat info: recentGlobalDataXmin = 213156, globalFrozenXid = 213156, next_xid = 214157,
urq_blocks = 6,                    +
   free  queue:    head page blkno = 0 min_xid = 211187 max_xid = 214157, tail page blkno = 0
min_xid = 211187 max_xid = 214157,+
               middle page min_xid = 1152921504606846975 max_xid = 0, valid_pages = 1, valid_items =
6, can_use_item = 3       +
   empty queue:    head page blkno = 1 min_xid = 212160 max_xid = 213160, tail page blkno = 3
min_xid = 213162 max_xid = 214156,+
               middle page min_xid = 1152921504606846975 max_xid = 0, valid_pages = 2, valid_items =
999, can_use_item = 498    +

(1 row)
```

Example (PCR):

```
gaussdb=# SELECT gs_urq_dump_stat(17260,0);
                                                    gs_urq_dump_stat
-----------------------------------------------------------------------------------------------------------------------------------
-------------------------
 urq stat info: globalRecycleXid = 22113, globalFrozenXid = 22107, next_xid = 22116, urq_blocks =
6,                                 +
   free  queue:    head page blkno = 0 min_xid = 1152921504606846975 max_xid = 0, tail page blkno
= 0 min_xid = 1152921504606846975 max_xid = 0,+
               middle page min_xid = 1152921504606846975 max_xid = 0, valid_pages = 1, valid_items =
0, can_use_item = 0                +
   empty queue:    head page blkno = 1 min_xid = 1152921504606846975 max_xid = 0, tail page blkno
= 1 min_xid = 1152921504606846975 max_xid = 0,+
               middle page min_xid = 1152921504606846975 max_xid = 0, valid_pages = 1, valid_items =
0, can_use_item = 0                +

(1 row)
```

NOTE

Currently, this API supports only Ustore index tables.

- gs_repair_urq(index_oid oid, partindex_oid oid)

  Description: Repairs (with loss) index recycling queues (potential and available queues). The recycling queue file of the current index is deleted and an empty recycling queue file is created. If the repair is successful, "reinitial the recycle queue of index relation successfully" is displayed.

  Parameters: See **Table 1-267**.

  Note: The current function can be called only on the primary node.

Table 1-267 gs_repair_urq parameters

Category	Parameter	Type	Description
Input parameter	index_oid	oid	UB-tree index OID. • Ordinary index: index OID. • Global index: GPI OID. • Local index: OID of the primary index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Input parameter	partindex_oid	oid	UB-tree partitioned index OID. • Ordinary index: 0. • Global index: 0. • Local index: OID of the partitioned index (primary or secondary).
Output parameter	result	text	When the rebuilding is successful, "reinitial the recycle queue of index relation successfully" is displayed. Otherwise, the rebuilding fails.

When using the example, set parameters based on the parameter description and use the actual OID. Otherwise, an error is reported.

Example:

```
gaussdb=# SELECT * FROM gs_repair_urq(16387, 0);
                            result
-------------------------------------------------------------
 reinitial the recycle queue of index relation successfully.
(1 row)
```

NOTE

Currently, this API supports only Ustore index tables.

- gs_check_table(table_oid oid, partition_oid oid, index_oid oid, partindex_oid oid, start_blk bigint, end_blk bigint, duration text, need_repair bool, mode text)

  Description: Verifies reverse data consistency between heap tables and indexes.

  Constraints: The Astore and Ustore are supported. MOT storage engines are not supported. The verification of standby nodes is supported, but the verification of the logger node is not supported. The verification of common row-store tables, ILM compressed tables, TDE tables, and hash bucket tables is supported by heap tables. The verification of temporary tables, global temporary tables, and unlogged tables is not supported by heap tables. Indexes support B-tree and UB-tree index verification and are compatible with segment-page storage.

  Parameters: For details, see **Table 1-268**.

  Return type: record

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-268 gs_check_table parameters

Category	Parameter	Type	Description
Input parameter	table_oid	oid	Heap table OID. • Ordinary table: heap table OID. • Partitioned table: main table OID.
Input parameter	partition_oid	oid	Partition OID. • Ordinary table: **0**. • Partitioned table: partition OID. • All partitions scanned in the partitioned table: **0**.
Input parameter	index_oid	oid	Index OID. • Ordinary index: index OID. • Global index: GPI OID. • Local index: OID of the primary index. • All scanned indexes: **0**.
Input parameter	partindex_oid	oid	Partitioned index OID. • Ordinary index: 0. • Global index: 0. • Local index: OID of the partitioned index (primary or secondary). • Local index of all scanned partitions: **0**.
Input parameter	start_blk	bigint	Number of the verification start page. If this parameter and **end_blk** are set to **-1**, the verification starts from the beginning by default. When all partitions in a partitioned table are scanned, only the verification from the beginning is supported. The values of **start_blk** and **end_blk** must be **-1**.
Input parameter	end_blk	bigint	Number of the verification end page.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Input parameter	duration	text	Verification duration. The value can be in millisecond, second, minute, hour, or day, for example, **500ms**, **60s**, **10min**, **2h**, or **1d**. The maximum value is **24h**. There may be a millisecond-level, seconds-level, or minute-level error. If this parameter is set to **0**, the verification is complete by default. The verification duration is the minimum value of the actual execution time and the configured time. If the execution time exceeds the configured time, the verification stops and the current verification position is recorded. The next scanning can be resumed from the recorded position.
Input parameter	need_repair	bool	Specifies whether to automatically repair inconsistency. That is, add or delete index tuples based on heap tables. Currently, this parameter can only be set to **false**. This parameter is a pre-embedded parameter. This version does not support automatic repair.
Input parameter	mode	text	Pre-embedded parameter of the validation mode. The value can be **increment** or **full**. Resumable verification, increment verification, and full verification are supported.
Output parameter	error_code	int	Error code.
Output parameter	output_filepath	text	Relative path for flushing parsed data to disks. The path is the **gs_log/dump** dictionary. The flushed file name is **dbverify**_*Error code_Heap table OID_Timestamp*.**log**. The following is an example: /*$GAUSSLOG*/**gs_log/dump/dbverify_0_16384_123456.log**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

When using the example, set parameters based on the parameter description and use the actual and matching OID and **blocknum**. Otherwise, an error is reported.

Example 1:

```
gaussdb=# select * from gs_check_table((select oid from pg_class where relname='t2'),0,0,0,-1,-1, '1h',
false, 'full');
 verify_code |                                output_filepath
-------------+-------------------------------------------------------------------------------------
           0 | /$GAUSSLOG/gs_log/dump/dbverify_0_17279_766073191583799.log
(1 row)
```

Example 2:

```
gaussdb=# select * from gs_check_table((select oid from pg_class where relname='t1'),0,(select oid
from pg_class where relname='index_t1'),0,2,10, '2s', false, 'full');
 verify_code |                                output_filepath
-------------+-------------------------------------------------------------------------------------
           0 | /$GAUSSLOG/gs_log/dump/dbverify_0_17168_766073338594117.log
(1 row)
```

NOTE

Calling this system function is a high-risk operation. Exercise caution when performing this operation.

Currently, this API supports consistency check between Astore and Ustore tables and between B-tree and UB-tree indexes. If no consistency problem is found after the verification, the error code is **0**. If an error code other than **0** is displayed, contact Huawei technical support to locate the fault.

Resumable verification and increment verification:

The information related to resumable verification and incremental verification is not persisted. After the database process is restarted, the previous verification is lost. When the verification is performed again, the full verification is performed from the beginning.

If the verification mode is **increment** and the last verification on the table and index is not complete, resumable verification is triggered and the verification starts from the position where the last verification is interrupted.

If the verification mode is **increment** and the last verification on the table and index is complete, increment verification is triggered to verify the incremental modification page.

If the verification mode is **full**, full verification is performed from the beginning.

I/O flow control:

By default, **io_priority** is set to a low priority for I/O flow control.

Use permission:

Only the initial user, users with the SYSADMIN, OPRADMIN, or MONADMIN permission can execute verification system functions. Only data tables or indexes in the current database can be verified.

Recommended application scenario:

You are advised to perform the verification during off-peak hours to ensure that most data is static data during the verification and ensure the integrity and accuracy of the verification.

- gs_check_index(index_oid oid, partindex_oid oid, duration text, need_repair bool, mode text)

  Description: Verifies forward data consistency between heap tables and indexes.

  Constraints: The Astore and Ustore are supported. MOT storage engines are not supported. The verification of standby nodes is supported, but the verification of the logger node is not supported. The verification of common

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

row-store tables, ILM compressed tables, TDE tables, and hash bucket tables is supported by heap tables. The verification of temporary tables, global temporary tables, and unlogged tables is not supported by heap tables. Indexes support B-tree and UB-tree index verification and are compatible with segment-page storage.

Parameters: For details, see **Table 1-269**.

Return type: record

Table 1-269 gs_check_index parameters

Category	Parameter	Type	Description
Input parameter	index_oid	oid	Index OID. • Ordinary index: index OID. • Global index: GPI OID. • Local index: OID of the primary index.
Input parameter	partindex_oid	oid	Partitioned index OID. • Ordinary index: 0. • Global index: 0. • Local index: OID of the partitioned index (primary or secondary). • Local index of all scanned partitions: **0**.
Input parameter	duration	text	Verification duration. The value can be in millisecond, second, minute, hour, or day, for example, **500ms**, **60s**, **10min**, **2h**, or **1d**. The maximum value is **24h**. There may be a millisecond-level, seconds-level, or minute-level error. If this parameter is set to **0**, the verification is complete by default. The verification duration is the minimum value of the actual execution time and the configured time. If the execution time exceeds the configured time, the verification stops and the current verification position is recorded. The next scanning can be resumed from the recorded position.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Parameter	Type	Description
Input parameter	need_repair	bool	Specifies whether to automatically repair inconsistency. That is, add or delete index tuples based on heap tables. Currently, this parameter can only be set to **false**. This parameter is a pre-embedded parameter. This version does not support automatic repair.
Input parameter	mode	text	Pre-embedded parameter of the validation mode. The value can be **increment** or **full**. Resumable verification, increment verification, and full verification are supported.
Output parameter	error_code	int	Error code.
Output parameter	output_filepath	text	Relative path for flushing parsed data to disks. The path is the **gs_log/dump** dictionary. The flushed file name is **dbverify**_*Error code_Heap table OID_Timestamp*.**log**. The following is an example: /*$GAUSSLOG*/**gs_log/dump/dbverify_0_16384_123456.log**

When using the example, set parameters based on the parameter description and use the actual and matching OID. Otherwise, an error is reported.

Example 1:

```
gaussdb=# select * from gs_check_index((select oid from pg_class where relname='index_t1'),0, '1h',
false, 'full');
 verify_code |                                output_filepath
-------------+-------------------------------------------------------------------------------------
           0 | /$GAUSSLOG/gaussdb/single_node/gs_log/dump/dbverify_0_17302_766074761373434.log
(1 row)
```

Example 2:

```
gaussdb=# select * from gs_check_index((select oid from pg_class where relname='index_range_2'),
(select oid from pg_partition where parentid=(select oid from pg_class where
relname='index_range_2')and relname='p1_a_idx'), '10s', false, 'full');
 verify_code |                                output_filepath
-------------+-------------------------------------------------------------------------------------
           0 | /$GAUSSLOG/gaussdb/single_node/gs_log/dump/dbverify_0_17306_766074834795344.log
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

Currently, this API supports consistency check between Astore and Ustore tables and between B-tree and UB-tree indexes. If no consistency problem is found after the verification, the error code is **0**. If an error code other than **0** is displayed, contact Huawei technical support to locate the fault.

Resumable verification and increment verification:

The information related to resumable verification and incremental verification is not persisted. After the database process is restarted, the previous verification is lost. When the verification is performed again, the full verification is performed from the beginning.

If the verification mode is **increment** and the last verification on the table and index is not complete, resumable verification is triggered and the verification starts from the position where the last verification is interrupted.

If the verification mode is **increment** and the last verification on the table and index is complete, increment verification is triggered to verify the incremental modification page.

If the verification mode is **full**, full verification is performed from the beginning.

I/O flow control:

By default, **io_priority** is set to a low priority for I/O flow control.

Use permission:

Only the initial user, users with the SYSADMIN, OPRADMIN, or MONADMIN permission can execute verification system functions. Only data tables or indexes in the current database can be verified.

Recommended application scenario:

You are advised to perform the verification during off-peak hours to ensure that most data is static data during the verification and ensure the integrity and accuracy of the verification.

CAUTION

Calling this system function is a high-risk operation. Exercise caution when performing this operation.

- gs_get_standby_bad_block_info()

  Description: Displays the pages that have been detected on the standby node but have not been repaired. By default, only the initial user, users with the SYSADMIN permission, users with the OPRADMIN permission in the O&M mode, and users with the MONADMIN permission on the standby DN can view the information. Other users can view the information only after being granted permissions. There are four return values in the **invalid_type** column: **NOT_PRESENT** (the page does not exist), **NOT_INITIALIZED** (the page initialization fails), **LSN_CHECK_ERROR** (the LSN check fails), and **CRC_CHECK_ERROR** (the CRC verification fails).

  Return type: record

  Example (If no page is detected but not repaired, no line is displayed.)

```
gaussdb=# SELECT * FROM gs_get_standby_bad_block_info();
 spc_node | db_node | rel_node | bucket_node | fork_num | block_num |  invalid_type   |
master_page_lsn
----------+---------+----------+-------------+----------+-----------+-----------------+-----------------
   1663   |   16552 |   24745  |      -1     |    0     |     0     | CRC_CHECK_ERROR | 0/B2009E8
(1 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.6.43 Multixact Recycling Function

- gs_vacuum_multixact()

  Description: Recycles data files in **offsets** and **members** in the **pg_multixact** directory.

  Return type: Boolean

  - If a file is successfully recycled, **true** is returned.
  - If no file can be recycled, **false** is returned.

  The limitations on this function are as follows:

  - This function can be executed only on the primary DN or on the corresponding primary DN using EXECUTE DIRECT.
  - Only the initial user or users with the SYSADMIN attribute can call this function.
  - Manual recycling cannot be performed on CNs.

  Example 1: If no file can be recycled, **false** is returned.

```
gaussdb=# SELECT gs_vacuum_multixact();
 gs_vacuum_multixact
---------------------
 f
(1 row)
```

  Example 2: If the file is successfully recycled, **true** is returned.

```
gaussdb=# SELECT gs_vacuum_multixact();
 gs_vacuum_multixact
---------------------
 t
(1 row)
```

  NOTE

  The logic for determining whether a file can be recycled is calculated based on **datminmxid** in the PG_DATABASE system catalog. In some scenarios, **datminmxid** may not be updated. For example, a global temporary table exists in the current database. The data in a global temporary table is isolated at the session level. Therefore, the VACUUM background thread cannot access the information about the global temporary table. As a result, the **relminmxid** column of the global temporary table cannot be automatically updated, and **datminmxid** cannot be updated. Therefore, you need to run the **VACUUM FREEZE** *Table name***;** command on all global temporary tables in the session of the global temporary table to update **relminmxid** of the current global temporary table.

1.6.44 Functions of the XML Type

NOTE

- In all the following XML functions, when the GUC parameter **xmloption** is set to **content**, the value of **encoding** in the XML declaration can be **ZHS16GBK**. When the GUC parameter **xmloption** is set to **document**, the value of **encoding** in the XML declaration cannot be **ZHS16GBK**. If **encoding** is set to **ZHS16GBK**, an error is reported.
- XML functions support a maximum of 256 nested layers in XML files.

- xmlparse ({ DOCUMENT | CONTENT } value [wellformed])

  Description: Generates XML values from character data.

  Parameter: data of the TEXT type

  Return type: XML

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT XMLPARSE (DOCUMENT '<?xml version="1.0"?><book><title>Manual</
title><chapter>...</chapter></book>');
                  xmlparse
-----------------------------------------------------------
 <book><title>Manual</title><chapter>...</chapter></book>
(1 row)
gaussdb=# SELECT XMLPARSE (CONTENT 'abc<foo>bar</foo><bar>foo</bar>');
         xmlparse
---------------------------------
 abc<foo>bar</foo><bar>foo</bar>
(1 row)
gaussdb=# SELECT XMLPARSE (CONTENT 'abc<foo>bar</foo' wellformed);
    xmlparse
------------------
 abc<foo>bar</foo
(1 row)
```

- xmlserialize({ DOCUMENT | CONTENT } value AS type)

  Description: Generates a string from an XML file.

  Parameter: data of the XML type

  Return type: The data type can be character, character varying, text, or any variant of them.

  Example:

  ```
  gaussdb=# SELECT xmlserialize(CONTENT 'good' AS CHAR(10));
   xmlserialize
  --------------
   good
  (1 row)
  gaussdb=# SELECT xmlserialize(DOCUMENT '<head>bad</head>' as text);
    xmlserialize
  ------------------
   <head>bad</head>
  (1 row)
  ```

  NOTE

  If a string value is converted to XML without using the xmlparse or xmlserialize function, the **XML OPTION** session parameter determines the value, **DOCUMENT** or **CONTENT**. The **XML OPTION** session parameter can be set by the standard command.

  ```
  SET XML OPTION { DOCUMENT | CONTENT };
  ```

  Or use similar syntax to set this parameter.

  ```
  SET xmloption TO { DOCUMENT | CONTENT };
  ```

- xmlcomment(text)

  Description: Creates an XML value that contains an XML comment with the specified text as the content. The text does not contain the "--" character and does not end with a "-" character. Besides, the text should meet the format requirements of XML comments. If the parameter is empty, the result is also empty.

  Parameter: data of the TEXT type

  Return type: XML

  Example:

  ```
  gaussdb=# SELECT xmlcomment('hello');
    xmlcomment
  --------------
   <!--hello-->
  ```

- xmlconcat(xml[, ...])

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Concatenates a list of single XML values into a single value that contains an XML content fragment. Null values are ignored, and the result is null only when all parameters are null. In ORA-compatible mode, you can set **a_format_version** to **10c** and **a_format_dev_version** to **s2** to check whether the input segment is well-formed XML text.

Parameter: data of the XML type.

Return type: XML

Example 1:

```
gaussdb=# SET xmloption=content;
SET
gaussdb=# SELECT XMLCONCAT(('<?xml version="1.0" encoding="GB2312" standalone="no"?
><bar>foo</bar>'),('<?xml version="1.0" encoding="GB2312" standalone="no" ?><bar>foo</bar>')) ;
                        xmlconcat
----------------------------------------------------------------------
<?xml version="1.0" standalone="no"?><bar>foo</bar><bar>foo</bar>
(1 row)
gaussdb=# SELECT XMLCONCAT('abc>');
 xmlconcat
-----------
 abc>
(1 row)
```

Example 2: Syntax of the ORA-compatible database

```
gaussdb=# CREATE DATABASE gaussdb_ora DBCOMPATIBILITY='ORA';
CREATE DATABASE
gaussdb=# \c gaussdb_ora
gaussdb_ora=# SET a_format_version='10c';
SET
gaussdb_ora=# SET a_format_dev_version=s2;
SET
gaussdb_ora=# SET xmloption=content;
SET
gaussdb_ora=# SELECT XMLCONCAT(('<?xml version="1.0" encoding="GB2312" standalone="no"?
><bar>foo</bar>'),('<?xml version="1.0" encoding="GB2312" standalone="no" ?><bar>foo</bar>')) ;
                        xmlconcat
----------------------------------------------------------------------
 <?xml version="1.0" standalone="no"?><bar>foo</bar><bar>foo</bar>
(1 row)

gaussdb_ora=# SELECT XMLCONCAT('abc>');
ERROR:  invalid XML document
DETAIL:  line 1: Start tag expected, '<' not found
abc>
^
CONTEXT:  referenced column: xmlconcat
gaussdb_ora=# \c postgres
gaussdb=# DROP DATABASE gaussdb_ora;
DROP DATABASE
```

NOTE

In ORA-compatible database, if **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s2**, and the **encoding** attribute in the XML declaration is set to **ZHS16GBK**, the XMLCONCAT function reports an error.

- xmlagg(xml [order_by_clause])

  Description: Concatenates the input values called by the aggregate function. Cross-line concatenation is supported. For details about **order_by_clause**, see **SELECT**. In ORA-compatible database mode, you can set **a_format_version** to **10c** and **a_format_dev_version** to **s2**. The **xmloption** parameter of the database is set to **content** by default. When **xmloption** is set to **document**, newline characters are used to concatenate multiple XML lines. If the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

encoding attribute value in the XML declaration is not the default UTF-8, the aggregation result contains the XML declaration.

Parameter: XML

Return type: XML

Example 1:

```
gaussdb=# CREATE TABLE xmltest (
           id int,
           data xml
          );
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'id' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
gaussdb=# INSERT INTO xmltest VALUES (1, '<value>one</value>');
INSERT 0 1
gaussdb=# INSERT INTO xmltest VALUES (2, '<value>two</value>');
INSERT 0 1
gaussdb=# SELECT xmlagg(data) FROM xmltest;
              xmlagg
--------------------------------------
 <value>one</value><value>two</value>
(1 row)
```

Example 2: Syntax of the ORA-compatible database

```
gaussdb=# SET xmloption=document;
SET
gaussdb=# SELECT xmlagg(data) FROM xmltest;
      xmlagg
--------------------
 <value>one</value>+
 <value>two</value>
(1 row)
gaussdb=# DELETE FROM XMLTEST;
DELETE 2
gaussdb=# INSERT INTO xmltest VALUES (1, '<?xml version="1.0" encoding="GBK"?><value>one</
value>');
INSERT 0 1
gaussdb=# INSERT INTO xmltest VALUES (2, '<?xml version="1.0" encoding="GBK"?><value>two</
value>');
INSERT 0 1
gaussdb=# SELECT xmlagg(data) FROM xmltest;
                       xmlagg
------------------------------------------------------------
 <?xml version="1.0" encoding="GBK"?><value>one</value>+
 <value>two</value>
(1 row)
gaussdb=# SELECT xmlagg(data order by id desc) FROM xmltest;
                       xmlagg
------------------------------------------------------------
 <?xml version="1.0" encoding="GBK"?><value>two</value>+
 <value>one</value>
(1 row)

gaussdb=# DROP TABLE xmltest;
```

NOTE

Both the input and return types of the xmlagg function are XML. The maximum size of a single XML data type is 1 GB. Therefore, xmlagg does not support scenarios where the aggregation result exceeds 1 GB.

- xmlelement([ENTITYESCAPING | NOENTITYESCAPING] { [NAME] element_name | EVALNAME element_name } [, xmlattributes([ENTITYESCAPING | NOENTITYESCAPING] value [[AS] attname | AS EVALNAME attname] [, ...])] [, content [[AS] alias] [, ...]])

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Generates an XML element with the given name, attribute, and content.

Return type: XML

Example:

```
gaussdb=# SELECT xmlelement(name foo);
 xmlelement
------------
 <foo/>

-- In ORA-compatible mode:
gaussdb=# CREATE DATABASE gaussdb_ora DBCOMPATIBILITY='ORA';
CREATE DATABASE
gaussdb=# \c gaussdb_ora
gaussdb_ora=# SET a_format_version='10c';
SET
gaussdb_ora=# SET a_format_dev_version=s2;
SET
-- If the keyword ENTITYESCAPING is not set in XMLElement by default or is set, the reserved
characters in the content of XMLElement are escaped.
gaussdb_ora=# SELECT xmlelement("entityescaping<>", 'a$><&"b');
                      xmlelement
-------------------------------------------------------------
 <entityescaping<>>a$&gt;&lt;&amp;&quot;b</entityescaping<>>
(1 row)

gaussdb_ora=# SELECT xmlelement(entityescaping "entityescaping<>", 'a$><&"b');
                      xmlelement
-------------------------------------------------------------
 <entityescaping<>>a$&gt;&lt;&amp;&quot;b</entityescaping<>>
(1 row)

-- When the keyword NOENTITYESCAPING is set in XMLElement, the reserved characters in the
content of XMLElement will not be escaped.
gaussdb_ora=# SELECT xmlelement(noentityescaping "entityescaping<>", 'a$><&"b');
                xmlelement
-------------------------------------------
 <entityescaping<>>a$><&"b</entityescaping<>>
(1 row)

-- When [AS] alias is used to declare an alias for the content in XMLElement, the content value type
must be XML.
gaussdb_ora=# SELECT xmlelement("entityescaping<>", '<abc/>' b);
ERROR:  argument of XMLELEMENT must be type xml, not type unknown
LINE 1: SELECT xmlelement("entityescaping<>", '<abc/>' b);
                                              ^
CONTEXT:  referenced column: xmlelement

gaussdb_ora=# SELECT xmlelement("entityescaping<>", '<abc/>' as b);
ERROR:  argument of XMLELEMENT must be type xml, not type unknown
LINE 1: SELECT xmlelement("entityescaping<>", '<abc/>' as b);
                                              ^
CONTEXT:  referenced column: xmlelement

gaussdb_ora=# SELECT xmlelement("entityescaping<>", xml('<abc/>') b);
               xmlelement
-----------------------------------------
 <entityescaping<>><abc/></entityescaping<>>
(1 row)

gaussdb_ora=# SELECT xmlelement("entityescaping<>", xml('<abc/>') as b);
               xmlelement
-----------------------------------------
 <entityescaping<>><abc/></entityescaping<>>
(1 row)

-- If the keyword ENTITYESCAPING is not set in XMLAttributes by default or is set, the reserved
characters in XMLAttributes are escaped.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb_ora=# SELECT xmlelement("entityescaping<>", xmlattributes('entityescaping<>'
"entityescaping<>"));
                       xmlelement
------------------------------------------------------------
 <entityescaping<> entityescaping<>="entityescaping&lt;&gt;"/>
(1 row)

gaussdb_ora=# SELECT xmlelement(name "entityescaping<>", xmlattributes(entityescaping
'entityescaping<>' "entityescaping<>"));
                       xmlelement
------------------------------------------------------------
 <entityescaping<> entityescaping<>="entityescaping&lt;&gt;"/>
(1 row)

-- When the NOENTITYESCAPING keyword is set in XMLAttributes, the reserved characters in
XMLAttributes will not be escaped.
gaussdb_ora=# SELECT xmlelement("entityescaping<>", xmlattributes(noentityescaping
'entityescaping<>' "entityescaping<>"));
                     xmlelement
------------------------------------------------------------
 <entityescaping<> entityescaping<>="entityescaping<>"/>
(1 row)

gaussdb_ora=# \c postgres
gaussdb=# DROP DATABASE gaussdb_ora;
DROP DATABASE
```

NOTE

- For **xmlelement** and **xmlattributes**, when the value of **name** is **NULL**, the database behavior is different from that of the O database. When the **name** field of **xmlelement** is set to **NULL**, the name information is empty and the attribute information is not displayed. When the **name** field of **xmlattributes** is set to **NULL**, the attribute information is not displayed.
- After the following two parameters are set, the content escape rule of **xmlelement** is ORA compatible. If the two parameters are not set, the content escape rule of **xmlelement** is PG compatible.

  ```
  SET a_format_version='10c';
  SET a_format_dev_version=s2;
  ```

- xmlforest(content [AS name] [, ...])

  Description: Generates an XML sequence of elements using the given name and content.

  Return type: XML

  Example:

  ```
  gaussdb=# SELECT xmlforest('abc' AS foo, 123 AS bar);
          xmlforest
  ------------------------------
   <foo>abc</foo><bar>123</bar>
  ```

- xmlpi(name target [, content])

  Description: Creates an XML processing instruction. If the content is not empty, the content cannot contain character collations.

  Return type: XML

  Example:

  ```
  gaussdb=# SELECT xmlpi(name php, 'echo "hello world";');
            xmlpi
  -----------------------------
   <?php echo "hello world";?>
  ```

- xmlroot(xml, version text | no value [, standalone yes|no|no value])

  Description: Modifies the attributes of the root node of an XML value. If a version is specified, it replaces the value in the version declaration of the root

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

node. If a standalone property is specified, it replaces the value of standalone declaration in the root node.

Example:

```
gaussdb=# SELECT xmlroot('<?xml version="1.1"?><content>abc</content>',version '1.0', standalone
yes);
                        xmlroot
-------------------------------------------------------------
 <?xml version="1.0" standalone="yes"?><content>abc</content>
(1 row)
```

- xmlexists(text passing [BY REF] xml [BY REF])

  Description: Evaluates an XPath 1.0 expression (the first parameter) with the passed XML value as its context item. If the evaluation result generates an empty set of nodes, the function returns **false**. If any other value is generated, the function returns **true**. If the value of any parameter is null, the function returns **Null**. The non-null value passed as a context item must be an XML document, not a content fragment or any non-XML value.

  Parameter: XML

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT xmlexists('//town[text() = ''Toronto'']' PASSING BY REF '<towns><town>Toronto</
  town><town>Ottawa</town></towns>');
   xmlexists
  ------------
   t
  (1 row)
  ```

- xml_is_well_formed(text)

  Description: Checks whether the text is in the correct XML format and returns a Boolean value.

  Parameter: text

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT xml_is_well_formed('<>');
   xml_is_well_formed
  --------------------
   f
  (1 row)
  ```

- xml_is_well_formed_document(text)

  Description: Checks whether the text is a well-formed XML document and returns a Boolean value.

  Parameter: text

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT xml_is_well_formed_document('<pg:foo xmlns:pg="http://postgresql.org/
  stuff">bar</pg:foo>');
   xml_is_well_formed_document
  -----------------------------
   t
  (1 row)
  ```

- xml_is_well_formed_content(text)

  Description: Checks whether the text is well-formed XML content and returns a Boolean value.

  Parameter: text

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT xml_is_well_formed_content('k');
 xml_is_well_formed_content
----------------------------
 t
(1 row)
```

- xpath(xpath, xml [, nsarray])

  Description: Calculates an XPath 1.0 expression, for example, xpath (a text value), on XML data. It returns an array of XML values corresponding to the node set generated by the XPath expression. If the XPath expression returns a scalar value rather than a node set, a single-element array is returned.

  The second parameter must be a well-formed XML document. It must have a single root node element.

  The optional third parameter of the function is an array of namespace mappings. This array should be a two-dimensional text array with the length of the second axis being equal to 2 (that is, it should be an array of arrays, each of which consists of exactly 2 elements). The first element of each array entry is the namespace name (alias), and the second element is the namespace URI. It is not required that aliases provided in this array be the same as those being used in the XML document itself (that is, both in the XML document and in the XPath function context, aliases are local).

  Return type: xml[].

  Example:

  ```
  gaussdb=# SELECT xpath('/my:a/text()', '<my:a xmlns:my="http://example.com">test</
  my:a>',ARRAY[ARRAY['my', 'http://example.com']]);
   xpath
  --------
   {test}
  (1 row)
  ```

- xpath_exists(xpath, xml [, nsarray])

  Description: This function is a special form of the xpath function. It returns a Boolean indicating whether the query is satisfied or not (that is, whether it produces any value other than an empty node set), instead of returning the individual XML values that satisfy the XPath 1.0 expression. This function is equivalent to the standard XMLEXISTS predicate, but it also provides support for a namespace mapping parameter.

  Return type: Boolean

  Example:

  ```
  gaussdb=# SELECT xpath_exists('/my:a/text()', '<my:a xmlns:my="http://example.com">test</
  my:a>',ARRAY[ARRAY['my', 'http://example.com']]);
   xpath_exists
  --------------
   t
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

The following XML function examples show the data you need to prepare:

```
gaussdb=# CREATE SCHEMA testxmlschema;
CREATE SCHEMA
gaussdb=# CREATE TABLE testxmlschema.test1 (a int, b text);
CREATE TABLE
gaussdb=# INSERT INTO testxmlschema.test1 VALUES (1, 'one'), (2, 'two'), (-1, null);
INSERT 0 3
gaussdb=# CREATE DATABASE test;
CREATE DATABASE

-- After the example is executed, run the following commands to delete the preceding data:
gaussdb=# DROP DATABASE test;
DROP DATABASE
gaussdb=# DROP TABLE testxmlschema.test1;
DROP TABLE
gaussdb=# DROP SCHEMA testxmlschema;
DROP SCHEMA
```

- query_to_xml(query text, nulls boolean, tableforest boolean, targetns text)

  Description: This function maps the contents of a query to an XML schema document.

  Return type: XML

  Example:

```
gaussdb=# SELECT query_to_xml('SELECT * FROM testxmlschema.test1', false, false, '');
                    query_to_xml
------------------------------------------------------------
 <table xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">+
                                                            +
 <row>                                                      +
   <a>1</a>                                                 +
   <b>one</b>                                               +
 </row>                                                     +
                                                            +
 <row>                                                      +
   <a>2</a>                                                 +
   <b>two</b>                                               +
 </row>                                                     +
                                                            +
 <row>                                                      +
   <a>-1</a>                                                +
 </row>                                                     +
                                                            +
 </table>                                                   +
(1 row)
```

- query_to_xmlschema(query text, nulls boolean, tableforest boolean, targetns text)

  Description: This function maps the contents of a query to an XML document and an XML schema document, and joins the two documents together.

  Return type: XML

  Example:

```
gaussdb=# SELECT query_to_xmlschema('SELECT * FROM testxmlschema.test1', false, false, '');
                                query_to_xmlschema
-----------------------------------------------------------------------------------------------
 <xsd:schema                                                                  +
     xmlns:xsd="http://www.w3.org/2001/XMLSchema">                                    +
                                                                         +
 <xsd:simpleType name="INTEGER">                                              +
   <xsd:restriction base="xsd:int">                                              +
     <xsd:maxInclusive value="2147483647"/>                                       +
     <xsd:minInclusive value="-2147483648"/>                                      +
   </xsd:restriction>                                                         +
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
</xsd:simpleType>                                                    +
                                                                     +
<xsd:simpleType name="UDT.regression.pg_catalog.text">                 +
  <xsd:restriction base="xsd:string">                                +
  </xsd:restriction>                                                 +
</xsd:simpleType>                                                    +
                                                                     +
<xsd:complexType name="RowType">                                     +
  <xsd:sequence>                                                     +
    <xsd:element name="a" type="INTEGER" minOccurs="0"></xsd:element>  +
    <xsd:element name="b" type="UDT.regression.pg_catalog.text" minOccurs="0"></xsd:element>+
  </xsd:sequence>                                                    +
</xsd:complexType>                                                   +
                                                                     +
<xsd:complexType name="TableType">                                   +
  <xsd:sequence>                                                     +
    <xsd:element name="row" type="RowType" minOccurs="0" maxOccurs="unbounded"/>  +
  </xsd:sequence>                                                    +
</xsd:complexType>                                                   +
                                                                     +
<xsd:element name="table" type="TableType"/>                         +
                                                                     +
</xsd:schema>
(1 row)
```

- query_to_xml_and_xmlschema(query text, nulls boolean, tableforest boolean, targetns text)

  Description: This function maps the contents of a query to an XML document and an XML schema document, and joins the two documents together.

  Return type: XML

  Example:

```
gaussdb=# SELECT query_to_xml_and_xmlschema('SELECT * FROM testxmlschema.test1', true, true, '');
                                query_to_xml_and_xmlschema
-------------------------------------------------------------------------------------------------
<xsd:schema                                                          +
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">                      +
                                                                     +
<xsd:simpleType name="INTEGER">                                      +
  <xsd:restriction base="xsd:int">                                   +
    <xsd:maxInclusive value="2147483647"/>                           +
    <xsd:minInclusive value="-2147483648"/>                          +
  </xsd:restriction>                                                 +
</xsd:simpleType>                                                    +
                                                                     +
<xsd:simpleType name="UDT.regression.pg_catalog.text">                 +
  <xsd:restriction base="xsd:string">                                +
  </xsd:restriction>                                                 +
</xsd:simpleType>                                                    +
                                                                     +
<xsd:complexType name="RowType">                                     +
  <xsd:sequence>                                                     +
    <xsd:element name="a" type="INTEGER" nillable="true"></xsd:element>  +
    <xsd:element name="b" type="UDT.regression.pg_catalog.text" nillable="true"></xsd:element>+
  </xsd:sequence>                                                    +
</xsd:complexType>                                                   +
                                                                     +
<xsd:element name="row" type="RowType"/>                             +
                                                                     +
</xsd:schema>                                                        +
                                                                     +
<row xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">            +
                                                                     +
  <a>1</a>                                                           +
  <b>one</b>                                                         +
</row>                                                               +
                                                                     +
<row xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">            +
```

```
                                                                +
  <a>2</a>                                                        +
  <b>two</b>                                                       +
</row>                                                           +
                                                                +
<row xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">                          +
                                                                +
  <a>-1</a>                                                       +
  <b xsi:nil="true"/>                                              +
</row>                                                           +
                                                                +
(1 row)
```

- cursor_to_xml(cursor refcursor, count int, nulls boolean,tableforest boolean, targetns text)

  Description: This function maps the contents of a cursor to an XML document.

  Return type: XML

  Example:

```
gaussdb=# CURSOR xc WITH HOLD FOR SELECT * FROM testxmlschema.test1 ORDER BY 1, 2;
DECLARE CURSOR
gaussdb=# SELECT cursor_to_xml('xc'::refcursor, 5, false, true, '');
                    cursor_to_xml
-------------------------------------------------------------
 <row xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">+
                                                +
  <a>-1</a>                                       +
 </row>                                          +
                                                +
 <row xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">+
                                                +
  <a>1</a>                                        +
  <b>one</b>                                       +
 </row>                                          +
                                                +
 <row xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">+
                                                +
  <a>2</a>                                        +
  <b>two</b>                                       +
 </row>                                          +
                                                +


(1 row)
```

- cursor_to_xmlschema(cursor refcursor, nulls boolean, tableforest boolean, targetns text)

  Description: This function maps the contents of a cursor to an XML schema document.

  Return type: XML

  Example:

```
gaussdb=# SELECT cursor_to_xmlschema('xc'::refcursor, true, false, '');
                                cursor_to_xmlschema
-----------------------------------------------------------------------------------------------
 <xsd:schema                                                                  +
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">                                    +
                                                                             +
 <xsd:simpleType name="INTEGER">                                                +
  <xsd:restriction base="xsd:int">                                               +
    <xsd:maxInclusive value="2147483647"/>                                        +
    <xsd:minInclusive value="-2147483648"/>                                       +
  </xsd:restriction>                                                           +
 </xsd:simpleType>                                                              +
                                                                             +
 <xsd:simpleType name="UDT.regression.pg_catalog.text">                                 +
  <xsd:restriction base="xsd:string">                                            +
  </xsd:restriction>                                                           +
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
</xsd:simpleType>                                                                +
                                                                               +
<xsd:complexType name="RowType">                                                 +
  <xsd:sequence>                                                               +
    <xsd:element name="a" type="INTEGER" nillable="true"></xsd:element>          +
    <xsd:element name="b" type="UDT.regression.pg_catalog.text" nillable="true"></xsd:element>+
  </xsd:sequence>                                                              +
</xsd:complexType>                                                               +
                                                                               +
<xsd:complexType name="TableType">                                               +
  <xsd:sequence>                                                               +
    <xsd:element name="row" type="RowType" minOccurs="0" maxOccurs="unbounded"/>      +
  </xsd:sequence>                                                              +
</xsd:complexType>                                                               +
                                                                               +
<xsd:element name="table" type="TableType"/>                                     +
                                                                               +
</xsd:schema>
(1 row)
```

- schema_to_xml(schema name, nulls boolean, tableforest boolean, targetns text)

  Description: This function maps the contents of the entire schema to an XML document.

  Return type: XML

  Example:

```
gaussdb=# SELECT schema_to_xml('testxmlschema', false, true, '');
                          schema_to_xml
-----------------------------------------------------------------------
 <testxmlschema xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">+
                                                                     +
 <test1>                                                               +
                                                                     +
   <a>1</a>                                                             +
   <b>one</b>                                                             +
 </test1>                                                              +
                                                                     +
 <test1>                                                               +
                                                                     +
   <a>2</a>                                                             +
   <b>two</b>                                                             +
 </test1>                                                              +
                                                                     +
 <test1>                                                               +
                                                                     +
   <a>-1</a>                                                            +
 </test1>                                                              +
                                                                     +
                                                                     +
 </testxmlschema>                                                          +
(1 row)
```

- schema_to_xmlschema(schema name, nulls boolean, tableforest boolean, targetns text)

  Description: This function maps the contents of the entire schema to an XML schema document.

  Return type: XML

  Example:

```
gaussdb=# SELECT schema_to_xmlschema('testxmlschema', false, true, '');
                                  schema_to_xmlschema
-----------------------------------------------------------------------------------------------------------
 <xsd:schema                                                                                  +
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">                                                  +
                                                                                             +
 <xsd:simpleType name="INTEGER">                                                                 +
```

```
  <xsd:restriction base="xsd:int">                                                +
    <xsd:maxInclusive value="2147483647"/>                                        +
    <xsd:minInclusive value="-2147483648"/>                                       +
  </xsd:restriction>                                                            +
</xsd:simpleType>                                                               +
                                                                                +
<xsd:simpleType name="UDT.t1.pg_catalog.text">                                  +
  <xsd:restriction base="xsd:string">                                           +
  </xsd:restriction>                                                            +
</xsd:simpleType>                                                               +
                                                                                +
<xsd:complexType name="SchemaType.t1.testxmlschema">                            +
  <xsd:sequence>                                                                +
    <xsd:element name="test1" type="RowType.t1.testxmlschema.test1" minOccurs="0"
maxOccurs="unbounded"/>+
  </xsd:sequence>                                                               +
</xsd:complexType>                                                              +
                                                                                +
<xsd:element name="testxmlschema" type="SchemaType.t1.testxmlschema"/>
+
                                                                                +
</xsd:schema>
(1 row)
```

- schema_to_xml_and_xmlschema(schema name, nulls boolean, tableforest boolean, targetns text)

  Description: This function maps the contents of the entire schema to an XML document and an XML schema document, and joins the two documents together.

  Return type: XML

  Example:

```
gaussdb=# SELECT schema_to_xml_and_xmlschema('testxmlschema', true, true, 'foo');
                                    schema_to_xml_and_xmlschema
-----------------------------------------------------------------------------------------------------------------
<testxmlschema xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="foo"
xsi:schemaLocation="foo #">+
                                                                                +
                                                                    +
<xsd:schema                                                                     +
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"                                +
    targetNamespace="foo"                                                       +
    elementFormDefault="qualified">                                             +
                                                                                +
<xsd:simpleType name="INTEGER">                                                 +
  <xsd:restriction base="xsd:int">                                              +
    <xsd:maxInclusive value="2147483647"/>                                      +
    <xsd:minInclusive value="-2147483648"/>                                     +
  </xsd:restriction>                                                            +
</xsd:simpleType>                                                               +
                                                                                +
<xsd:simpleType name="UDT.t1.pg_catalog.text">                                  +
  <xsd:restriction base="xsd:string">                                           +
  </xsd:restriction>                                                            +
</xsd:simpleType>                                                               +
                                                                                +
<xsd:complexType name="SchemaType.t1.testxmlschema">                            +
  <xsd:sequence>                                                                +
    <xsd:element name="test1" type="RowType.t1.testxmlschema.test1" minOccurs="0"
maxOccurs="unbounded"/>   +
  </xsd:sequence>                                                               +
</xsd:complexType>                                                              +
                                                                                +
<xsd:element name="testxmlschema"
type="SchemaType.t1.testxmlschema"/>                            +
                                                                                +
</xsd:schema>                                                                   +
                                                                                +
```

```
<test1>                                                                 +
                                                                       +
  <a>1</a>                                                              +
  <b>one</b>                                                             +
</test1>                                                                +
                                                                       +
<test1>                                                                 +
                                                                       +
  <a>2</a>                                                              +
  <b>two</b>                                                             +
</test1>                                                                +
                                                                       +
<test1>                                                                 +
                                                                       +
  <a>-1</a>                                                             +
  <b xsi:nil="true"/>                                                    +
</test1>                                                                +
                                                                       +
                                                                       +
</testxmlschema>                                                              +

(1 row)
```

- database_to_xml(nulls boolean, tableforest boolean, targetns text)

  Description: This function maps the contents of the entire database to an XML document.

  Return type: XML

  Example:

```
gaussdb=# SELECT database_to_xml(true, true, 'test');
                          database_to_xml
-------------------------------------------------------------------------
 <test xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="test">+
                                                                       +
 <dbe_x005F_xml>                                                        +
                                                                       +
 </dbe_x005F_xml>                                                       +
                                                                       +
 <dbe_x005F_xmldom>                                                      +
                                                                       +
 </dbe_x005F_xmldom>                                                     +
                                                                       +
 <dbe_x005F_xmlparser>                                                   +
                                                                       +
 </dbe_x005F_xmlparser>                                                  +
                                                                       +
 <public>                                                                +
                                                                       +
 </public>                                                               +
                                                                       +
 </test>                                                                 +

(1 row)
```

- database_to_xmlschema(nulls boolean, tableforest boolean, targetns text)

  Description: This function maps the contents of the entire database to an XML schema document.

  Return type: XML

  Example:

```
gaussdb=# SELECT database_to_xmlschema(true, true, 'test');
                             database_to_xmlschema
-----------------------------------------------------------------------------------------
 <xsd:schema                                                                    +
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"                                 +
    targetNamespace="test"                                                      +
    elementFormDefault="qualified">                                              +
                                                                               +
```

```
<xsd:complexType name="CatalogType.test">                                    +
  <xsd:all>                                                                  +
    <xsd:element name="dbe_x005F_xml" type="SchemaType.test.dbe_x005F_xml"/>  +
    <xsd:element name="dbe_x005F_xmldom" type="SchemaType.test.dbe_x005F_xmldom"/>  +
    <xsd:element name="dbe_x005F_xmlparser" type="SchemaType.test.dbe_x005F_xmlparser"/>+
    <xsd:element name="public" type="SchemaType.test.public"/>               +
  </xsd:all>                                                                 +
</xsd:complexType>                                                           +
                                                                           +
<xsd:element name="test" type="CatalogType.test"/>                           +
                                                                           +
</xsd:schema>
(1 row)
```

- database_to_xml_and_xmlschema(nulls boolean, tableforest boolean, targetns text)

  Description: This function maps the contents of the entire database to an XML document and an XML schema document, and joins the two documents together.

  Return type: XML

  Example:

```
gaussdb=# SELECT database_to_xml_and_xmlschema(true, true, 'test');
                                database_to_xml_and_xmlschema
-------------------------------------------------------------------------------------------------------
 <test xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="test"
xsi:schemaLocation="test #">+
                                                                           +
<xsd:schema                                                                  +
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"                               +
    targetNamespace="test"                                                   +
    elementFormDefault="qualified">                                            +
                                                                           +
<xsd:complexType name="CatalogType.test">                                      +
  <xsd:all>                                                                +
    <xsd:element name="dbe_x005F_xml" type="SchemaType.test.dbe_x005F_xml"/>                    +
    <xsd:element name="dbe_x005F_xmldom"
type="SchemaType.test.dbe_x005F_xmldom"/>                +
    <xsd:element name="dbe_x005F_xmlparser"
type="SchemaType.test.dbe_x005F_xmlparser"/>           +
    <xsd:element name="public" type="SchemaType.test.public"/>                               +
  </xsd:all>                                                               +
</xsd:complexType>                                                           +
                                                                           +
<xsd:element name="test" type="CatalogType.test"/>                             +
                                                                           +
</xsd:schema>                                                                +
                                                                           +
<dbe_x005F_xml>                                                              +
                                                                           +
</dbe_x005F_xml>                                                             +
                                                                           +
<dbe_x005F_xmldom>                                                             +
                                                                           +
</dbe_x005F_xmldom>                                                            +
                                                                           +
<dbe_x005F_xmlparser>                                                          +
                                                                           +
</dbe_x005F_xmlparser>                                                         +
                                                                           +
<public>                                                                     +
                                                                           +
</public>                                                                    +
                                                                           +
</test>                                                                      +

(1 row)
```

- table_to_xml(tbl regclass, nulls boolean, tableforest boolean, targetns text)

  Description: This function maps the contents of a relational table to an XML document.

  Return type: XML

  Example:

```
gaussdb=# SELECT table_to_xml('testxmlschema.test1', false, false, '');
                      table_to_xml
-------------------------------------------------------------
 <test1 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">+
                                                            +
 <row>                                                      +
   <a>1</a>                                                  +
   <b>one</b>                                                 +
 </row>                                                     +
                                                            +
 <row>                                                      +
   <a>2</a>                                                  +
   <b>two</b>                                                 +
 </row>                                                     +
                                                            +
 <row>                                                      +
   <a>-1</a>                                                 +
 </row>                                                     +
                                                            +
 </test1>                                                   +

(1 row)
```

- table_to_xmlschema(tbl regclass, nulls boolean, tableforest boolean, targetns text)

  Description: This function maps the contents of a relational table to an XML schema document.

  Return type: XML

  Example:

```
gaussdb=# SELECT table_to_xmlschema('testxmlschema.test1', false, false, '');
                                        table_to_xmlschema
-------------------------------------------------------------------------------------------------------------------
 <xsd:schema                                                                                                  +
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">                                                                +
                                                                                                               +
 <xsd:simpleType name="INTEGER">                                                                              +
   <xsd:restriction base="xsd:int">                                                                            +
     <xsd:maxInclusive value="2147483647"/>                                                                    +
     <xsd:minInclusive value="-2147483648"/>                                                                   +
   </xsd:restriction>                                                                                        +
 </xsd:simpleType>                                                                                           +
                                                                                                               +
 <xsd:simpleType name="UDT.regression.pg_catalog.text">                                                                +
   <xsd:restriction base="xsd:string">                                                                         +
   </xsd:restriction>                                                                                        +
 </xsd:simpleType>                                                                                           +
                                                                                                               +
 <xsd:complexType name="RowType.regression.testxmlschema.test1">
+
   <xsd:sequence>                                                                                            +
     <xsd:element name="a" type="INTEGER" minOccurs="0"></
xsd:element>                                        +
     <xsd:element name="b" type="UDT.regression.pg_catalog.text" minOccurs="0"></
xsd:element>                    +
   </xsd:sequence>                                                                                           +
 </xsd:complexType>                                                                                          +
                                                                                                               +
 <xsd:complexType name="TableType.regression.testxmlschema.test1">
+
   <xsd:sequence>                                                                                            +
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    <xsd:element name="row" type="RowType.regression.testxmlschema.test1" minOccurs="0"
maxOccurs="unbounded"/>+
  </xsd:sequence>                                                                 +
 </xsd:complexType>                                                                 +
                                                                                  +
 <xsd:element name="test1"
type="TableType.regression.testxmlschema.test1"/>                                +
                                                                                   +
 </xsd:schema>
(1 row)
```

- table_to_xml_and_xmlschema(tbl regclass, nulls boolean, tableforest boolean, targetns text)

  Description: This function maps the contents of a relational table to an XML document and an XML schema document, and joins the two documents together.

  Return type: XML

  Example:

```
gaussdb=# SELECT table_to_xml_and_xmlschema('testxmlschema.test1', false, false, '');
                                    table_to_xml_and_xmlschema
-----------------------------------------------------------------------------------------------------------------
 <test1 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="#">                +
                                                                                  +
 <xsd:schema                                                                        +
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">                                     +
                                                                                  +
 <xsd:simpleType name="INTEGER">                                                     +
  <xsd:restriction base="xsd:int">                                                    +
   <xsd:maxInclusive value="2147483647"/>                                              +
   <xsd:minInclusive value="-2147483648"/>                                             +
  </xsd:restriction>                                                                +
 </xsd:simpleType>                                                                   +
                                                                                  +
 <xsd:simpleType name="UDT.regression.pg_catalog.text">                                 +
  <xsd:restriction base="xsd:string">                                                 +
  </xsd:restriction>                                                                +
 </xsd:simpleType>                                                                   +
                                                                                  +
 <xsd:complexType name="RowType.regression.testxmlschema.test1">
+
  <xsd:sequence>                                                                   +
   <xsd:element name="a" type="INTEGER" minOccurs="0"></
xsd:element>                                    +
   <xsd:element name="b" type="UDT.regression.pg_catalog.text" minOccurs="0"></
xsd:element>                +
  </xsd:sequence>                                                                  +
 </xsd:complexType>                                                                 +
                                                                                  +
 <xsd:complexType name="TableType.regression.testxmlschema.test1">
+
  <xsd:sequence>                                                                   +
   <xsd:element name="row" type="RowType.regression.testxmlschema.test1" minOccurs="0"
maxOccurs="unbounded"/>+
  </xsd:sequence>                                                                  +
 </xsd:complexType>                                                                 +
                                                                                  +
 <xsd:element name="test1"
type="TableType.regression.testxmlschema.test1"/>                                +
                                                                                   +
 </xsd:schema>                                                                       +
                                                                                  +
 <row>                                                                              +
  <a>1</a>                                                                           +
  <b>one</b>                                                                         +
 </row>                                                                             +
                                                                                  +
```

```
<row>                                                                +
  <a>2</a>                                                            +
  <b>two</b>                                                            +
</row>                                                               +
                                                                   +
<row>                                                                +
  <a>-1</a>                                                           +
</row>                                                               +
                                                                   +
</test1>                                                             +

(1 row)
```

NOTE

- For XPath-related functions, only XPath() and XPath_exists() are supported. These functions use the XPath language to query XML files and depend on the Libxml2 library, which is provided only in XPath1.0. Therefore, only XPath 1.0 is supported.
- The XQuery, XML extension, and XSLT functions are not supported.

- getclobval(xml)

  Description: Converts an XML type to a CLOB type. This function is valid only when parameters **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s4**.

  Parameter: The input parameter is of the XML type.

  Return type: CLOB

  Example:

```
gaussdb=# CREATE DATABASE gaussdb_ora DBCOMPATIBILITY='ORA';
CREATE DATABASE
gaussdb=# \c gaussdb_ora
gaussdb_ora=# SET a_format_version='10c';
SET
gaussdb_ora=# SET a_format_dev_version='s4';
SET
gaussdb_ora=# declare
 xmldata xml;
 result clob;
begin
 xmldata := '<a>123</a>';
 result := getclobval(xmldata);
 RAISE NOTICE 'result is : %',result;
END;
/
NOTICE:  result is : <a>123</a>
gaussdb_ora=# SELECT getclobval(xmlparse(document '<a>123</a>'));
 getclobval
------------
 <a>123</a>
(1 row)

gaussdb_ora=# \c postgres
gaussdb=# DROP DATABASE gaussdb_ora;
DROP DATABASE
```

- getstringval(xml)

  Description: Converts an XML type to a string. This function is valid only when **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s4**.

  Parameter: The input parameter is of the XML type.

  Return type: VARCHAR2

  Example:

```
gaussdb=# CREATE DATABASE gaussdb_ora DBCOMPATIBILITY='ORA';
CREATE DATABASE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# \c gaussdb_ora
gaussdb_ora=# SET a_format_version='10c';
SET
gaussdb_ora=# SET a_format_dev_version='s4';
SET
gaussdb_ora=# declare
 xmldata xml;
 result varchar2;
begin
 xmldata := '<a>123<b>456</b></a>';
 result := getstringval(xmldata);
 RAISE NOTICE 'result is : %',result;
END;
/
NOTICE:  result is : <a>123<b>456</b></a>
gaussdb_ora=# select getstringval(xmlparse(document '<a>123<b>456</b></a>'));
    getstringval
----------------------
 <a>123<b>456</b></a>
(1 row)

gaussdb_ora=# \c postgres
gaussdb=# DROP DATABASE gaussdb_ora;
DROP DATABASE
```

- xmlsequence(xml)

  Description: Converts an XML parameter into an array of the XMLTYPE type. Each array element is an XMLTYPE object. The input parameter of this function cannot be null and must be a valid XML document. If the input parameter does not meet the requirements, the function returns a null value or throws an exception. This function can be used to process multiple child nodes in an XML document or split an XML document into multiple fragments.

  Parameter: XML

  Return type: array of the XMLType type.

  Example: If you want to convert this document into an array containing three elements, each of which is a book node, use the following statement:

```
gaussdb=# SELECT xmlsequence(xml('<books><book><title>The Catcher in the Rye</
title><author>J.D. Salinger</author><year>1951</year></book><book><title>1984</
title><author>George Orwell</author><year>1949</year></book><book><title>The Hitchhiker''s
Guide to the Galaxy</title><author>Douglas Adams</author><year>1979</year></book></books>'));
                      xmlsequence
------------------------------------------------------------
 {"<books>                                          +
    <book>                                         +
      <title>The Catcher in the Rye</title>           +
      <author>J.D. Salinger</author>                  +
      <year>1951</year>                               +
    </book>                                        +
    <book>                                         +
      <title>1984</title>                             +
      <author>George Orwell</author>                    +
      <year>1949</year>                               +
    </book>                                        +
    <book>                                         +
      <title>The Hitchhiker's Guide to the Galaxy</title>+
      <author>Douglas Adams</author>                    +
      <year>1979</year>                               +
    </book>                                        +
  </books>"}
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

If an XML file contains double quotation marks, the result of the xmlsequence function contains escape characters of double quotation marks when you query the result of the xmlsequence function independently. The use of the xmlsequence function is not affected.

- xmlcast(value_expression as datatype)

  Description: Converts the input data of the XML type to the specified target data type after extracting the text content from the document.

  Parameter: **value_expression** indicates the document content of the XML type.

  Return type: specified target data type.

  Example:

```
-- Convert data of the XML type to data of the NUMBER type.
gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA';
CREATE DATABASE
gaussdb=# \c ora_db
ora_db=# SELECT XMLCAST(XML('<number_value>123.45</number_value>') AS NUMBER) AS
number_value FROM dual;
 number_value
--------------
       123.45
(1 row)
ora_db=# \c postgres
gaussdb=# DROP DATABASE ora_db;
DROP DATABASE
```

NOTE

- This function can be used only in Oracle-compatible mode.
- The malformed XML files support only the XML files with multiple root nodes returned by the value_expression expression.
- The following target data types cannot be converted: BYTEAWITHOUTORDERWITHEQUALCOL, BYTEAWITHOUTORDERCOL, _BYTEAWITHOUTORDERWITHEQUALCOL, _BYTEAWITHOUTORDERCOL, and some pseudo types.
- This function cannot be used during the upgrade observation period.

1.6.45 Functions of the XMLType Type

NOTE

XMLType functions support a maximum of 256 nested layers in XML files.

- createxml(varchar2[,varchar2,numeric,numeric])

  Description: Statically creates the XMLType type. The input parameters are of the varchar2 type.

  Parameters: The first parameter is the character string to be converted to XMLType (required column). The second parameter is the optional schema URL used to make the input comply with the specified schema (optional column, which is empty by default and does not take effect currently). The third parameter is the flag indicating whether the instance is valid according to the given XML schema (optional column, which is **0** by default and does not take effect currently). The fourth parameter is the flag indicating whether the instance is well-formed (optional column, which is **0** by default and does not take effect currently).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: XMLType

Example:

```
gaussdb=# SELECT createxml('<a>123</a>');
 createxml
--------------
 <a>123</a>
(1 row)
```

NOTE

- Different from that in the Oracle Database, in PL/SQL, createxml allows input parameters to be empty strings and returns **NULL**.
- For character encoding, only UTF-8, GBK, ZHS16GBK, and LATIN1 to LATIN10 are supported, and the **version** column can only be set to **1.x**.
- The createxml function can be called using the xmltype.createxml() syntax.

  Example:

  ```
  gaussdb=# SELECT xmltype.createxml('<a>123</a>');
   createxml
  ------------
   <a>123</a>
  (1 row)
  ```

- In this chapter, the function whose input parameter is xmltype() can be called in xmltype().func() mode. The XMLType type returned by a function is passed to the next function as the input parameter. This syntax supports multi-layer nesting. (If the input parameter is defined as XMLType by a user, this syntax is not supported.)

  Example:

  ```
  gaussdb=# SELECT xmltype('<a>123<b>456</b></a>').extract('/a/b').getstringval();
   xmltypefunc
  -------------
   <b>456</b>
  (1 row)
  ```

  The actual effect of the preceding example is the same as that of the following function nesting:

  ```
  gaussdb=# SELECT getstringval(extractxml(xmltype('<a>123<b>456</b></a>'),'/a/b'));
   getstringval
  --------------
   <b>456</b>
  (1 row)
  ```

- In a stored procedure, variables of the XMLType type can call functions in a.func() mode. This syntax supports one-layer nesting.

  Example:

  ```
  gaussdb=# declare
          a xmltype;
          b varchar2;
  begin
          a:=xmltype('<a>123<b>456</b></a>');
          b:=a.getstringval();
          RAISE NOTICE 'xmltype_str is : %',b;
  end;
  /
  NOTICE:  xmltype_str is : <a>123<b>456</b></a>
  ```

- createxml(clob [,varchar2,numeric ,numeric])

  Description: Statically creates the XMLType type. The input parameters are of the CLOB type.

  Parameters: The first parameter is the CLOB to be converted to XMLType (required column). The second parameter is the optional schema URL used to make the input comply with the specified schema (optional column, which is empty by default and does not take effect currently). The third parameter is the flag indicating whether the instance is valid according to the given XML

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

schema (optional column, which is **0** by default and does not take effect currently). The fourth parameter is the flag indicating whether the instance is well-formed (optional column, which is **0** by default and does not take effect currently).

Return type: XMLType

Example:

```
gaussdb=# declare
 xmltype_clob clob;
 xmltype_obj xmltype;
 xmltype_str varchar2(1000);
begin
 xmltype_clob := '<a>123</a>';
 xmltype_obj := createxml(xmltype_clob);
 xmltype_str := xmltype_obj.getstringval();
 RAISE NOTICE 'xmltype_str is : %',xmltype_str;
end;
/
NOTICE:  xmltype_str is : <a>123</a>
```

NOTE

The maximum size of the input parameter of the CLOB type is 1 GB minus 1 byte.

- createxml(blob,numeric[,varchar2,numeric,numeric])

  Description: Statically creates the XMLType type. The input parameters are of the BLOB type.

  Parameters: The first parameter is the BLOB to be converted to XMLType (required column). The second parameter is the character set ID of the input XML data (required column). The third parameter is the optional schema URL used to make the input comply with the specified schema (optional column, which is empty by default and does not take effect currently). The fourth parameter is the flag indicating whether the instance is valid according to the given XML schema (optional column, which is **0** by default and does not take effect currently). The fifth parameter is the flag indicating whether the instance is well-formed (optional column, which is **0** by default and does not take effect currently).

  Return type: XMLType

  Example:

```
gaussdb=# declare
 xmltype_blob blob;
 xmltype_obj xmltype;
 xmltype_str varchar2(1000);
begin
 xmltype_blob := xmltype('<a>123</a>').getblobval(7);
 xmltype_obj := createxml(xmltype_blob,7);
 xmltype_str := xmltype_obj.getstringval();
 RAISE NOTICE 'xmltype_str is : %',xmltype_str;
end;
/
NOTICE:  xmltype_str is : <?xml version="1.0" encoding="UTF8"?>
<a>123</a>
```

NOTE

  - The maximum size of input parameters of the BLOB type is 256 MB minus 1 byte.
  - The character set ID ranges from 1 to 43.

- getblobval(xmltype,numeric)

  Description: Converts the XMLType type to the BLOB type. The xmltype().func() method can be called.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: The first parameter is of the XMLType type, and the second parameter is the ID of the target character set to be converted.

Return type: BLOB

Example:

```
gaussdb=# SELECT getblobval(xmltype('<asd/>'),7);
                                 getblobval
----------------------------------------------------------------------------------------------------

3C3F786D6C2076657273696F6E3D22312E302220656E636F64696E673D2255544638223F3E0A3C6173
642F3E
(1 row)
```

xmltype ().func ():

```
gaussdb=# SELECT xmltype('<asd/>').getblobVal(7);
                                 xmltypefunc
----------------------------------------------------------------------------------------------------

3C3F786D6C2076657273696F6E3D22312E302220656E636F64696E673D2255544638223F3E0A3C6173
642F3E
(1 row)
```

NOTE

The maximum length of the input parameter of the XMLType type is 256 MB minus 1 byte.

- getclobval(xmltype)

  Description: Converts the XMLType type to the CLOB type. The xmltype().func() method can be called.

  Parameter: The input parameter is of the XMLType type.

  Return type: CLOB

  Example:

  ```
  gaussdb=# SELECT getclobval(xmltype('<a>123</a>'));
   getclobval
  --------------
   <a>123</a>
  (1 row)
  ```

  xmltype ().func ():

  ```
  gaussdb=# SELECT xmltype('<a>123</a>').getclobval();
   xmltypefunc
  --------------
   <a>123</a>
  (1 row)
  ```

- getnumberval(xmltype)

  Description: Converts the XMLType type to the numeric type. The xmltype().func() method can be called.

  Parameter: The input parameter is of the XMLType type.

  Return type: numeric

  Example:

  ```
  gaussdb=# SELECT getnumberval(xmltype('<a>123</a>').extract('/a/text()'));
   getnumberval
  --------------
            123
  (1 row)
  ```

  xmltype ().func ():

  ```
  gaussdb=# SELECT xmltype('<a>123</a>').extract('/a/text()').getnumberval();
   xmltypefunc
  --------------
            123
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- isfragment(xmltype)

  Description: Returns a result indicating whether the XMLType type is fragment (1) or document (0). The xmltype().func() method can be called.

  Parameter: The input parameter is of the XMLType type.

  Return type: numeric

  Example:

```
gaussdb=# SELECT isfragment(xmltype('<a>123</a>'));
 isfragment
--------------
            0
(1 row)
```

  xmltype ().func ():

```
gaussdb=# SELECT xmltype('<a>123</a>').isfragment();
 xmltypefunc
--------------
            0
(1 row)
```

- xmltype(varchar2[,varchar2,numeric,numeric])

  Description: Creates the XMLType type from the varchar2 type.

  Parameters: The first parameter is the character string to be converted to XMLType (required column). The second parameter is the optional schema URL used to make the input comply with the specified schema (optional column, which is empty by default and does not take effect currently). The third parameter is the flag indicating whether the instance is valid according to the given XML schema (optional column, which is **0** by default and does not take effect currently). The fourth parameter is the flag indicating whether the instance is well-formed (optional column, which is **0** by default and does not take effect currently).

  Return type: XMLType

  Example:

```
gaussdb=# SELECT xmltype('<a>123</a>');
 xmltype
--------------
 <a>123</a>
(1 row)
```

  NOTE

  - Different from that in the Oracle Database, in PL/SQL, XMLType allows input parameters to be empty strings and returns **NULL**.
  - For character encoding, only UTF-8, GBK, ZHS16GBK, and LATIN1 to LATIN10 are supported, and the **version** column can only be set to **1.x**.

- xmltype(clob[,varchar2,numeric,numeric])

  Description: Creates the XMLType type from the CLOB type.

  Parameters: The first parameter is the CLOB to be converted to XMLType (required column). The second parameter is the optional schema URL used to make the input comply with the specified schema (optional column, which is empty by default and does not take effect currently). The third parameter is the flag indicating whether the instance is valid according to the given XML schema (optional column, which is **0** by default and does not take effect currently). The fourth parameter is the flag indicating whether the instance is well-formed (optional column, which is **0** by default and does not take effect currently).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: XMLType

Example:

```
gaussdb=# declare
 xmltype_clob clob;
 xmltype_obj xmltype;
 xmltype_str varchar2(1000);
begin
 xmltype_clob := '<a>123</a>';
 xmltype_obj := xmltype(xmltype_clob);
 xmltype_str := xmltype_obj.getstringval();
 RAISE NOTICE 'xmltype_str is : %',xmltype_str;
end;
/
NOTICE:  xmltype_str is : <a>123</a>
```

NOTE

The maximum size of the input parameter of the CLOB type is 1 GB minus 1 byte.

- xmltype(blob, numeric [,varchar2,numeric ,numeric])

  Description: Creates the XMLType type from the BLOB type.

  Parameters: The first parameter is the BLOB to be converted to XMLType (required column). The second parameter is the character set ID of the input XML data. The third parameter is the optional schema URL used to make the input comply with the specified schema (optional column, which is empty by default and does not take effect currently). The fourth parameter is the flag indicating whether the instance is valid according to the given XML schema (optional column, which is **0** by default and does not take effect currently). The fifth parameter is the flag indicating whether the instance is well-formed (optional column, which is **0** by default and does not take effect currently).

  Return type: XMLType

  Example:

```
gaussdb=# declare
 xmltype_blob blob;
 xmltype_obj xmltype;
 xmltype_str varchar2(1000);
begin
 xmltype_blob := getblobval(createxml('<a>123</a>'),7);
 xmltype_obj := xmltype(xmltype_blob,7);
 xmltype_str := xmltype_obj.getstringval();
 RAISE NOTICE 'xmltype_str is : %',xmltype_str;
end;
/
NOTICE:  xmltype_str is : <?xml version="1.0" encoding="UTF8"?>
<a>123</a>
```

NOTE

- The maximum size of input parameters of the BLOB type is 256 MB minus 1 byte.
- The character set ID ranges from 1 to 42.

- getstringval(xmltype)

  Description: Converts the XMLType to a string.

  Parameter: XMLType to be converted.

  Return type: varchar2

  The getstringval function can be called in either of the following ways:

  Example 1:

```
gaussdb=# SELECT getstringval('<a>123<b>456</b></a>');
    getstringval
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
----------------------
 <a>123<b>456</b></a>
(1 row)
```

Example 2: The calling mode is compatible with the ORA syntax.

```
gaussdb=# SELECT xmltype('<a>123<b>456</b></a>').getstringval();
    xmltypefunc
----------------------
 <a>123<b>456</b></a>
(1 row)
```

- getrootelement(xmltype)

  Description: Gets the root element of the XMLType type.

  Parameter: XMLType whose root element is to be obtained.

  Return type: varchar2

  The getrootelement function can be called in either of the following ways:

  Example 1:

```
gaussdb=# SELECT getrootelement('<a>123<b>456</b></a>');
 getrootelement
----------------
 a
(1 row)
```

  Example 2: The calling mode is compatible with the ORA syntax.

```
gaussdb=# SELECT xmltype('<a>123<b>456</b></a>').getrootelement();
 xmltypefunc
-------------
 a
(1 row)
```

- getnamespace(xmltype)

  Description: Gets the namespace of the XMLType top-level element.

  Parameter: XMLType whose namespace is to be obtained.

  Return type: varchar2

  The getnamespace function can be called in either of the following ways:

  Example 1:

```
gaussdb=# SELECT getnamespace('<c:a xmlns:c="asd">123<d:b xmlns:d="qwe">456</d:b></c:a>');
 getnamespace
--------------
 asd
(1 row)
```

  Example 2: The calling mode is compatible with the ORA syntax.

```
gaussdb=# SELECT xmltype('<c:a xmlns:c="asd">123<d:b xmlns:d="qwe">456</d:b></
c:a>').getnamespace();
 xmltypefunc
-------------
 asd
(1 row)
```

- existsnode(xmltype, varchar2[, varchar2])

  Description: Determines whether the XML node exists in XMLType based on the XPath expression. If the XML node exists, **1** is returned. Otherwise, **0** is returned.

  Parameters: XMLType to be queried, path of the XPath node to be queried, and namespace of the XPath path (If the input parameter has a namespace, aliases must be defined for both the XPath and namespace, as shown in example 3.)

  Return type: numeric

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The existsnode function can be called in either of the following ways:

Example 1:

```
gaussdb=# SELECT existsnode('<a>123<b>456</b></a>','/a/b');
 existsnode
------------
          1
(1 row)
```

Example 2: The calling mode is compatible with the ORA syntax.

```
gaussdb=# SELECT xmltype('<a>123<b>456</b></a>').existsnode('/a/b');
 xmltypefunc
-------------
           1
(1 row)
```

Example 3:

```
gaussdb=# SELECT existsnode('<a:b xmlns:a="asd">123<c>456</c></a:b>','/a:b/c','xmlns:a="asd"');
 existsnode
------------
          1
(1 row)
```

Example 4: The calling mode is compatible with the ORA syntax.

```
gaussdb=# SELECT xmltype('<a:b xmlns:a="asd">123<c>456</c></a:b>').existsnode('/a:b/
c','xmlns:a="asd"');
 xmltypefunc
-------------
           1
(1 row)
```

- extractxml(xmltype, varchar2[, varchar2])

  Description: Checks whether an XML node exists in the given XMLType based on the XPath expression. If so, the XMLType containing the node is returned. If not, **NULL** is returned. The return value can be inserted into a table of the XMLType type.

  Parameters: XMLType to be queried, path of the XPath node to be queried, and namespace of the XPath path (If the input parameter has a namespace, aliases must be defined for both the XPath and namespace, as shown in example 3.)

  Return type: XMLType

  The extractxml function can be called in either of the following ways:

  Example 1:

  ```
  gaussdb=# SELECT extractxml('<a>123<b>456</b></a>','/a/b');
   extractxml
  ------------
   <b>456</b>
  (1 row)
  ```

  Example 2: The calling mode is compatible with the ORA syntax.

  ```
  gaussdb=# SELECT xmltype('<a>123<b>456</b></a>').extract('/a/b');
   xmltypefunc
  -------------
   <b>456</b>
  (1 row)

  gaussdb=# SELECT xmltype('<a>123<b>456</b></a>').extractxml('/a/b');
   xmltypefunc
  -------------
   <b>456</b>
  (1 row)
  ```

  Example 3:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT extractxml('<a:b xmlns:a="asd">123<c>456</c></a:b>','/a:b','xmlns:a="asd"');
              extractxml
----------------------------------------
 <a:b xmlns:a="asd">123<c>456</c></a:b>
(1 row)
```

Example 4: The calling mode is compatible with the ORA syntax.

```
gaussdb=# SELECT xmltype('<a:b xmlns:a="asd">123<c>456</c></a:b>').extract('/a:b','xmlns:a="asd"');
             xmltypefunc
----------------------------------------
 <a:b xmlns:a="asd">123<c>456</c></a:b>
(1 row)

gaussdb=# SELECT xmltype('<a:b xmlns:a="asd">123<c>456</c></a:b>').extractxml('/
a:b','xmlns:a="asd"');
             xmltypefunc
----------------------------------------
 <a:b xmlns:a="asd">123<c>456</c></a:b>
(1 row)
```

- extractvalue(xmltype | xml, varchar2[, varchar2])

  Description: Extracts the value of an XPath expression from an XML file based on the XPath expression (only XPath 1.0 is supported). The result of the XPath expression must be a single node and must be a text node, attribute, or element. If the XPath expression contains an element expression, the element must have a text node as a child node. The function returns the text. If the result is an attribute, the function returns the value of the attribute.

  Parameters: **xmltype | xml**: XML text to be queried; **varchar2**: XPath expression to be queried (XPath node path); **[, varchar2]** (optional): Namespace of the XPath node path. (If the input parameter contains a namespace with an alias, you need to define an alias for both the XPath expression and namespace. If the default namespace is used, you do not need to define an alias, as shown in example 3.)

  Return type: varchar2

  The extractvalue function can be called in two ways: input without namespace and input with namespace.

  Example 1: The input does not contain a namespace.

```
gaussdb=# set a_format_version='10c';
SET
gaussdb=# set a_format_dev_version='s5';
SET
gaussdb=# SELECT EXTRACTVALUE(xmltype('<book><title>Harry Potter</title><author>J.K. Rowling</
author></book>'), '/book/title') AS book_title;
  book_title
--------------
 Harry Potter
(1 row)
```

  Example 2: The input contains a namespace.

```
gaussdb=# SELECT EXTRACTVALUE(xmltype('<ns:book xmlns:ns="http://
www.example.com"><ns:title>Harry Potter</ns:title><ns:author>J.K. Rowling</ns:author></ns:book>'),
'/ns:book/ns:title', 'xmlns:ns="http://www.example.com"') AS book_title;
  book_title
--------------
 Harry Potter
(1 row)
```

  Example 3: The input contains multiple namespaces.

```
gaussdb=# SELECT EXTRACTVALUE(xmltype('<ns:book xmlns:ns="http://www.example.com"
xmlns:ff="http://www.ff.com"><ff:title>Harry Potter</ff:title><ns:author>J.K. Rowling</ns:author></
ns:book>'), '/ns:book/ff:title', 'xmlns:ns="http://www.example.com" xmlns:ff="http://www.ff.com"') AS
book_title;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 book_title
--------------
 Harry Potter
(1 row)
gaussdb=# SELECT EXTRACTVALUE(xmltype('<store><book xmlns="abc"><root xmlns="abcd">mike</
root><root>mikeab</root></book><root xmlns="abcd">mikedwsa</root></store>'),
'//root', 'xmlns="abc" xmlns:ns2="abc1" xmlns="abcd"') FROM dual;
 extractvalue
--------------
 mikeab
(1 row)
```

NOTE

- If the input contains multiple namespaces, they can be separated by one or more spaces (or newline characters). However, the namespace expression is in the **xmlns:***Name*="*Namespace*" format and the default namespace rule is in the **xmlns='***URL*' format.
- This function is compatible with the xmltype expression function, but the node value returned by the xmltype text must be unique.
- The XPath expression supports only XPath 1.0.
- Currently, the namespace URL in the XML text cannot be a space, and the namespace URL in the namespace expression cannot be a space.
- In the default namespace scenario, the default namespace declared first in the namespace expression is the default namespace of the current XML text.
- This function can be used only in Oracle-compatible mode after setting **a_format_version** to **'10c'** and **a_format_dev_version** to **'s5'**.

- xmlsequence(xmltype)

  Description: Converts an XMLTYPE parameter into an array of the XMLTYPE type. Each array element is an XMLTYPE object. The input parameter of this function cannot be null and must be a valid XML document. If the input parameter does not meet the requirements, the function returns a null value or throws an exception. This function can be used to process multiple child nodes in an XML document or split an XML document into multiple fragments.

  Parameter: The input parameter is of the XMLType type.

  Return type: array of the XMLType type.

  Example 1: If you want to convert this document into an array containing three elements, each of which is a book node, use the following statement:

```
gaussdb=# SELECT xmlsequence(xmltype('<books><book><title>The Catcher in the Rye</
title><author>J.D. Salinger</author><year>1951</year></book><book><title>1984</
title><author>George Orwell</author><year>1949</year></book><book><title>The Hitchhiker''s
Guide to the Galaxy</title><author>Douglas Adams</author><year>1979</year></book></books>'));
                    xmlsequence
------------------------------------------------------------
{"<books>                                    +
   <book>                                   +
     <title>The Catcher in the Rye</title>          +
     <author>J.D. Salinger</author>                 +
     <year>1951</year>                          +
   </book>                                  +
   <book>                                   +
     <title>1984</title>                        +
     <author>George Orwell</author>                 +
     <year>1949</year>                          +
   </book>                                  +
   <book>                                   +
     <title>The Hitchhiker's Guide to the Galaxy</title>+
     <author>Douglas Adams</author>                 +
     <year>1979</year>                          +
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    </book>                                    +
  </books>"}
(1 row)
```

Example 2: If you want to extract the title and author of each book from this array, use the following statement:

```
gaussdb=# SELECT unnest(xmlsequence(xmltype('<books><book><title>The Catcher in the Rye</
title><author>J.D. Salinger</author><year>1951</year></book><book><title>1984</
title><author>George Orwell</author><year>1949</year></book><book><title>The Hitchhiker''s
Guide to the Galaxy</title><author>Douglas Adams</author><year>1979</year></book></
books>').extract('//title/text()'))) AS title
 , unnest(xmlsequence(xmltype('<books><book><title>The Catcher in the Rye</title><author>J.D.
Salinger</author><year>1951</year></book><book><title>1984</title><author>George Orwell</
author><year>1949</year></book><book><title>The Hitchhiker''s Guide to the Galaxy</
title><author>Douglas Adams</author><year>1979</year></book></books>').extract('//author/
text()'))) AS author;
                          title                           |               author
----------------------------------------------------------+----------------------------------------
 The Catcher in the Rye1984The Hitchhiker's Guide to the Galaxy | J.D. SalingerGeorge OrwellDouglas
Adams
(1 row)
```

Example 3: If you want to convert the array into a JSON string, use the following statement:

```
gaussdb=# SELECT array_to_json(array_agg(row_to_json(t)))
FROM (
 SELECT unnest(xmlsequence(xmltype('<books><book><title>The Catcher in the Rye</
title><author>J.D. Salinger</author><year>1951</year></book><book><title>1984</
title><author>George Orwell</author><year>1949</year></book><book><title>The Hitchhiker''s
Guide to the Galaxy</title><author>Douglas Adams</author><year>1979</year></book></
books>').extract('//title/text()'))) AS title
  , unnest(xmlsequence(xmltype('<books><book><title>The Catcher in the Rye</title><author>J.D.
Salinger</author><year>1951</year></book><book><title>1984</title><author>George Orwell</
author><year>1949</year></book><book><title>The Hitchhiker''s Guide to the Galaxy</
title><author>Douglas Adams</author><year>1979</year></book></books>').extract('//author/
text()'))) AS author
) t;
                                array_to_json
-------------------------------------------------------------------------------------------------------------
 [{"title":"The Catcher in the Rye1984The Hitchhiker's Guide to the Galaxy","author":"J.D.
SalingerGeorge OrwellDouglas Adams"}]
(1 row)
```

NOTE

If an XML file contains double quotation marks, the result of the xmlsequence function contains escape characters of double quotation marks when you query the xmlsequence function independently. The use of the xmlsequence function is not affected.

- xmlcast(value_expression as datatype)

  Description: Converts the input data of XMLType to the specified target data type after extracting the text content from the document.

  Parameter: **value_expression** indicates the document content of XMLType.

  Return type: specified target data type.

  Example:

```
-- Convert the data of XMLType to the data of the VARCHAR2 type.
gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA';
CREATE DATABASE
gaussdb=# \c ora_db
ora_db=# SELECT XMLCAST(XMLTYPE('<varchar2_value>Hello, GaussDB!</varchar2_value>') AS
VARCHAR2(50)) AS varchar2_value FROM dual;
 varchar2_value
-----------------
 Hello, GaussDB!
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ora_db=# \c postgres
gaussdb=# DROP DATABASE ora_db;
DROP DATABASE
```

NOTE

- This function can be used only in Oracle-compatible mode.
- The malformed XML files support only the XML files with multiple root nodes returned by the value_expression expression.
- The following target data types cannot be converted: BYTEAWITHOUTORDERWITHEQUALCOL, BYTEAWITHOUTORDERCOL, _BYTEAWITHOUTORDERWITHEQUALCOL, _BYTEAWITHOUTORDERCOL, and some pseudo types.
- This function cannot be used during the upgrade observation period.

- xmlquery(sequence_string RETURNING SEQUENCE)

  Description: Generates a sequence from the specified **sequence_string** when XMLTABLE is used in SQL queries (that is, **xmltable_clause** is specified in SELECT queries).

  Parameter: **sequence_string**, which is a string in the format of "a to b", indicates that a sequence from number a to number b is generated.

  Return type: numeric sequence of the XMLType.

  Example:

```
-- Generate a sequence.
gaussdb=# SELECT * FROM XMLTABLE ( '/' PASSING XMLQUERY ( '1 to 10' RETURNING SEQUENCE )
COLUMNS N NUMBER PATH '.' );
 n
----
 1
 2
 3
 4
 5
 6
 7
 8
 9
10
(10 rows)
```

  NOTE

  - This function can be used only in Oracle-compatible mode.
  - Only character strings in the format of "a to b" are supported. In this scenario, this function can be used only in XMLTABLE. For details about XMLTABLE, see **xmltable_clause specified in from_item**.
  - The generated sequence is an int64 number ranging from –9223372036854775808 to 9223372036854775807. The maximum number of generated sequences is 2147483647.
  - This function cannot be used during the upgrade observation period.

- xmlquery(xpath_string PASSING document_expression RETURNING CONTENT)

  Description: Parses the XPath expression to locate and return the specified node set or atomic value in the document content of XMLType.

  The following lists related parameters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
xpath_string	XPath expression used for search. • Only the XPath 1.0 syntax is supported. XPath expressions such as **namespace::*** are not supported. • The XQuery expressions are not supported.
document_expression	Document content of XMLType. Currently, only single-root data is supported. The root node of the XML text is fixed in XMLQUERY as the context space where text search is started.

Return type: query result of the XMLType.

Example:

```
-- Calculate the XPath expression query result of the XMLType.
gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA';
CREATE DATABASE
gaussdb=# \c ora_db
ora_db=#
SELECT xmlquery( '/' PASSING XMLTYPE( '<PDRecord> <PDName>Leo</PDName> </PDRecord>')
RETURNING CONTENT);
                xmlquery
-------------------------------------------
 <PDRecord> <PDName>Leo</PDName> </PDRecord>+

(1 row)
ora_db=# \c postgres
gaussdb=# DROP DATABASE ora_db;
DROP DATABASE
```

NOTE

- This function can be used only in Oracle-compatible mode.
- This function cannot be used during the upgrade observation period.

1.6.46 Pivot Table Functions

- tablefunc()

  Description: Extended API for processing table data, including pivot table functions. Only system administrators can install extensions.

  NOTE

  By default, the extension is installed in the public schema. You are advised to install the extension in the user schema by running **create extension tablefunc [schema {user_schema}]**.

- crosstab(source_sql text [, N int])

  Description: Uses the result of **source_sql** as the source data to generate a pivot table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: SETOF record

Example:

```
gaussdb=# CREATE extension tablefunc;
CREATE EXTENSION
gaussdb=# CREATE TABLE cross_test(group_id text, id int, var text);
CREATE TABLE
gaussdb=# SELECT * FROM cross_test;
 group_id | id | var
----------+----+-----
(0 rows)

gaussdb=# SELECT * FROM crosstab('SELECT group_id, var FROM cross_test order by 1, 2;') AS
c(group_ text, cat1 text, cat2 text, cat3 text);
 group_ | cat1 | cat2 | cat3
--------+------+------+------
(0 rows)
```

NOTE

N is an invalid parameter and has no impact on the result of the function.

- crosstabN(source_sql text)

  Description: Uses the result of **source_sql** as the source data to generate a pivot table with "*N*+1" columns. crosstabN is a group of functions, including crosstab2, crosstab3, and crosstab4.

  Return type: setof tablefunc_crosstab_N. tablefunc_crosstab_N includes tablefunc_crosstab_2, tablefunc_crosstab_3, and tablefunc_crosstab_4.

  Example:

  ```
  -- If N in crossbatN(source_sql text) is set to 2, a pivot table with three columns is generated.
  gaussdb=# SELECT * FROM crosstab2('SELECT group_id, var FROM cross_test ORDER BY 1, 2;');
   row_name | category_1 | category_2
  ----------+------------+------------
  (0 rows)
  ```

- crosstab(source_sql text, category_sql text)

  Description: Uses the result of **source_sql** as the source data to generate a pivot table based on the result category of **category_sql**.

  Return type: SETOF record

  Example:

  ```
  gaussdb=# SELECT * FROM crosstab('SELECT group_id, var FROM cross_test order by 1, 2;', 'SELECT
  generate_series(1, 4)') AS c(group_ text, cat1 text, cat2 text, cat3 text, cat4 text);
   group_ | cat1 | cat2 | cat3 | cat4
  --------+------+------+------+------
  (0 rows)

  gaussdb=# DROP TABLE cross_test;
  gaussdb=# DROP EXTENSION tablefunc;
  ```

1.6.47 Global PL/SQL Cache Functions

- invalidate_plsql_object(), invalidate_plsql_object(schema, objname, objtype)

  Description: Invalidates objects in the global PL/SQL cache. This function is available only when **enable_global_plsqlcache** is set to **on**. The current distributed version does not support this function. The user who calls this function must have the SYSADMIN permission.

  Parameters: This function is an overloaded function. If there is no input parameter, all global cache objects in all databases are invalidated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

If the parameters **schema**, **objname**, and **objtype** are specified, the specified global cache object in the current database becomes invalid. **schema** indicates the name of the schema to which the object belongs. **objname** indicates the object name. **objtype** indicates the object type. If the object type is a package, the value is **package**. If the object type is a function or stored procedure, the value is **function**.

- gs_plsql_memory_object_detail(db_oid, obj_oid, obj_type)

  Description: Obtains the valid memory usage of packages and functions created during stored procedures. When **enable_global_plsqlcache** is set to **off**, this function displays the memory usage of stored procedures in the current session. Cross-session query is not supported. The current distributed version does not support this function. The user who calls this function must have the SYSADMIN permission.

  Parameters: This function requires the **db_oid**, **obj_oid**, and **obj_type** input parameters. If the input parameters do not match, an empty line is returned. For details about the parameters, see **Table 1-270**. If the **db_oid** and **obj_oid** input parameters are set to **0**, the valid memory usage of all packages and functions in the cache is queried by default.

  Return type: tuple

Table 1-270 gs_plsql_memory_object_detail parameters

Parameter	Type	Description	Value Range
db_oid	uint32	The valid memory usage of the stored procedure in the database whose OID is **db_oid**. The default value **0** indicates all database instances in the current cache.	0 to 2^32 – 1.
obj_oid	uint32	OID of the queried object. The default value **0** indicates all packages and functions in the current cache.	0 to 2^32 – 1.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description	Value Range
obj_type	text	Type of the queried object. The target object is a package or function.	**all**: queries all object types by default. **pkg**: queries the valid memory usage of packages that meet the conditions. **func**: queries the valid memory usage of functions that meet the conditions. **func_in_pkg**: queries the valid memory usage of functions in the package that meets the conditions.

The gs_plsql_memory_object_detail function displays the valid memory usage that meets the query conditions in the database, as described in **Table 1-271**.

Table 1-271 gs_plsql_memory_object_detail return values

Name	Type	Description
object_oid	uint32	OID of the queried memory object.
context_name	text	Name of a memory object.
item	text	Name of the queried memory object item.
searchpath	text	Schema and other environment variables of the compilation product accessed by the memory object.
guc	uint64	Environment parameter when an object is created, that is, the value of **behavior_compat_flags**.
file	text	File where the memory object is created.
line	uint32	Number of lines in the file where the memory object is created.
size	uint32	Size of a memory object.
db_oid	uint32	OID of the database where the queried memory object is located.

- dump_plsql_compiled_object(object_id, object_type)

  Description: Prints the package and function compilation product information of a stored procedure. If the GUC parameter **enable_global_plsqlcache** is set

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

to **off**, this function can only search for compilation product information in the current session. Cross-session or cross-database query is not supported. The user who calls this function must have the SYSADMIN permission.

Parameters: This function requires the input parameters **object_oid** and **object_type**. If the number or type of the input parameters does not match, an error is reported. **Table 1-272** lists the parameters and their descriptions.

Return type: text.

Table 1-272 dump_plsql_compiled_object parameters

Parameter	Type	Description	Value Range
object_oid	uint32	OID of the queried object.	0 to 2^32 – 1.
object_type	text	Type of the queried object (case-insensitive).	• **"package"**: The object type is package. • **"function"**: The object type is function. (The query effect is equivalent to that of procedure.) • **"procedure"**: The object type is procedure. (The query effect is equivalent to that of function.)

Example:

```
-- When the GUC parameter enable_global_plsqlcache is set to off, create a stored procedure and call dump_plsql_compiled_object to print the compilation product information of the stored procedure.
gaussdb=# \x
Expanded display is on.
gaussdb=# CREATE OR REPLACE PROCEDURE test_proc(a out int, b out int) AS
gaussdb$# BEGIN
gaussdb$# a := 1;
gaussdb$# b := 1;
gaussdb$# END;
gaussdb$# /
CREATE PROCEDURE
gaussdb=# SELECT test_proc();
-[ RECORD 1 ]----
test_proc | (1,1)


gaussdb=# SELECT oid FROM pg_proc WHERE proname = 'test_proc';
-[ RECORD 1 ]
oid | 16388


gaussdb=# SELECT * FROM dump_plsql_compiled_object(16388, 'function');
-[ RECORD 1 ]--------------
+-----------------------
```

```
                    | fn_xmin: 195809
                    | fn_signature: test_proc()
                    | fn_searchpath:{ 2200 , addCatalog: true, addTemp: true, addUser: false}
                    | fn_rettype: 2249
                    | fn_rettyplen: 4294967295
                    | fn_rettypioparam: 0
                    | glc_func_life: 2
                    | fn_retbyval: false
                    | fn_retistuple: true
                    | fn_retset: false
                    | fn_readonly: false
                    | is_private: false
                    | fn_is_trigger: false
                    | pre_parse_trig: false
                    | is_autonomous: false
                    | is_inline_handler: false
                    | is_plpgsql_func_with_outparam: false
                    | need_skip_process_autonm_pkg: false
                    | remembered_by_resowner: false
                    | out_param_varno: 2
                    | found_varno: 3
                    | fn_nallargs: 2
                    | sql_cursor_found_varno: 4
                    | sql_notfound_varno: 5
                    | sql_isopen_varno: 6
                    | sql_rowcount_varno: 7
                    | sql_bulk_exceptions_varno: 0
                    | sqlcode_varno: 8
                    | sqlstate_varno: 9
                    | sqlerrm_varno: 10
                    | retvarno: 0
                    | guc_stat: 0
                    | ref_count: 0
                    | resolve_option: 0
                    | ndatums: 11
                    | datums:{
                    | type: GSPLSQL_DTYPE_VAR, dno: 0 ispkg: false refname: $1 isImplicit: false
addNamespace: false pkg_oid: 0 varname: a isconst: 0,notnull: 0, isnull: true freeval: false
is_place_holder: false isbind: false is_cursor_var: false is_cursor_open: false is_diff_pkg_cursor: false
tableOfIndexType: 0 isIndexByTblOf: false
                    | GSPLSQLType: typname: int4, typoid: 23, ttype: 0, typlen: 4, typrelid: 0, typioparam:
23, collation: 0, typinput_oid: 42, atttypmod: -1, typbyval: true, typtype: b, typnamespace: pg_catalog,
collectionType: 0, tableofOid: 0, tableOfIndexType: 0, arrayTypOid: 0, cursorCompositeOid: 0,
subtypOid: 0, is_subtype: false, notnull: false, temptypmod: -1
                    | default_val: cursor_explicit_expr:
                    | type: GSPLSQL_DTYPE_VAR, dno: 1 ispkg: false refname: $2 isImplicit: false
addNamespace: false pkg_oid: 0 varname: b isconst: 0,notnull: 0, isnull: true freeval: false
is_place_holder: false isbind: false is_cursor_var: false is_cursor_open: false is_diff_pkg_cursor: false
tableOfIndexType: 0 isIndexByTblOf: false
                    | GSPLSQLType: typname: int4, typoid: 23, ttype: 0, typlen: 4, typrelid: 0, typioparam:
23, collation: 0, typinput_oid: 42, atttypmod: -1, typbyval: true, typtype: b, typnamespace: pg_catalog,
collectionType: 0, tableofOid: 0, tableOfIndexType: 0, arrayTypOid: 0, cursorCompositeOid: 0,
subtypOid: 0, is_subtype: false, notnull: false, temptypmod: -1
                    | default_val: cursor_explicit_expr:
                    | type: GSPLSQL_DTYPE_ROW, dno: 2 ispkg: false refname: (null) isImplicit: true
addNamespace: false pkg_oid: 0, varname: (null),
                    | TupleDesc: tdTableAmType: 0, tdisredistable: false, attrs:{ 0: 23 1: 23 }
                    |  tdtypeid: 2249, tdtypmod: -1, tdhasoid: false, tdrefcount: -1, tdhasuids: false,
                    | customErrorCode: 0, intoplaceholders: 0, recordVarTypOid: 0, hasExceptionInit:
false, is_diff_pkg_cursor: false
                    | default_val: nfields: 2, row fields:{ $1 = var 0 $2 = var 1 }
                    | type: GSPLSQL_DTYPE_VAR, dno: 3 ispkg: false refname: found isImplicit: true
addNamespace: true pkg_oid: 0 varname: (null) isconst: 0,notnull: 0, isnull: true freeval: false
is_place_holder: false isbind: false is_cursor_var: false is_cursor_open: false is_diff_pkg_cursor: false
tableOfIndexType: 0 isIndexByTblOf: false
                    | GSPLSQLType: typname: bool, typoid: 16, ttype: 0, typlen: 1, typrelid: 0,
typioparam: 16, collation: 0, typinput_oid: 1242, atttypmod: -1, typbyval: true, typtype: b,
typnamespace: pg_catalog, collectionType: 0, tableofOid: 0, tableOfIndexType: 0, arrayTypOid: 0,
cursorCompositeOid: 0, subtypOid: 0, is_subtype: false, notnull: false, temptypmod: -1
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
                    | default_val: cursor_explicit_expr:
                    | type: GSPLSQL_DTYPE_VAR, dno: 4 ispkg: false refname:
__gsdb_sql_cursor_attri_found__ isImplicit: true addNamespace: true pkg_oid: 0 varname: (null)
isconst: 0,notnull: 0, isnull: true freeval: false is_place_holder: false isbind: false is_cursor_var: false
is_cursor_open: false is_diff_pkg_cursor: false tableOfIndexType: 0 isIndexByTblOf: false
                    | GSPLSQLType: typname: bool, typoid: 16, ttype: 0, typlen: 1, typrelid: 0,
typioparam: 16, collation: 0, typinput_oid: 1242, atttypmod: -1, typbyval: true, typtype: b,
typnamespace: pg_catalog, collectionType: 0, tableofOid: 0, tableOfIndexType: 0, arrayTypOid: 0,
cursorCompositeOid: 0, subtypOid: 0, is_subtype: false, notnull: false, temptypmod: -1
                    | default_val: cursor_explicit_expr:
                    | type: GSPLSQL_DTYPE_VAR, dno: 5 ispkg: false refname:
__gsdb_sql_cursor_attri_notfound__ isImplicit: true addNamespace: true pkg_oid: 0 varname: (null)
isconst: 0,notnull: 0, isnull: true freeval: false is_place_holder: false isbind: false is_cursor_var: false
is_cursor_open: false is_diff_pkg_cursor: false tableOfIndexType: 0 isIndexByTblOf: false
                    | GSPLSQLType: typname: bool, typoid: 16, ttype: 0, typlen: 1, typrelid: 0,
typioparam: 16, collation: 0, typinput_oid: 1242, atttypmod: -1, typbyval: true, typtype: b,
typnamespace: pg_catalog, collectionType: 0, tableofOid: 0, tableOfIndexType: 0, arrayTypOid: 0,
cursorCompositeOid: 0, subtypOid: 0, is_subtype: false, notnull: false, temptypmod: -1
                    | default_val: cursor_explicit_expr:
                    | type: GSPLSQL_DTYPE_VAR, dno: 6 ispkg: false refname:
__gsdb_sql_cursor_attri_isopen__ isImplicit: true addNamespace: true pkg_oid: 0 varname: (null)
isconst: 0,notnull: 0, isnull: true freeval: false is_place_holder: false isbind: false is_cursor_var: false
is_cursor_open: false is_diff_pkg_cursor: false tableOfIndexType: 0 isIndexByTblOf: false
                    | GSPLSQLType: typname: bool, typoid: 16, ttype: 0, typlen: 1, typrelid: 0,
typioparam: 16, collation: 0, typinput_oid: 1242, atttypmod: -1, typbyval: true, typtype: b,
typnamespace: pg_catalog, collectionType: 0, tableofOid: 0, tableOfIndexType: 0, arrayTypOid: 0,
cursorCompositeOid: 0, subtypOid: 0, is_subtype: false, notnull: false, temptypmod: -1
                    | default_val: cursor_explicit_expr:
                    | type: GSPLSQL_DTYPE_VAR, dno: 7 ispkg: false refname:
__gsdb_sql_cursor_attri_rowcount__ isImplicit: true addNamespace: true pkg_oid: 0 varname: (null)
isconst: 0,notnull: 0, isnull: true freeval: false is_place_holder: false isbind: false is_cursor_var: false
is_cursor_open: false is_diff_pkg_cursor: false tableOfIndexType: 0 isIndexByTblOf: false
                    | GSPLSQLType: typname: int4, typoid: 23, ttype: 0, typlen: 4, typrelid: 0, typioparam:
23, collation: 0, typinput_oid: 42, atttypmod: -1, typbyval: true, typtype: b, typnamespace: pg_catalog,
collectionType: 0, tableofOid: 0, tableOfIndexType: 0, arrayTypOid: 0, cursorCompositeOid: 0,
subtypOid: 0, is_subtype: false, notnull: false, temptypmod: -1
                    | default_val: cursor_explicit_expr:
                    | type: GSPLSQL_DTYPE_VAR, dno: 8 ispkg: false refname: sqlcode isImplicit: true
addNamespace: true pkg_oid: 0 varname: (null) isconst: 0,notnull: 0, isnull: true freeval: false
is_place_holder: false isbind: false is_cursor_var: false is_cursor_open: false is_diff_pkg_cursor: false
tableOfIndexType: 0 isIndexByTblOf: false
                    | GSPLSQLType: typname: int4, typoid: 23, ttype: 0, typlen: 4, typrelid: 0, typioparam:
23, collation: 0, typinput_oid: 42, atttypmod: -1, typbyval: true, typtype: b, typnamespace: pg_catalog,
collectionType: 0, tableofOid: 0, tableOfIndexType: 0, arrayTypOid: 0, cursorCompositeOid: 0,
subtypOid: 0, is_subtype: false, notnull: false, temptypmod: -1
                    | default_val: cursor_explicit_expr:
                    | type: GSPLSQL_DTYPE_VAR, dno: 9 ispkg: false refname: sqlstate isImplicit: true
addNamespace: true pkg_oid: 0 varname: (null) isconst: 0,notnull: 0, isnull: true freeval: false
is_place_holder: false isbind: false is_cursor_var: false is_cursor_open: false is_diff_pkg_cursor: false
tableOfIndexType: 0 isIndexByTblOf: false
                    | GSPLSQLType: typname: text, typoid: 25, ttype: 0, typlen: -1, typrelid: 0,
typioparam: 25, collation: 100, typinput_oid: 46, atttypmod: -1, typbyval: false, typtype: b,
typnamespace: pg_catalog, collectionType: 0, tableofOid: 0, tableOfIndexType: 0, arrayTypOid: 0,
cursorCompositeOid: 0, subtypOid: 0, is_subtype: false, notnull: false, temptypmod: -1
                    | default_val: cursor_explicit_expr:
                    | type: GSPLSQL_DTYPE_VAR, dno: 10 ispkg: false refname: sqlerrm isImplicit: true
addNamespace: true pkg_oid: 0 varname: (null) isconst: 0,notnull: 0, isnull: true freeval: false
is_place_holder: false isbind: false is_cursor_var: false is_cursor_open: false is_diff_pkg_cursor: false
tableOfIndexType: 0 isIndexByTblOf: false
                    | GSPLSQLType: typname: text, typoid: 25, ttype: 0, typlen: -1, typrelid: 0,
typioparam: 25, collation: 100, typinput_oid: 46, atttypmod: -1, typbyval: false, typtype: b,
typnamespace: pg_catalog, collectionType: 0, tableofOid: 0, tableOfIndexType: 0, arrayTypOid: 0,
cursorCompositeOid: 0, subtypOid: 0, is_subtype: false, notnull: false, temptypmod: -1
                    | default_val: cursor_explicit_expr:
                    | }
                    | datum_need_free:{ 0: true 1: true 2: true 3: true 4: true 5: true 6: true 7: true 8:
true 9: true 10: true }
                    | namespace_name: public
                    | glc_status:{m_type: GLC_FUNCTION_OBJ, m_location: in_session,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
m_glc_object_state: valid, m_refcount: 0}
                        | fn_nargs: 0
                        | plan_total_mem_size: 30720
                        | dump count: 1
                        | ************************************************************************
                        |
```

1.6.48 UUID Functions

- sys_guid()

  Description: Generates and returns a 16-byte universally unique identifier (UUID). The generated UUID contains the IP address of the current node, generation timestamp, and random number.

  Parameter: none

  Return type: raw

  Example:

```
gaussdb=# SET a_format_version='10c';
SET
gaussdb=# SET a_format_dev_version='s5';
SET
gaussdb=# SELECT sys_guid();
             sys_guid
----------------------------------
 9010675E560CB33C1BDAFA163E378F87
 (1 row)
```

- uuid()

  Description: Returns a UUID defined in RFC 4122, ISO/IEF 9834-8:2005, and related standards. The identifier is a string of lowercase hexadecimal digits, which are divided into several groups: a group of 8 digits, three groups of 4 digits, and a group of 12 digits. A total of 32 digits represent 128 bits.

  Parameter: none

  Return type: varchar

  Example:

```
gaussdb=# SELECT uuid();
                 uuid
--------------------------------------
 dd8cbe92-1a25-013c-a514-e435c87e9182
(1 row)
```

- uuid_short()

  Description: Returns a short UUID under certain conditions. This identifier is a 64-bit unsigned integer.

  The returned value is unique when the following conditions are met:

  – The number of service nodes in the current cluster cannot exceed 256.

  – You cannot set the system time of the server host between node restarts.

  – The average number of uuid_short() calls per second between node restarts is less than 16 million.

  Parameter: none

  Return type: uint64

  Example:

```
gaussdb=# SELECT uuid_short();
    uuid_short
--------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
100440026956955649
(1 row)
```

NOTE

For upgrade from a version to GaussDB Kernel 505.0.0, the uuid_short() function cannot be used if the upgrade is not committed.

1.6.49 SQL Statement Concurrency Control Functions

- gs_add_workload_rule(rule_type, rule_name, databases, start_time, end_time, max_workload, option_val)

  Description: Creates an SQL statement concurrency control rule. Users must have the SYSADMIN permission. This function can be executed only on the CN.

  Parameters: For details, see **Table 1-273**.

  Return type: int8

  Table 1-273 gs_add_workload_rule parameters

Parameter	Type	Description	Value Range
rule_type	text	Type of the concurrency control rule, which is case-insensitive.	**"sqlid"**: Concurrency control is based on the unique SQL ID. **"select"**, **"insert"**, **"update"**, **"delete"**, and **"merge"**: Concurrency control is based on the query type and keyword. **"resource"**: CN process–level concurrency control is based on the system resource usage. **"client"**: Concurrency control is based on client information, including IP addresses and application names. **NOTE** The priorities of concurrency control rules are as follows: client > resource > sqlid > select = insert = update = delete = merge
rule_name	name	Name of a concurrency control rule, which is used to search for the concurrency control rule.	Any character string or **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description	Value Range
databases	name[]	Array of database names for which the concurrency control rule takes effect. The value is case-sensitive.	List of names of created databases. The value can be **NULL**, indicating that the configuration takes effect in all databases. The relationship between multiple databases is OR. Currently, the database list takes effect only when **rule_type** is set to a query type or client type because a unique SQL ID is bound to a database and belongs to only one database. The database is specified when a client is connected. The concurrency control rules based on resource usage take effect for the entire CN process, that is, all databases.
start_time	timestamptz	Start time of a concurrency control rule.	The value can be **NULL**, indicating that it takes effect from now on. **NOTE** When creating a concurrency control rule, you are advised not to use **sysdate** as the start time and end time of the rule in the SQL statement. The precision of **sysdate** is only accurate to seconds, and the fractional part of the second is rounded off. As a result, the concurrency control rule may not take effect in a very short period of time. You are advised to use **now()** instead of **sysdate**.
end_time	timestamptz	End time of a concurrency control rule.	The value can be **NULL** or **''**, indicating that the rule is always effective.
max_workload	int8	Maximum number of concurrent requests set in a concurrency control rule.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description	Value Range
option_val	text[]	Supplementary information about the concurrency control rule.	It matches **rule_type**. The matching relationship is as follows: • **"sqlid"**: specifies the unique ID of the SQL statement whose concurrency is to be controlled and slow SQL control rule. The format is '{id=1234, time_limit=100, max_execute_time=500, max_iops=1}', in which **id** indicates the unique SQL ID and is required. You can obtain it from the dbe_perf.statement or pg_stat_activity view. Other options are optional. For details about their meanings, see "Hint-based Optimization > Hints for Setting Slow SQL Control Rules" in *SQL Optimization Guide*. • **"select"**, **"insert"**, **"update"**, **"delete"**, and **"merge"**: keyword sequence for concurrency control, which is case-insensitive and can be **NULL**. The relationship between multiple keywords is AND. • **"resource"**: resource threshold for traffic limiting. The format is '{cpu-80, memory-70}'. Note that the values of cpu and memory are considered as **0** regardless of the current values. That is, this concurrency control rule is triggered regardless of the actual CPU and memory usage. • **"client"**: information about the client whose concurrency is to be controlled, including the IP address and application name. The format is '{ip=*192.168.1.1*, appname=*app1*}'. At least one of **ip** and **appname** must be specified.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=#  SELECT gs_add_workload_rule('sqlid', 'rule for one query', '{}', now(), NULL, 20,
'{id=32413214}');
 gs_add_workload_rule
----------------------
                    1
(1 row)

gaussdb=#  CREATE DATABASE db1;
gaussdb=#  CREATE DATABASE db2;
gaussdb=#  SELECT gs_add_workload_rule('select', 'rule for select', '{db1, db2}', NULL, NULL, 100,
'{tb1, tb2}');
 gs_add_workload_rule
----------------------
                    2
(1 row)

gaussdb=#  SELECT gs_add_workload_rule('resource', 'rule for resource', '{}', NULL, NULL, 20,
'{cpu-80}');
 gs_add_workload_rule
----------------------
                    3
(1 row)

gaussdb=# SELECT gs_add_workload_rule('client', 'rule for client', '{}', NULL, NULL, 20,
'{appname=app1}');
 gs_add_workload_rule
----------------------
                    4
(1 row)

gaussdb=#  SELECT gs_delete_workload_rule(rule_id) FROM gs_workload_rule; -- Delete all
concurrency control rules.
 gs_delete_workload_rule
-------------------------
 t
 t
 t
 t
(4 rows)

gaussdb=#  DROP DATABASE db1;
gaussdb=#  DROP DATABASE db2;
```

- gs_update_workload_rule(rule_id, rule_name, databases, start_time, end_time, max_workload, option_val)

  Description: To update an SQL statement concurrency control rule, users need to reset all parameters instead of only some parameters. Users must have the SYSADMIN permission. This function can be executed only on the CN.

  Parameters: For details, see **Table 1-274**.

  Return type: Boolean

  Table 1-274 gs_update_workload_rule parameters

Parameter	Type	Description	Value Range
rule_id	int8	ID of the concurrency control rule to be updated.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description	Value Range
rule_name	name	Name of a concurrency control rule, which is used to search for the concurrency control rule.	Any character string or **NULL**.
databases	name[]	Array of database names for which the concurrency control rule takes effect. The value is case-sensitive.	List of names of created databases. The value can be **NULL**, indicating that the configuration takes effect in all databases. The relationship between multiple databases is OR. Currently, the database list takes effect only when **rule_type** is set to a query type or the client type because a unique SQL ID is bound to a database and belongs to only one database. The database is specified when a client is connected. The concurrency control rules based on resource usage take effect for the entire CN process, that is, all databases.
start_time	timestamptz	Start time of a concurrency control rule.	The value can be **NULL**, indicating that it takes effect from now on. **NOTE** When updating a concurrency control rule, you are advised not to use **sysdate** as the start time and end time of the rule in the SQL statement. The precision of **sysdate** is only accurate to seconds, and the fractional part of the second is rounded off. As a result, the concurrency control rule may not take effect in a very short period of time. You are advised to use **now()** instead of **sysdate**.
end_time	timestamptz	End time of a concurrency control rule.	The value can be **NULL**, indicating that the rule is always effective.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description	Value Range
max_workload	int8	Maximum number of concurrent requests set in a concurrency control rule.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description	Value Range
option_val	text[]	Supplementary information about the concurrency control rule.	It matches **rule_type**. The matching relationship is as follows: • **"sqlid"**: specifies the unique ID of the SQL statement whose concurrency is to be controlled and slow SQL control rule. The format is '{id=1234, time_limit=100, max_execute_time=500, max_iops=1}', in which **id** indicates the unique SQL ID and is required. You can obtain it from the dbe_perf.statement or pg_stat_activity view. Other options are optional. For details about their meanings, see "Hint-based Optimization > Hints for Setting Slow SQL Control Rules" in *SQL Optimization Guide*. • **"select"**, **"insert"**, **"update"**, **"delete"**, and **"merge"**: keyword sequence for concurrency control, which is case-insensitive and can be **NULL**. The relationship between multiple keywords is AND. • **"resource"**: resource threshold for traffic limiting. The format is '{cpu-80, memory-70}'. Note that the values of cpu and memory are considered as **0** regardless of the current values. That is, this concurrency control rule is triggered regardless of the actual CPU and memory usage. • **"client"**: information about the client whose concurrency is to be controlled, including the IP address and application name. The format is '{ip=*192.168.1.1*, appname=*app1*}'. At least one of **ip** and **appname** must be specified.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=#  CREATE DATABASE db1;
gaussdb=#  SELECT gs_add_workload_rule('select', 'rule for select', '{db1}', NULL, NULL, 100, '{tb1,
tb2}');
 gs_add_workload_rule
----------------------
                   5
(1 row)

gaussdb=#  SELECT gs_update_workload_rule(5, 'rule for select 2', '{db1}', now(), NULL, 50, '{tb1}');
 gs_update_workload_rule
-------------------------
 t
(1 row)

gaussdb=#  SELECT gs_delete_workload_rule(rule_id) FROM gs_workload_rule; -- Delete all
concurrency control rules.
 gs_delete_workload_rule
-------------------------
 t
(1 row)

gaussdb=#  DROP DATABASE db1;
```

- gs_delete_workload_rule(rule_id)

  Description: Deletes an SQL statement concurrency control rule. Users must have the SYSADMIN permission. This function can be executed only on the CN.

  Parameter: **rule_id** indicates the ID of the concurrency control rule to be deleted. The type is int8.

  Return type: Boolean

  Example:

  ```
  gaussdb=#  SELECT gs_add_workload_rule('sqlid', 'rule for one query', '{}', NULL, NULL, 20,
  '{id=32413214}');
   gs_add_workload_rule
  ----------------------
                     6
  (1 row)

  gaussdb=#  SELECT gs_delete_workload_rule(6);
   gs_delete_workload_rule
  -------------------------
   t
  (1 row)
  ```

- gs_get_workload_rule_stat(rule_id)

  Description: Queries the number of times that SQL statements are blocked by SQL statement concurrency control rules. Users must have the SYSADMIN permission. This function can be executed only on the CN.

  Parameter: **rule_id** indicates the ID of the concurrency control rule to be queried. The type is int8. You can set **rule_id** to **-1**, indicating that all SQL statement concurrency control rules are queried.

  Table 3 Return value types

Name	Type	Description
rule_id	int8	ID of the SQL statement concurrency control rule.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
validate_count	int8	Number of SQL statements intercepted by the SQL statement concurrency control rule.

Example:

```
gaussdb=#  SELECT gs_add_workload_rule('sqlid', 'rule for one query', '{}', now(), NULL, 20,
'{id=32413214}');
 gs_add_workload_rule
----------------------
                    7
(1 row)

gaussdb=#  SELECT gs_add_workload_rule('resource', 'rule for resource', '{}', NULL, NULL, 20,
'{cpu-80}');
 gs_add_workload_rule
----------------------
                    8
(1 row)

gaussdb=#  SELECT * FROM gs_get_workload_rule_stat(7);
 rule_id | validate_count
---------+----------------
       7 |              0
(1 row)

gaussdb=#  SELECT * FROM gs_get_workload_rule_stat(-1);
 rule_id | validate_count
---------+----------------
       7 |              0
       8 |              0
(2 rows)

gaussdb=#  SELECT gs_delete_workload_rule(rule_id) FROM gs_workload_rule;
 gs_delete_workload_rule
-------------------------
 t
 t
(2 rows)
```

- gs_refresh_workload_rule_cache()

  Description: Updates the cache of the concurrency control rule to keep it consistent with the data in the concurrency control rule system catalog. It cannot be executed in a transaction. Users must have the SYSADMIN permission.

  Return type: Boolean

  NOTE

  If a concurrency control rule is added, deleted, or modified in a transaction and the transaction is rolled back, the system catalog is restored to the status before the transaction starts, but the concurrency control rule cache still retains the changed content. As a result, the system catalog is inconsistent with the concurrency control rule cache. In this case, you need to call this function to manually synchronize the data.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.6.50 SQL Tool Functions

The tools are used to manage SQL patches, including creating, deleting, enabling, and disabling SQL patches. Common users have only the USAGE permission and do not have the CREATE, ALTER, DROP, or COMMENT permission.

For details, see "Performing Optimization with SQL Patches" in *SQL Optimization Guide*.

- DBE_SQL_UTIL.create_hint_sql_patch(patch_name, unique_sql_id, hint_string, description, enabled, query_string, is_sql_hash)

  Description: create_hint_sql_patch creates hint SQL patches on the connected CN and returns whether the execution is successful.

  Parameters: For details, see **Table 1-275**.

  Return type: Boolean

  Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

  NOTE

  - SQL patches are not shared among CNs and need to be created on each CN.
  - If load balancing is enabled or a CN needs to be specified, you are advised to use the **DBE_SQL_UTIL.create_remote_hint_sql_patch** API to create a CN.
  - When **query_string** is passed to an EXECUTE statement, the system obtains the PREPARE statement corresponding to the current session and creates an SQL patch. If there is no corresponding prepared statement, the operation is invalid.

  Table 1-275 DBE_SQL_UTIL.create_hint_sql_patch input parameters and return values

Parameter	Type	Description
patch_name	IN name	Patch name.
unique_sql_id	IN bigint	Globally unique ID.
hint_string	IN text	Hint text.
description	IN text	Patch description. The default value is **NULL**.
enabled	IN bool	Specifies whether the patch takes effect. The default value is **true**.
query_string	IN text	Original SQL statement corresponding to the SQL hash. The default value is **NULL**.
is_sql_hash	IN bool	Specifies whether the patch is created based on the SQL hash value. The default value is **false**.
result	OUT bool	Specifies whether the execution is successful.

- DBE_SQL_UTIL.create_hint_sql_patch(patch_name, unique_sql_id, parent_unique_sql_id, hint_string, description, enabled)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: create_hint_sql_patch creates hint SQL patches on the connected CN and returns whether the execution is successful. This function is an overloaded function of the original function. It allows you to restrict the scope of the hint patch by specifying the value of **parent_unique_sql_id**.

Parameters: For details, see **Table 1-276**.

Return type: Boolean

Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

Table 1-276 Input parameters and return values of the overloaded function of DBE_SQL_UTIL.create_hint_sql_patch

Parameter	Type	Description
patch_name	IN name	Patch name.
unique_sql_id	IN bigint	Globally unique ID.
parent_unique_sql_id	IN bigint	Globally unique ID of the outer SQL statement, indicating the scope where the SQL patch takes effect. • **0**: The SQL patch takes effect only in the outer SQL statement. • Non-zero value: The SQL patch takes effect only in a specific stored procedure.
hint_string	IN text	Hint text.
description	IN text	Patch description. The default value is **NULL**.
enabled	IN bool	Specifies whether the patch takes effect. The default value is **true**.
result	OUT bool	Specifies whether the execution is successful.

- DBE_SQL_UTIL.create_remote_hint_sql_patch(node_name, patch_name, unique_sql_id, hint_string, description, enabled, query_string, is_sql_hash)

  Description: create_remote_hint_sql_patch creates hint SQL patches on a specified CN and returns whether the execution is successful.

  Parameters: For details, see **Table 1-277**.

  Return type: Boolean

  Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

  NOTE

  When **query_string** is passed to an EXECUTE statement, the system obtains the PREPARE statement corresponding to the current session and creates an SQL patch. If there is no corresponding prepared statement, the operation is invalid.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-277 DBE_SQL_UTIL.create_remote_hint_sql_patch input parameters and return values

Parameter	Type	Description
node_name	IN text	Name of the target node. If the value is **ALL**, SQL patches are created for all CNs.
patch_name	IN name	Patch name.
unique_sql_id	IN bigint	Globally unique ID.
hint_string	IN text	Hint text.
description	IN text	Patch description. The default value is **NULL**.
enabled	IN bool	Specifies whether the patch takes effect. The default value is **true**.
query_string	IN text	Original SQL statement corresponding to the SQL hash. The default value is **NULL**.
is_sql_hash	IN bool	Specifies whether the patch is created based on the SQL hash value. The default value is **false**.
result	OUT bool	Specifies whether the execution is successful.

- DBE_SQL_UTIL.create_abort_sql_patch(patch_name, unique_sql_id, description, enabled, query_string, is_sql_hash)

  Description: create_abort_sql_patch creates abort SQL patches on the connected CN and returns whether the execution is successful.

  Parameters: For details, see **Table 1-278**.

  Return type: Boolean

  Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

  NOTE

  - SQL patches are not shared among CNs and need to be created on each CN.
  - If load balancing is enabled or a CN needs to be specified, you are advised to use the **DBE_SQL_UTIL.create_remote_abort_sql_patch** API to create a CN.
  - When **query_string** is passed to an EXECUTE statement, the system obtains the PREPARE statement corresponding to the current session and creates an SQL patch. If there is no corresponding prepared statement, the operation is invalid.

Table 1-278 DBE_SQL_UTIL.create_abort_sql_patch input parameters and return values

Parameter	Type	Description
patch_name	IN name	Patch name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
unique_sql_id	IN bigint	Globally unique ID.
description	IN text	Patch description. The default value is **NULL**.
enabled	IN bool	Specifies whether the patch takes effect. The default value is **true**.
query_string	IN text	Original SQL statement corresponding to the SQL hash. The default value is **NULL**.
is_sql_hash	IN bool	Specifies whether the patch is created based on the SQL hash value. The default value is **false**.
result	OUT bool	Specifies whether the execution is successful.

- DBE_SQL_UTIL.create_abort_sql_patch(patch_name, unique_sql_id, parent_unique_sql_id, description, enabled)

  Description: create_abort_sql_patch creates abort SQL patches on the connected CN and returns whether the execution is successful. This function is an overloaded function of the original function. It allows you to restrict the scope of the abort SQL patch by specifying the value of **parent_unique_sql_id**.

  Parameters: For details, see **Table 1-279**.

  Return type: Boolean

  Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

  Table 1-279 Input parameters and return values of the overloaded function of DBE_SQL_UTIL.create_abort_sql_patch

Parameter	Type	Description
patch_name	IN name	Patch name.
unique_sql_id	IN bigint	Globally unique ID.
parent_unique_sql_id	IN bigint	Globally unique ID of the outer SQL statement, indicating the scope where the SQL patch takes effect. • **0**: The SQL patch takes effect only in the outer SQL statement. • Non-zero value: The SQL patch takes effect only in a specific stored procedure.
description	IN text	Patch description. The default value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
enabled	IN bool	Specifies whether the patch takes effect. The default value is **true**.
result	OUT bool	Specifies whether the execution is successful.

- DBE_SQL_UTIL.create_remote_abort_sql_patch(node_name, patch_name, unique_sql_id, description, enabled, query_string, is_sql_hash)

  Description: create_remote_abort_sql_patch creates abort SQL patches on the connected CN and returns whether the execution is successful.

  Parameters: For details, see **Table 1-280**.

  Return type: Boolean

  Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

  NOTE

  When **query_string** is passed to an EXECUTE statement, the system obtains the PREPARE statement corresponding to the current session and creates an SQL patch. If there is no corresponding prepared statement, the operation is invalid.

  Table 1-280 DBE_SQL_UTIL.create_remote_abort_sql_patch input parameters and return values

Parameter	Type	Description
node_name	IN text	Name of the target node. If the value is **ALL**, SQL patches are created for all CNs.
patch_name	IN name	Patch name.
unique_sql_id	IN bigint	Globally unique ID.
description	IN text	Patch description. The default value is **NULL**.
enabled	IN bool	Specifies whether the patch takes effect. The default value is **true**.
query_string	IN text	Original SQL statement corresponding to the SQL hash. The default value is **NULL**.
is_sql_hash	IN bool	Specifies whether the patch is created based on the SQL hash value. The default value is **false**.
result	OUT bool	Specifies whether the execution is successful.

- DBE_SQL_UTIL.drop_sql_patch(patch_name)

  Description: drop_sql_patch deletes SQL patches on the connected CN and returns whether the execution is successful.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: For details, see **Table 1-281**.

Return type: Boolean

Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

NOTE

- SQL patches are not shared among CNs and need to be executed on each CN.
- If load balancing is enabled or a CN needs to be specified, you are advised to use the **DBE_SQL_UTIL.drop_remote_sql_patch** API to delete a CN.

Table 1-281 DBE_SQL_UTIL.drop_sql_patch input parameters and return values

Parameter	Type	Description
patch_name	IN name	Patch name.
result	OUT bool	Specifies whether the execution is successful.

- DBE_SQL_UTIL.drop_remote_sql_patch(node_name, patch_name)

  Description: drop_remote_sql_patch deletes SQL patches from a specified CN and returns whether the execution is successful.

  Parameters: For details, see **Table 1-282**.

  Return type: Boolean

  Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

  Table 1-282 DBE_SQL_UTIL.drop_remote_sql_patch input parameters and return values

Parameter	Type	Description
node_name	IN text	Target node name.
patch_name	IN name	Patch name.
result	OUT bool	Specifies whether the execution is successful.

- DBE_SQL_UTIL.enable_sql_patch(patch_name)

  Description: enable_sql_patch enables SQL patches on the connected CN and returns whether the execution is successful.

  Parameters: For details, see **Table 1-283**.

  Return type: Boolean

  Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- SQL patches are not shared among CNs and need to be executed on each CN.
- If load balancing is enabled or a CN needs to be specified, you are advised to use the **DBE_SQL_UTIL.enable_remote_sql_patch** API to enable SQL patches.

Table 1-283 DBE_SQL_UTIL.enable_sql_patch input parameters and return values

Parameter	Type	Description
patch_name	IN name	Patch name.
result	OUT bool	Specifies whether the execution is successful.

- DBE_SQL_UTIL.enable_remote_sql_patch(node_name, patch_name)

  Description: enable_remote_sql_patch enables SQL patches on a specified CN and returns whether the execution is successful.

  Parameters: For details, see **Table 1-284**.

  Return type: Boolean

  Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

  Table 1-284 DBE_SQL_UTIL.enable_remote_sql_patch input parameters and return values

Parameter	Type	Description
node_name	IN text	Target node name.
patch_name	IN name	Patch name.
result	OUT bool	Specifies whether the execution is successful.

- DBE_SQL_UTIL.disable_sql_patch(patch_name)

  Description: disable_sql_patch disables SQL patches on the connected CN and returns whether the execution is successful.

  Parameters: For details, see **Table 1-285**.

  Return type: Boolean

  Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

  NOTE

  - SQL patches are not shared among CNs and need to be executed on each CN.
  - If load balancing is enabled or a CN needs to be specified, you are advised to use the **DBE_SQL_UTIL.disable_remote_sql_patch** API to disable SQL patches.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-285 DBE_SQL_UTIL.disable_sql_patch input parameters and return values

Parameter	Type	Description
patch_name	IN name	Patch name.
result	OUT bool	Specifies whether the execution is successful.

- DBE_SQL_UTIL.disable_remote_sql_patch(node_name, patch_name)

  Description: disable_remote_sql_patch disables SQL patches on a specified CN and returns whether the execution is successful.

  Parameters: For details, see **Table 1-286**.

  Return type: Boolean

  Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

  Table 1-286 DBE_SQL_UTIL.disable_remote_sql_patch input parameters and return values

Parameter	Type	Description
node_name	IN text	Target node name.
patch_name	IN name	Patch name.
result	OUT bool	Specifies whether the execution is successful.

- DBE_SQL_UTIL.show_sql_patch(patch_name, unique_sql_id, status, enabled, abort)

  Description: show_sql_patch displays the SQL patch corresponding to a specified patch name and returns the running result.

  Parameters: For details, see **Table 1-287**.

  Return type: record.

  Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

  Table 1-287 DBE_SQL_UTIL.show_sql_patch input parameters and return values

Parameter	Type	Description
patch_name	IN name	Patch name.
unique_sql_id	OUT bigint	Globally unique ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
status	OUT char	Specifies whether the patch is created based on a unique SQL ID or SQL hash value. • **'d'**: unique SQL ID. • **'h'**: SQL hash value.
enabled	OUT bool	Specifies whether the patch takes effect.
abort	OUT bool	Specifies whether the patch is an abort hint.
hint_str	OUT text	Hint text.

- DBE_SQL_UTIL.sqlpatch_query_hash(query_string, need_validate)

  Description: sqlpatch_query_hash obtains the SQL hash value of a specified SQL statement.

  Parameters: For details, see **Table 1-288**.

  Return type: int8

  Only the initial user, SYSADMIN, OPRADMIN, and MONADMIN have the permission to call it.

  NOTE

  When the **cursor_sharing** parameter is set to **force**, the SQL hash value is calculated based on the parameterized SQL text. In this case, you need to pass the parameterized SQL text and set **need_validate** to **false** to obtain the SQL hash value after automatic parameterization. For example: SELECT DBE_SQL_UTIL.sqlpatch_query_hash('SELECT * FROM t WHERE a =?', false);

  Table 1-288 DBE_SQL_UTIL.sqlpatch_query_hash input parameters and return values

Parameter	Type	Description
query_string	IN text	SQL text.
need_validate	IN boolean	Specifies whether to perform syntax check.
result	OUT bigint	Hash value corresponding to the SQL text.

- DBE_SQL_UTIL.gs_show_plan_by_hash(sql_hash, plan_hash)

  Description: In this version, if the gs_show_plan_by_hash function is used, a message is returned indicating that the function is not supported in the current form.

  Parameters: For details, see **Table 1-289**.

  Return type: text.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-289 DBE_SQL_UTIL.gs_show_plan_by_hash input parameters and return values

Parameter	Type	Description
sql_hash	IN bigint	Hash value of SQL text.
plan_hash	IN bigint	Hash value of the SQL plan.
gs_show_plan_by_hash	OUT text	The returned result is a message indicating that the function is not supported in the current form.

1.6.51 Multi-Tenant Database Functions

The following functions are used for the multi-tenant database feature and are not supported in the distributed system: The following functions are not supported in the distributed system:

- gs_reload_pdb_conf(name pdbname)
- gs_resplan_cgroup_info()
- gs_resplan_stat_info(view_type integer)
- gs_get_pdb_tablespace_location(pdb_id oid in, tablespace_oid oid in)
- get_instr_db_rt_percentile()
- get_instr_user_login_info()
- gs_resplan_shared_memory_info()

The following functions are used in internal system views and are not recommended in the distributed system:

- get_mtd_user()
- get_instr_sql_count_info()

1.6.52 Global Temporary Table Functions

- pg_get_gtt_relstats(relOid)

  Description: Displays basic information about a global temporary table specified by the current session. The query result of the distributed global temporary table on a CN is not meaningful, and the query result on a DN is empty. Therefore, it is not recommended in distributed mode.

  Parameter: OID of the global temporary table

  Return type: record

  Example:

```
gaussdb=# SELECT * FROM pg_get_gtt_relstats(24576);
 relfilenode | relpages | reltuples | relallvisible | relfrozenxid | relminmxid
-------------+----------+-----------+---------------+--------------+------------
       24576 |        0 |         0 |             0 |        53969 |          0
(1 row)
```

- pg_get_gtt_statistics(relOid, attnum, ''::text)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Displays statistics about a single column in a global temporary table specified by the current session. Before using this function, perform ANALYZE on the table.

Parameter: OID and the **attnum** attribute of the global temporary table

Return type: record

Example:

```
gaussdb=# SELECT * FROM  pg_get_gtt_statistics(24576, 1,''::text);
 starelid | starelkind | staattnum | stainherit | stanullfrac | stawidth | stadistinct | stakind1 | stakind2 |
stakind3 | stakind4 | stakind5 | staop1 | staop2 | staop3 | staop4 | staop5 | stanumbers1 | stanumbers2
| stanumbers3 | stanumbers
4 | stanumbers5
|
                                  stavalues1

          | stavalues2 | stavalues3 | stavalues4 | stavalues5 | stadndistinct | staextinfo

----------+------------+-----------+------------+-------------+----------+-------------+----------+----------
+----------+----------+----------+--------+--------+--------+--------+--------+-------------+-------------
+-------------+-----------
--+-------------
+-----------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------
-----------------------------------------+------------+------------+------------+------------+---------------
+-----------
-
   24576 | c          |         1 | f          |           0 |        4 |          -1 |        2 |        3 |        0 |        0 |
0 |     97 |     97 |      0 |      0 |      0 |            | {.495105}   |             |
  |             |
{1,5,10,15,20,25,30,35,40,45,50,55,60,65,70,75,80,85,90,95,100,105,110,115,120,125,130,135,140,145,15
0,155,160,165,170,175,180,185,190,195,200,205,210,215,220,225,230,235,240,245,250,255,260,265,270,
275,280,285,290,295,300,
305,310,315,320,325,330,335,340,345,350,355,360,365,370,375,380,385,390,395,400,405,410,415,420,4
25,430,435,440,445,450,455,460,465,470,475,480,485,490,495,500} |             |             |             |
|             1 |
(1 row)
```

- pg_list_gtt_relfrozenxids()

  Description: Displays the frozen XID of each session. If the value of **pid** is **0**, the earliest frozen XID of all sessions is displayed. In distributed mode, gs_global_list_gtt_relfrozenxids() is recommended.

  Parameter: none

  Return type: record

  Example:

  ```
  gaussdb=# SELECT * FROM pg_list_gtt_relfrozenxids();
        pid        | relfrozenxid
  -----------------+--------------
   139988750038784 |        53969
                 0 |        53969
  (2 rows)
  ```

- pg_gtt_attached_pid(relOid)

  Description: Displays PIDs and session IDs of all threads that are using the specified global temporary table. If the thread pool is enabled and the session that is using the global temporary table is in the detach state, the PID is **0** and **sessionid** indicates the session ID. In distributed mode, gs_gtt_attached_pid(relOid) is recommended.

  Parameter: OID of the global temporary table

  Return type: record

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT * FROM pg_gtt_attached_pid(24576);
 relid |       pid       | sessionid
-------+-----------------+-----------
 24576 | 139988750038784 |         1
(1 row)
```

- gs_gtt_attached_pid(relOid)

  Description: Displays the IDs of all sessions that are using a specified distributed global temporary table. This is executed on DNs, but not supported on CNs.

  Parameter: OID of the global temporary table

  Return type: record

  Example:

  ```
  gaussdb=# SELECT * FROM gs_gtt_attached_pid(16384);
   relid |       pid       |   cn_name    | cn_timeline | cn_sequence | cn_session_id
  -------+-----------------+--------------+-------------+-------------+---------------
   16384 | 140648560719616 | coordinator1 |           2 |           1 |             1
  (1 row)
  ```

- gs_global_list_gtt_relfrozenxids()

  Description: Displays XIDs of frozen transactions on each DN. This is executed on CNs. PID is **–1** when the current session is not joined with a global temporary table. This cannot be executed on DNs.

  Parameter: none

  Return type: record

  Example:

  ```
  gaussdb=# SELECT * FROM gs_global_list_gtt_relfrozenxids();
   node_name |       pid       | relfrozenxid
  -----------+-----------------+--------------
   datanode1 | 140648560719616 |        53959
   datanode1 |               0 |        53959
   datanode2 | 140539996403456 |        53959
   datanode2 |               0 |        53959
  (4 rows)
  ```

- gs_local_gtt_clean_session(clean_type, clean_obj, clean_table)

  Description: Clears global temporary tables joined with global sessions on CNs. This is executed on DNs, but not supported on CNs.

  The following lists related parameters.

Name	Type	Description
clean_type	Enumeration	• **0**: All session mappings whose timeline is earlier than the current timeline are cleared. • **1**: The session mapping joined with a specified CN session is cleared. • **2**: Data of a specified table in all sessions is cleared. (This is for internal use only and cannot be executed by users.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

clean_obj	string	Constructor: • coorname_timeline_seq_sess (when **clean_type** is set to **1**) • coorname_timeline (when **clean_type** is set to **0**) • schema name (when **clean_type** is set to **2**)
clean_table	string	If **clean_type** is set to **2**, the value is a table name. If **clean_type** is set to other values, the value is null.

Return type: void

Example:

Use gs_local_get_cn_session_key on a CN to obtain the latest **cn_session_key**.

```
gaussdb=# SELECT * FROM gs_local_get_cn_session_key();
   cn_name    | cn_timeline | cn_sequence | cn_session_id
--------------+-------------+-------------+---------------
 coordinator1 |           2 |           1 |             1
(1 row)
```

Use gs_local_dump_gtt_sess_map to obtain the residual **cn_session_key** on a DN that needs to be manually cleared.

```
gaussdb=# SELECT * FROM gs_local_dump_gtt_sess_map();
   cn_name    | cn_timeline | cn_sequence | cn_session_id | uniq_sess_id | refcnt | database_id |
current_database_id
--------------+-------------+-------------+---------------+--------------+--------+-------------
+----------------------
 coordinator1 |           1 |           1 |             1 |            1 |      0 |       13771 |              13771
(1 row)
```

Clear the residual **cn_session_key** that meets the following conditions:

– timeline < CN timeline

– refcnt = 0

```
gaussdb=# SELECT * FROM gs_local_gtt_clean_session(0, 'coordinator1_2', '');
 gs_local_gtt_clean_session
----------------------------


(1 row)
```

- gs_global_dist_gtt_clean_remain_sessions()

  Description: Clears all residual session information in a global temporary table. This is executed on CNs, but not supported on DNs.

  Parameter: none

  Return type: void

  Example:

  ```
  gaussdb=# SELECT * FROM gs_global_dist_gtt_clean_remain_sessions();
   gs_global_dist_gtt_clean_remain_sessions
  ------------------------------------------


  (1 row)
  ```

- gs_local_get_cn_session_key()

  Description: Displays the session key of the current session. This is executed on CNs. This cannot be executed on DNs.

  Parameter: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: record

Example:

```
gaussdb=# SELECT * FROM gs_local_get_cn_session_key();
   cn_name    | cn_timeline | cn_sequence | cn_session_id
--------------+-------------+-------------+---------------
 coordinator1 |           2 |           1 |             1
(1 row)
```

- gs_local_dump_gtt_sess_map()

  Description: Displays the unique session ID **cn_session_key** of a distributed global temporary table stored on a DN. The ID consists of the following columns: **cn_name**, **cn_timeline**, **cn_sequence**, **cn_session_id**, **uniq_sess_id**, **refcnt**, **database_id**, and **current_database_id**. This is executed on DNs, but not supported on CNs.

  Parameter: none

  Return type: record

  Example:

```
gaussdb=# SELECT * FROM gs_local_dump_gtt_sess_map();
   cn_name    | cn_timeline | cn_sequence | cn_session_id | uniq_sess_id | refcnt | database_id |
current_database_id
--------------+-------------+-------------+---------------+--------------+--------+-------------
+---------------------
 coordinator1 |           1 |           1 |             1 |            1 |      0 |       13771 |             13771
(1 row)
```

- gs_global_gtt_clean_data(schema_name, table_name)

  Description: Clears the data files of the specified global temporary table in all sessions. This is for internal use only and cannot be executed by users.

  The following lists related parameters.

Name	Type	Description
schema_name	string	Name of a schema to be cleared.
table_name	string	Name of a table to be cleared.

  Return type: void

  Example:

```
gaussdb=# SELECT gs_global_gtt_clean_data('public', 'gtt');
ERROR:  The input parameter is invalid.
CONTEXT:  referenced column: gs_local_gtt_clean_session
SQL statement "EXECUTE DIRECT ON DATANODES 'SELECT pg_catalog.gs_local_gtt_clean_session(2,
''public'', ''gtt'')'"
PL/pgSQL function gs_global_gtt_clean_data(text,text) line 6 at EXECUTE statement
referenced column: gs_global_gtt_clean_data
```

- gs_gtt_local_clean_cn_remain_session_keys()

  Description: Clears residual data in global temporary tables on other DNs when a DN is faulty. This is executed on CNs, not supported on DNs.

  Return type: void

```
gaussdb=# select * from gs_gtt_local_clean_cn_remain_session_keys();
 gs_gtt_local_clean_cn_remain_session_keys
-------------------------------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)
```

1.6.53 Database Link Functions

- close_database_link(dblink_name name)

  Description: Closes a database link in the current session based on the database link name. A database link with active transactions cannot be closed.

  Table 1-290 close_database_link(dblink_name name) parameters

Name	Type	Input/ Output	Description
dblink_name	name	Input	Name of a database link.
close_database_link	boolean	Output	Result of closing the DATABASE LINK (**t**: success; **f**: failure).

  Example:

```
gaussdb=# SELECT close_database_link('xxx');
 close_database_link
---------------------
 t
(1 row)

gaussdb=# BEGIN;
BEGIN
gaussdb=# SELECT * FROM t1@dblink;
c1 | c2 | c3
----+----+----
1 | |
(1 row)
gaussdb=# INSERT INTO t1@dblink VALUES (2,0,0);
INSERT 0 1
gaussdb=# SELECT * FROM t1@dblink;
c1 | c2 | c3
----+----+----
1 | |
2 | 0 | 0
(2 rows)
gaussdb=# SELECT close_database_link('dblink');
ERROR: Dblink dblink is in use.
CONTEXT: referenced column: close_database_link
gaussdb=# END;
ROLLBACK
```

- gs_ora_diag(dblink_name name DEFAULT NULL::name)

  Description: Obtains information about the database link that connects to the Oracle Database. The following information is obtained based on the database link name: gs_ora_fdw version, GaussDB version, Oracle Database's client version, and Oracle Database's server version. If the parameter is set to **NULL** and the query user is an administrator, information such as ORACLE_HOME, ORACLE_SID, TNS_ADMIN, TWO_TASK, and LDAP_ADMIN is also returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-291 gs_ora_diag(dblink_name name DEFAULT NULL::name) parameters

Name	Type	Input/ Output	Description
dblink_name	name	Input	Name of a database link.
gs_ora_diag	text	Output	Diag information of a database link.

Example:

```
-- Administrator.
gaussdb=# SELECT gs_ora_diag('xxx');
                                  gs_ora_diag
-------------------------------------------------------------------------------
 gs_ora_fdw 1.0.0, GaussDB 9.2.4, Oracle client 19.24.0.0.0, Oracle server 19.0.0.0.0
(1 row)
gaussdb=# SELECT gs_ora_diag();

gs_ora_diag
-------------------------------------------------------------------------------------------------------------------------------
-----------------------------
 gs_ora_fdw 1.0.0, GaussDB 9.2.4, Oracle client 19.24.0.0.0, ORACLE_HOME=/xxx/instantclient_19_24,
TNS_ADMIN=/xxx/instantclient_19_24/network/admin
(1 row)

-- Common user.
gaussdb=# SELECT gs_ora_diag('xxx');
                                  gs_ora_diag
-------------------------------------------------------------------------------
 gs_ora_fdw 1.0.0, GaussDB 9.2.4, Oracle client 19.24.0.0.0, Oracle server 19.0.0.0.0
(1 row)

gaussdb=# SELECT gs_ora_diag();
                      gs_ora_diag
-------------------------------------------------------------
 gs_ora_fdw 1.0.0, GaussDB 9.2.4, Oracle client 19.24.0.0.0
(1 row)
```

- gs_ora_get_handle_alloc(dblink_oid oid)

  Description: Obtains the size (in bytes) of the memory used by the database link through the OCI based on the OID of the database link.

Table 1-292 gs_ora_get_handle_alloc(dblink_oid oid) parameters

Name	Type	Input/ Output	Description
dblink_oid	oid	Input	OID of the specified database link.
gs_ora_get_handle_alloc	numeric	Output	Size of the memory used by the database link through the OCI, in bytes.

Example:

```
gaussdb=# SELECT gs_ora_get_handle_alloc(xxx);
 gs_ora_get_handle_alloc
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------------------------
                151224
(1 row)
```

- gs_ora_get_handle_alloc_all()

  Description: Obtains the environment handle of each node and calculates the total memory size used by all database links through OCI, in bytes.

  Table 1-293 gs_ora_get_handle_alloc_all() parameters

Name	Type	Input/ Output	Description
gs_ora_get_handle_alloc_all	numeric	Output	Total size of memory used by all database links through OCI, in bytes.

  Example:

```
gaussdb=# SELECT gs_ora_get_handle_alloc_all();
 gs_ora_get_handle_alloc_all
-----------------------------
                      429000
(1 row)
```

- gs_ora_connection_status(dblink_oid oid)

  Description: Obtains the connection status of a database link in the current session based on the OID of the database link.

  Table 1-294 gs_ora_connection_status(dblink_oid oid) parameters

Name	Type	Input/ Output	Description
dblink_oid	oid	Input	OID of the specified database link.
gs_ora_connection_status	text	Output	Connection status of the database link in the current session (**CONNECTED**: connected; **LOST**: disconnected).

  Example:

```
gaussdb=# SELECT gs_ora_connection_status(xxx);
 gs_ora_connection_status
--------------------------
 CONNECTED
(1 row)
```

- gs_ora_fdw_handler()

  Description: The processing function of the foreign data wrapper oracle_fdw, which returns an fdw_handler structure containing a pointer to the callback function.

  Input parameter: none

  Return type: fdw_handler

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_ora_fdw_validator(option text[], dblink_oid oid)

  Description: The check function of the foreign data wrapper gs_ora_fdw, which inspects the common options provided for gs_ora_fdw. If the option or option value is invalid or missing, an error is reported.

  Table 1-295 gs_ora_fdw_validator(option text[], dblink_oid oid) parameters

Name	Type	Input/ Output	Description
dblink_oid	oid	Input	OID of the specified database link.
option	text[]	Input	Option of a database link.

  Return type: void

- is_dblink_in_use(dblink_oid oid)

  Description: Obtains whether a database link is used based on the OID of the database link.

  Table 1-296 is_dblink_in_use(dblink_oid oid) parameters

Name	Type	Input/Output	Description
dblink_oid	oid	Input	OID of the specified database link.
is_dblink_in_use	bool	Output	Usage status of the database link.

  Example:

```
gaussdb=# SELECT is_dblink_in_use(xxx);
 is_dblink_in_use
------------------------------
 f
(1 row)
```

- dblink_has_updatasent (dblink_oid oid)

  Description: Checks whether a DML statement has been sent through the database link corresponding to an OID in the current transaction but has not been committed yet.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-297 dblink_has_updatasent (dblink_oid oid) parameters

Name	Type	Input/Output	Description
dblink_oid	oid	Input	OID of the specified database link.
dblink_has_updatasent	bool	Output	Specifies whether the database link has sent DML statements and is in the uncommitted phase.

Example:

```
gaussdb=# SELECT dblink_has_updatasent(xxx);
 dblink_has_updatasent
-----------------------------
 f
(1 row)
```

- is_dblink_in_transaction(dblink_oid oid)

  Description: Obtains whether a database link is in a transaction based on the OID of the database link.

  Table 1-298 is_dblink_in_transaction(dblink_oid oid) parameters

Name	Type	Input/Output	Description
dblink_oid	oid	Input	OID of the specified database link.
is_dblink_in_transaction	bool	Output	Specifies whether the database link is in a transaction.

  Example:

```
gaussdb=# SELECT is_dblink_in_transaction(xxx);
 is_dblink_in_transaction
-----------------------------
 f
(1 row)
```

1.6.54 GSMonitor System Resource Statistics Functions

- gs_get_cpustat(start_time timestamptz, end_time timestamptz)

  Description: Queries CPU resource information within a period of time. To call this function, you must have the MONADMIN or SYSADMIN permission.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The following describes the parameters.

Parameter	Description	Value Range
start_time	Required. Start time of the query.	Timestamptz type.
end_time	Required. End time of the query.	Timestamptz type.

Return type: SETOF record

The following describes the return columns.

Column	Type	Description
collection_time	timestamptz	Collection time.
cpu_name	text	Name.
%total	float8	Total: CPU usage (user + kernel).
%user	float8	User mode: CPU usage (user space).
%system	float8	System mode: CPU usage (kernel space).
%idle	float8	CPU usage in the idle state.
%iowait	float8	Percentage of CPU resources waiting for I/Os.
lavg_1	float8	One-minute load (unit: count).
lavg_5	float8	Five-minute load (unit: count).
lavg_15	float8	15-minute load (unit: count).

- gs_get_procstat(start_time timestamptz, end_time timestamptz, sort_order text[, display_type text])

  Description: Queries process information within a period of time. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following describes the parameters.

Parameter	Description	Value Range
start_time	Required. Start time of the query.	Timestamptz type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range
end_time	Required. End time of the query.	Timestamptz type.
sort_order	Required. Sorting rules.	Text type. The value can be **'IO_R'**, **'IO_W'**, **'CPU'** or **'MEM'** (case-insensitive). • **'IO_R'**: sorts data by I/O read rate in descending order. • **'IO_W'**: sorts data by I/O write rate in descending order. • **'CPU'**: sorts data by CPU usage in descending order. • **'MEM'**: sorts data by used memory size in descending order.
display_type	Optional. Display mode.	Text type. Currently, only **'pretty'** (case-insensitive) is supported, indicating that the affiliation relationship is displayed.

Return type: SETOF record

The following describes the return columns.

Column	Type	Description
collection_time	timestamptz	Collection time.
pid	int8	Process name.
tid	int8	Thread name.
state	text	Status.
ppid	int8	Parent process ID.
priority	int8	Priority (dynamic).
nice	int8	Priority (static).
num_threads	int8	Number of threads.
utime	float8	User-mode runtime (unit: jiffies).
stime	float8	Kernel-mode runtime (unit: jiffies).
ssize	int8	Size of the shared memory (unit: byte).
vsize	int8	Virtual memory size of a process (unit: byte).
io_r	float8	I/O read rate (unit: operations/s).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
io_w	float8	I/O write rate (unit: operations/s).
io	float8	Total I/O rate (unit: operations/s).
rss	int8	Size of the resident set used by a process (unit: page).
%cpu_usage	float8	CPU usage.
comm	text	Command name of a process.

- gs_get_iostat(start_time timestamptz, end_time timestamptz[, display_type text])

  Description: Queries I/O resource information within a period of time. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following describes the parameters.

Parameter	Description	Value Range
start_time	Required. Start time of the query.	Timestamptz type.
end_time	Required. End time of the query.	Timestamptz type.
display_type	Optional. Display mode.	Text type. Currently, only **'pretty'** (case-insensitive) is supported, indicating that the affiliation relationship is displayed.

  Return type: SETOF record

  The following describes the return columns.

Column	Type	Description
collection_time	timestamptz	Collection time.
devname	text	Device name.
rkB/s	float8	I/O read rate (unit: KB/s).
wkB/s	float8	I/O write rate (unit: KB/s).
r/s	float8	I/O read operations per second (unit: operations/s).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
w/s	float8	I/O write operations per second (unit: operations/s).
rawait	float8	Average wait time of read operations (unit: ms).
wawait	float8	Average wait time of write operations (unit: ms).
svctm	float8	Average I/O service time (unit: ms).
avgqu-sz	int8	Number of current I/O waiting queues.
%util	float8	Disk usage: Percentage of time a device spends on I/O operations.

- gs_get_memstat(start_time timestamptz, end_time timestamptz)

  Description: Queries memory resource information within a period of time. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following describes the parameters.

Parameter	Description	Value Range
start_time	Required. Start time of the query.	Timestamptz type.
end_time	Required. End time of the query.	Timestamptz type.

  Return type: SETOF record

  The following describes the return columns.

Column	Type	Description
collection_time	timestamptz	Collection time.
mem_total	int8	Total memory size (unit: byte).
mem_used	int8	Size of the used memory (unit: byte).
mem_free	int8	Size of the idle memory (unit: byte).
mem_usage	float8	Memory usage.
buffer	int8	Size of the buffer memory (unit: byte).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
cache	int8	Size of the cache stack memory (unit: byte).
mem_available	int8	Size of the memory available for starting a new application (unit: byte).
active	int8	Size of actively cached memory (unit: byte).
inactive	int8	Size of inactively cached memory (unit: byte).
swap_total	int8	Total size of the swap space (unit: byte).
swap_free	int8	Size of the idle swap space (unit: byte).

- gs_get_netstat(start_time timestamptz, end_time timestamptz)

  Description: Queries network resource information within a period of time. To call this function, you must have the MONADMIN or SYSADMIN permission.

  The following describes the parameters.

Parameter	Description	Value Range
start_time	Required. Start time of the query.	Timestamptz type.
end_time	Required. End time of the query.	Timestamptz type.

  Return type: SETOF record

  The following describes the return columns.

Column	Type	Description
collection_time	timestamptz	Collection time.
interface	text	API name.
tag	text	Tag.
rkb/s	float8	Rx rate (kbit/s).
tkb/s	float8	Tx rate (kbit/s).
kb/s	float8	Total rate (kbit/s).
rp/s	float8	Rx packet rate (packets/s).
tp/s	float8	Tx packet rate (packets/s).
p/s	float8	Total data packet rate (packets/s).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
rerr/s	float8	Rx packet error rate (packets/s).
terr/s	float8	Tx packet error rate (packets/s).
%err	float8	Total error rate.

1.6.55 Sorting Calculation Functions

The sorting calculation functions provide basic calculation capabilities for sorting scenarios.

Table 1-299 Sorting calculation functions

Description	Functions
Sorts a given column by time sequence and returns the first value.	**first(value_array anyarray, time_array timearray)**
Sorts a given column by time sequence and returns the last value.	**last(value_array anyarray, time_array timearray)**
Sorts a given column by sequence and returns a percentile value.	**percentile_cont(percentile float)**
Sorts a given column in ascending order and returns a percentile value.	**value_of_percentile(column float, percentile float, compression float)**
Sorts a given column in ascending order and calculates the percentile based on the percentile value.	**percentile_of_value(column float, percentilevalue float, compression float)**
Sorts a given column and returns the minimum *K* tuples.	**top_k(array real[],k int)**
Sorts a given column and returns the maximum *K* tuples.	**bottom_k(array real[],k int)**

first(value_array anyarray, time_array timearray)

Description: Sorts a given column by time sequence and returns the first value.

Parameters: See **Table 1-300**.

Return type: anyelement

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-300 Parameters of the first function

Parameter	Type	Description
value_array	anyarray	Array of the queried column.
time_array	timearray	Array of time values.

NOTE

- The function uses the INTEGER type as an array index internally and does not support data volumes exceeding the value range supported by the INTEGER type.

Example:

```
gaussdb=# SELECT
gaussdb-#     first(array_agg(value), array_agg(time))
gaussdb-# FROM
gaussdb-#     (VALUES ('2019-07-12 00:00:00'::timestamptz,'c','a'),
gaussdb(#             ('2019-07-12 00:01:00'::timestamptz,'b','a'),
gaussdb(#             ('2019-07-12 00:02:01'::timestamptz,'a','b')) v(time, value, tag)
gaussdb-# GROUP BY tag;
 first
-------
 c
 a
(2 rows)
```

last(value_array anyarray, time_array timearray)

Description: Sorts a given column by time sequence and returns the last value.

Parameters: See **Table 1-301**.

Return type: anyelement

Table 1-301 Parameters of the last function

Parameter	Type	Description
value_array	anyarray	Array of the queried column.
time_array	timearray	Array of time values.

NOTE

- The function uses the INTEGER type as an array index internally and does not support data volumes exceeding the value range supported by the INTEGER type.

Example:

```
gaussdb=# SELECT
gaussdb-#     last(array_agg(value), array_agg(time))
gaussdb-# FROM
gaussdb-#     (VALUES ('2019-07-12 00:00:00'::timestamptz,'c','a'),
gaussdb(#             ('2019-07-12 00:01:00'::timestamptz,'b','a'),
gaussdb(#             ('2019-07-12 00:02:01'::timestamptz,'a','b')) v(time, value, tag)
gaussdb-# GROUP BY tag;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 last
------
 b
 a
(2 rows)
```

percentile_cont(percentile float)

Description: Sorts a given column by sequence and returns a percentile value. It must be used together with WITHIN GROUP (ORDER BY) to specify the column to be calculated. The column must be of the numeric or interval type.

Parameters: See **Table 1-302**.

Return type: float or interval, depending on whether the value of the column specified by WITHIN GROUP (ORDER BY) is of the numeric or interval type.

Table 1-302 Parameters of the percentile_cont function

Parameter	Type	Description
percentile	float	Specifies the percentile. The value range is [0,1]. The precision value is of the floating-point type. For example, 0.95 means the 95th percentile value.

NOTE

- **percentile** is a decimal in the range [0,1]. The value is of the floating-point type and does not support the percent sign (%), for example, 95%.

Example:

```
gaussdb=# SELECT
gaussdb-#      percentile_cont(0.5) WITHIN GROUP (ORDER BY value)
gaussdb-# FROM
gaussdb-#     (VALUES ('2019-07-12 00:00:01'::timestamptz,'red',1),
gaussdb(#     ('2019-07-12 00:00:01'::timestamptz,'blue',2)) v(time,color,value);
 percentile_cont
-----------------
             1.5
(1 row)
```

value_of_percentile(column float, percentile float, compression float)

Description: Sorts a given column in ascending order and returns a percentile value.

Parameters: See **Table 1-303**.

Return type: float

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-303 Parameters of the value_of_percentile function

Parameter	Type	Description
column	float	Specifies the column name.
percentile	float	Specifies the percentile. The value range is [0,1]. For example, **0.95** means the 95th percentile value.
compression	float	Specifies the compression coefficient. The value range is [0,500]. The default value is **300**. If you want to use the default value, set this parameter to **0**. Otherwise, set this parameter based on the value range. For example, if the value is **300**, the total data volume is compressed based on 300. The larger the data volume, the larger the memory usage.

NOTE

- You need to specify the column name.
- **percentile** is a decimal in the range [0,1]. The value is of the floating-point type and does not support the percent sign (%), for example, 95%.
- **compression** is a number in the range [0,500]. The default value is **300**. If the input value is greater than 500 or less than 0, the default value is used.
- You are advised to use this function in scenarios that meet the following conditions. Under the premise of satisfying these conditions, the accuracy exceeds 75% in most cases.

  (1) The overall data follows a normal distribution.

  (2) The data is evenly distributed on each DN, following a normal distribution as well.

  (3) The data volume must be greater than 10,000.

  (4) It is only suitable for extreme data, specifically data with percentiles less than 1% and greater than 99%.

Example:

```
gaussdb=#DROP TABLE IF EXISTS partition_scan2;
gaussdb=#CREATE TABLE partition_scan2(a int);
gaussdb=#insert into partition_scan2 values(generate_series(1,100,1));
gaussdb=# SELECT value_of_percentile(a,0.8,0) from partition_scan2;
 value_of_percentile
---------------------
                80.5
(1 row)
```

percentile_of_value(column float, percentilevalue float, compression float)

Description: Sorts a given column in ascending order and returns a percentile.

Parameters: See **Table 1-304**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: float

Table 1-304 Parameters of the percentile_of_value function

Parameter	Type	Description
column	float	Specifies the column name.
percentilevalue	float	Specifies the percentile value. The percentile is calculated based on the percentile value. The precision is a floating-point value.
compression	float	Specifies the compression coefficient. The value range is [0,500]. The precision is a floating-point value. The default value is **300**. If the default value is used, set this parameter to **0**. Otherwise, set this parameter based on the value range. For example, if the value is **300**, the total data volume is compressed based on 300. The larger the data volume, the larger the memory usage.

NOTE

- You need to specify the column name.
- **percentilevalue**: any value. The precision is a floating-point value.
- **compression** is a number in the range [0,500]. The default value is **300**. If the input value is greater than 500 or less than 0, the default value is used.
- You are advised to use this function in scenarios that meet the following conditions. Under the premise of satisfying these conditions, the accuracy exceeds 75% in most cases.

  (1) The overall data follows a normal distribution.

  (2) The data is evenly distributed on each DN, following a normal distribution as well.

  (3) The data volume must be greater than 10,000.

  (4) It is only suitable for extreme data, specifically data with percentiles less than 1% and greater than 99%.

Example:

```
gaussdb=#DROP TABLE IF EXISTS partition_scan2;
gaussdb=#CREATE TABLE partition_scan2(a int);
gaussdb=#insert into partition_scan2 values(generate_series(1,100,1));
gaussdb=# SELECT percentile_of_value(a,80,0) from partition_scan2;
 percentile_of_value
---------------------
               .795
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

top_k(array real[],k int)

Description: Sorts a given column in ascending order and returns the minimum *K* tuples.

Parameters: See **Table 1-305**.

Return type: real[]

Table 1-305 Parameters of the top_k function

Parameter	Type	Description
array	real[]	Array to be input.
k	int	Specifies the minimum *K* tuples.

NOTE

- *K* should be a non-negative number. If it is negative, a null value is returned.
- *K* should be less than the array size. If it is greater than the current array length, an error is reported.
- array_sort is used for internal sorting, supporting only one-dimensional arrays. For details, see **Array Functions and Operators**.
- The **array** parameter cannot be empty. If it is empty, the function returns null directly.
- Arrays with data volumes exceeding the value range of the int type cannot be used.

Example:

```
gaussdb=# SELECT
gaussdb-#       top_k(array_agg(v), 0.5),v2
gaussdb-# FROM
gaussdb-#       (VALUES (1,'a'),(2,'a'),(3,'a'),(4,'a'),(5,'a'),(6,'a')) v(v,v2)
gaussdb-# GROUP BY v2;
 top_k | v2
-------+----
 {1}   | a
(1 row)
```

bottom_k(array real[],k int)

Description: Sorts a given column in ascending order and returns the maximum *K* tuples.

Parameters: See **Table 1-306**.

Return type: real[]

Table 1-306 Parameters of the bottom_k function

Parameter	Type	Description
k	int	Specifies the maximum *K* tuples.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- *K* should be a non-negative number. If it is negative, a null value is returned.
- *K* should be less than the array size. If it is greater than the current array length, an error is reported.
- array_sort is used for internal sorting, supporting only one-dimensional arrays. For details, see **Array Functions and Operators**.
- The **array** parameter cannot be empty. If it is empty, the function returns null directly.
- Arrays with data volumes exceeding the value range of the int type cannot be used.

Example:

```
gaussdb=# SELECT
gaussdb-#       bottom_k(array_agg(v), 0.5),v2
gaussdb-# FROM
gaussdb-#       (VALUES (1,'a'),(2,'a'),(3,'a'),(4,'a'),(5,'a'),(6,'a')) v(v,v2)
gaussdb-# GROUP BY v2;
 bottom_k | v2
----------+----
 {6}      | a
(1 row)
```

1.6.56 Autonomous Transaction Feature Functions

- gs_autonomous_transaction_detail()

  Description: Displays the autonomous transaction sessions to which the user belongs and the relationships of parent transactions and sub-transactions on the current CN. Users with the MONADMIN or SYSADMIN permission can view all autonomous transaction sessions and the relationships of parent transactions and sub-transactions on the current CN. Calling on DNs is not allowed.

  Parameter: none.

  Return type: SETOF record

  Return values: Refer to **Table 1-307**.

  Table 1-307 Return values of the gs_autonomous_transaction_detail function

Name	Type	Description
autonomous_transaction_session_id	bigint	Session ID of an autonomous transaction.
autonomous_transaction_thread_id	bigint	Thread ID of an autonomous transaction.
parent_autonomous_transaction_session_id	bigint	Parent session ID of an autonomous transaction.

  Example:

```
gaussdb=#
CREATE OR REPLACE PROCEDURE proc_test()  AS
DECLARE
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
pg_sleep(60);
END;
/
gaussdb=# CALL proc_test();

-- A new session needs to be created for query. --
gaussdb=# SELECT * FROM pg_catalog.gs_autonomous_transaction_detail() AS
gaussdb-#     (autonomous_transaction_session_id bigint, autonomous_transaction_thread_id bigint,
parent_autonomous_transaction_session_id bigint);
 autonomous_transaction_session_id | autonomous_transaction_thread_id |
parent_autonomous_transaction_session_id
-----------------------------------+-----------------------------------+-----------------------------------------
                  139778155607808 |                  139778155607808 |                      139777830549248
(1 row)
```

- gs_get_autonomous_transaction_count()

  Description: Queries the autonomous transaction count on the current CN. Calling on DNs is not allowed.

  Parameter: none.

  Return type: integer

  Return values: See **Table 1-308**.

  Table 1-308 Return values of the gs_get_autonomous_transaction_count function

Name	Type	Description
gs_get_autonomous_transaction_count	integer	Current autonomous transaction count.

  Example:

  ```
  gaussdb=#
  CREATE OR REPLACE PROCEDURE proc_test()  AS
  DECLARE
  PRAGMA AUTONOMOUS_TRANSACTION;
  BEGIN
  pg_sleep(60);
  END;
  /
  gaussdb=# CALL proc_test();

  -- A new session needs to be created for query. --
  gaussdb=# SELECT gs_get_autonomous_transaction_count();
   gs_get_autonomous_transaction_count
  -------------------------------------
                                     1
  (1 row)
  ```

- gs_reset_autonomous_transaction_count()

  Description: Restores the autonomous transaction count on the current CN to the correct value and returns the correct autonomous transaction count. To call this function, you must have the MONADMIN or SYSADMIN permission. Calling on DNs is not allowed.

  Parameter: none.

  Return type: integer

  Return values: See **Table 1-309**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-309 Return values of the gs_reset_autonomous_transaction_count function

Name	Type	Description
gs_reset_autonomous_transaction_count	integer	Correct autonomous transaction count after restoration.

Example:

```
gaussdb=# SELECT gs_reset_autonomous_transaction_count();
 gs_reset_autonomous_transaction_count
---------------------------------------
                                     0
(1 row)

gaussdb=#
CREATE OR REPLACE PROCEDURE proc_test()  AS
DECLARE
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
pg_sleep(60);
END;
/
gaussdb=# CALL proc_test();

-- A new session needs to be created for query. --
gaussdb=# SELECT gs_reset_autonomous_transaction_count();
 gs_reset_autonomous_transaction_count
---------------------------------------
                                     1
(1 row)
```

- gs_global_autonomous_transaction_detail()

  Description: Displays the autonomous transaction sessions to which the user belongs and the relationships of parent transactions and sub-transactions on each CN. Users with the MONADMIN or SYSADMIN permission can view all autonomous transaction sessions and the relationships of parent transactions and sub-transactions on each CN. Calling on DNs is not allowed.

  Parameter: none.

  Return type: SETOF record

  Return values: Refer to **Table 1-310**.

  NOTE

  When the GUC parameter **enable_nonsysadmin_execute_direct** is set to **off**, this function can be called only by users with the MONADMIN or SYSADMIN permission.

Table 1-310 Return values of the gs_global_autonomous_transaction_detail function

Name	Type	Description
node_name	text	CN name.
autonomous_transaction_session_id	bigint	Session ID of an autonomous transaction.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
autonomous_transaction_thread_id	bigint	Thread ID of an autonomous transaction.
parent_autonomous_transaction_session_id	bigint	Parent session ID of an autonomous transaction.

Example:

```
gaussdb=#
CREATE OR REPLACE PROCEDURE proc_test()  AS
DECLARE
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
pg_sleep(60);
END;
/
gaussdb=# CALL proc_test();

-- A new session needs to be created for query. --
gaussdb=# SELECT * FROM gs_global_autonomous_transaction_detail();
  node_name   | autonomous_transaction_session_id | autonomous_transaction_thread_id |
parent_autonomous_transaction_session_id
--------------+-----------------------------------+----------------------------------
+-----------------------------------------
 coordinator1 |                140477506516736 |                140477506516736 |
140477683726080
(1 row)
```

- gs_global_get_autonomous_transaction_count()

  Description: Queries the current autonomous transaction count on each CN. Calling on DNs is not allowed.

  Parameter: none.

  Return type: SETOF record

  Return values: Refer to **Table 1-311**.

  NOTE

  When the GUC parameter **enable_nonsysadmin_execute_direct** is set to **off**, this function can be called only by users with the MONADMIN or SYSADMIN permission.

Table 1-311 Return values of the gs_global_get_autonomous_transaction_count function

Name	Type	Description
node_name	text	CN name.
gs_get_autonomous_transaction_count	integer	Current autonomous transaction count.

Example:

```
gaussdb=#
CREATE OR REPLACE PROCEDURE proc_test()  AS
```

```
DECLARE
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
pg_sleep(60);
END;
/
gaussdb=# CALL proc_test();

-- A new session needs to be created for query. --
gaussdb=# SELECT * FROM gs_global_get_autonomous_transaction_count();
  node_name   | gs_get_autonomous_transaction_count
--------------+-------------------------------------
 coordinator1 |                                   1
(1 row)
```

- gs_global_reset_autonomous_transaction_count([exec_node_name text])

  Description: Restores the autonomous transaction count on each CN to the correct value and returns the correct autonomous transaction count on each CN after the restoration. If the **exec_node_name** parameter is specified, the autonomous transaction count on the specified node is restored to the correct value. To call this function, you must have the MONADMIN or SYSADMIN permission. Calling on DNs is not allowed.

  Parameter: **exec_node_name**, specifying the name of a CN. This parameter is optional.

  Return type: SETOF record

  Return values: Refer to **Table 1-312**.

  Table 1-312 Return values of the gs_global_reset_autonomous_transaction_count function

Name	Type	Description
node_name	text	CN name.
gs_reset_autonomous_transaction_count	integer	Correct autonomous transaction count after restoration.

  Example:

```
gaussdb=#
CREATE OR REPLACE PROCEDURE proc_test()  AS
DECLARE
PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
pg_sleep(60);
END;
/
gaussdb=# CALL proc_test();

-- A new session needs to be created for query. --
gaussdb=# SELECT * FROM gs_global_reset_autonomous_transaction_count();
  node_name   | gs_reset_autonomous_transaction_count
--------------+---------------------------------------
 coordinator1 |                                     1
(1 row)

gaussdb=# SELECT * FROM gs_global_reset_autonomous_transaction_count('coordinator1');
  node_name   | gs_reset_autonomous_transaction_count
--------------+---------------------------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 coordinator1 |                      1
(1 row)
```

1.6.57 DSL Rule Management Functions

- dsl_add_rule()

  Description: Adds a DSL rule. Only the initial user can call this function.

  Parameters: See **Table 1-313**.

  Return type: text.

  Return value: DSL rule name.

  Table 1-313 Parameters of the dsl_add_rule function

Input Parameter	Type	Required	Description
rulename	name	Y	Name of a DSL rule. The name must start with a letter and can have up to 64 bytes, containing a mix of letters, digits, underscores (_), and hyphens (-). Chinese characters and other special characters are not allowed.
ruletext	text	Y	Text content of the DSL rule.
priority	integer	Y	Priority of the DSL rule. The priority must be globally unique and fall within the range of 1 to 9999. A larger value indicates a lower priority.
enabled	boolean	N	Specifies whether to enable the rule. The default value is **FALSE**.
ruledesc	name	N	Description of the DSL rule. The description can be left blank or contain a maximum of 64 bytes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Input Parameter	Type	Required	Description
ruledbcompatibility	text	N	Compatibility mode of the database that houses the DSL rule. Values: **ORA**, **MYSQL**, **TD**, **PG**, **M**, and **ANY**. They represent O-, MY-, TD-, PG-, M-, and all-compatible modes, respectively. You can set multiple compatibility modes, separated by vertical bars (|). For example, **A|PG** indicates compatibility with both Oracle Database and PG. Default value: **ANY**.
planmode	text	N	Plan type supported by the rule. The options are **Stream**, **RemoteQuery**, and **ANY**, with **ANY** being the default value.

Example:

```
gaussdb=# SELECT dsl_add_rule(
'compute_const',
'Filter<> {args: $expr: *}
=>
Filter<> {args: (expr_compute $expr)
}',
3,
true);
 dsl_add_rule
---------------
 compute_const
(1 row)
```

- dsl_del_rule()

  Description: Deletes a DSL rule. Only the initial user and system administrators can call this function.

  Parameters: See **Table 1-314**.

  Return type: integer.

  Return value: number of DSL rules deleted.

  Table 1-314 Parameters of the dsl_del_rule function

Input Parameter	Type	Description
rulename	name	Name of a DSL rule.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# SELECT dsl_del_rule('compute_const');
 dsl_del_rule
--------------
            1
(1 row)
```

- dsl_set_rule()

  Description: Modifies a DSL rule (or rather, a specific attribute of the rule). The content of the rule cannot be modified. Only the initial user and system administrators can call this function.

  Parameters: See **Table 1-315**.

  Return type: integer.

  Return value: number of DSL rules modified.

  Table 1-315 Parameters of the dsl_set_rule function

Input Parameter	Type	Description
rulename	name	Name of a DSL rule.
attrname	text	Attribute name. The supported attributes are **enabled**, **priority**, **ruledesc**, **dbcompatibility**, and **planmode**.
attrval	text	Attribute value.

  Example:

```
gaussdb=# SELECT dsl_set_rule('compute_const', 'priority', '8');
 dsl_set_rule
--------------
            1
(1 row)

gaussdb=# \x
Expanded display is on.

gaussdb=# SELECT * FROM GS_DSL_REWRITERULE WHERE rulename = 'compute_const';
-[ RECORD 1 ]---+-----------------------------------
enabled         | t
externalrule    | t
costbased       | f
priority        | 8  -- The priority of the rule is set to 8.
dbcompatibility | 31
planmode        | 3
createtime      | xxxx-xx-xx 16:43:18.766029+08
lastupdatetime  | Parameters of the xxdsl_reload_rule function xx-xx-xx 16:44:55.785637+08
rulename        | compute_const
rulehash        | 8576886722021209825
ruletext        | Filter<> {args: $expr: *}
                | =>
                | Filter<> {args: (expr_compute $expr)
                | }
ruledesc        |
```

- dsl_reload_rule()

  Description: Loads a DSL rule to the cache.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: See **Table 1-316**.

Return type: text.

Return value: **success** or **failed**.

Table 1-316 Parameters of the dsl_reload_rule function

Input Parameter	Type	Description
nodename	text	Node name. An empty value indicates that the rule will be executed on all nodes.

Example:

```
gaussdb=# SELECT dsl_reload_rule();
 dsl_reload_rule
-----------------
 success
(1 row)
```

1.6.58 ROWID Functions

- rowid_tableoid(rowid)

  Description: Obtains the **table_oid** value of the ROWID type.

  Parameter: rowid

  Return type: OID

  Examples:

  ```
  gaussdb=# CREATE DATABASE ora_test DBCOMPATIBILITY='ORA';
  CREATE DATABASE
  gaussdb=# \c ora_test
  Non-SSL connection (SSL connection is recommended when requiring high-security)
  You are now connected to database "ora" as user "$user".
  ora_test=# SELECT rowid_tableoid('AAAAAAAAAP//BBBBBAAAAAAAAAAAC');
   rowid_tableoid
  ----------------
      1090785344
  (1 row)
  ora_test=# \c postgres
  Non-SSL connection (SSL connection is recommended when requiring high-security)
  You are now connected to database "postgres" as user "$user".
  gaussdb=# DROP DATABASE ora_test;
  DROP DATABASE
  ```

- rowid_sequence(rowid)

  Description: Obtains the **row_no** value of the ROWID type.

  Parameter: rowid

  Return type: int8

  Examples:

  ```
  gaussdb=# CREATE DATABASE ora_test DBCOMPATIBILITY='ORA';
  CREATE DATABASE
  gaussdb=# \c ora_test
  Non-SSL connection (SSL connection is recommended when requiring high-security)
  You are now connected to database "ora" as user "$user".
  ora_test=# SELECT rowid_sequence('AAAAAAAAAP//BBBBBAAAAAAAAAAAC');
   rowid_sequence
  ----------------
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
           2
(1 row)
ora_test=# \c postgres
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "postgres" as user "$user".
gaussdb=# DROP DATABASE ora_test;
DROP DATABASE
```

- rowid_nodeid(rowid)

  Description: Obtains the **node_slice** value of the ROWID type.

  Parameter: rowid

  Return type: int2

  Examples:

```
gaussdb=# CREATE DATABASE ora_test DBCOMPATIBILITY='ORA';
CREATE DATABASE
gaussdb=# \c ora_test
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "ora" as user "$user".
ora_test=# SELECT rowid_nodeid('AAAAAAAAAP//BBBBBAAAAAAAAAAAAC');
 rowid_nodeid
--------------
           0
(1 row)
ora_test=# \c postgres
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "postgres" as user "$user".
gaussdb=# DROP DATABASE ora_test;
DROP DATABASE
```

- rowid_hashbucketid(rowid)

  Description: Obtains the **bucket_id** value of the ROWID type.

  Parameter: rowid

  Return type: int2

  Examples:

```
gaussdb=# CREATE DATABASE ora_test DBCOMPATIBILITY='ORA';
CREATE DATABASE
gaussdb=# \c ora_test
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "ora" as user "$user".
ora_test=# SELECT rowid_hashbucketid('AAAAAAAAAP//BBBBBAAAAAAAAAAAAC');
 rowid_hashbucketid
--------------------
                 -1
(1 row)
ora_test=# \c postgres
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "postgres" as user "$user".
gaussdb=# DROP DATABASE ora_test;
DROP DATABASE
```

1.6.59 Functions for Recompilation upon Invalidation

- transform_view_dep_source()

  Description: Updates the **source** column in the **GS_DEP_SOURCE** system catalog. Specifically, it extends the asterisk (*) in "SELECT *" within the view definition statement, appends the schema used by objects at the time of view creation, and updates view dependencies. This function can be only called by administrators.

  Parameter: none.

  Return type: void.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_compile_schema(schema_name in varchar2 default null, compile_all in boolean default false, retry_times in int default 10)

  Description: Recompiles packages, functions, stored procedures, and views in the specified namespace.

  Parameters: Refer to **Table 1-317**.

  Return type: Boolean, specifically **true** (successful) and **false** (failed).

  Table 1-317 gs_compile_schema parameters

Parameter	Type	IN/OUT	Can Be Empty	Description
schema_name	VARCHAR2	IN	Yes	Specifies the namespace name.
compile_all	Boolean	IN	Yes	Specifies whether to recompile all objects, including the views that are currently valid. • **false**: The invalid views in the PG_OBJECT table is compiled. • **true**: All views in the PG_OBJECT table are compiled.
retry_times	INT	IN	Yes	Specifies the number of retry times when recompilation fails.

  Example:

```
-- Enable recompilation upon invalidation.
gaussdb=# SET enable_view_invalidation='on';

-- Create a schema, a table, a view that depends on the table, and another view that depends on this view.
gaussdb=# DROP SCHEMA IF EXISTS tesch2 CASCADE;
gaussdb=# CREATE SCHEMA tesch2;
gaussdb=# SET current_schema = tesch2;
gaussdb=# CREATE TABLE tab1 (c1 INT, c2 INT);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'c1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# CREATE VIEW v_tab1 AS SELECT * FROM tab1;
gaussdb=# CREATE VIEW v_tab2 AS SELECT * FROM v_tab1;
gaussdb=# CREATE VIEW v_tab3 AS SELECT * FROM tab1;

-- Drop the table.
gaussdb=# DROP TABLE tab1;

-- Check the object's validity.
gaussdb=# SELECT c.relname, o.valid FROM pg_catalog.pg_object o JOIN pg_catalog.pg_class c ON c.oid = o.object_oid WHERE c.relname = 'v_tab1' or c.relname = 'v_tab2' or c.relname = 'v_tab3';
 relname | valid
---------+-------
 v_tab1  | f
 v_tab2  | f
 v_tab3  | f
(3 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Restore the dropped table.
gaussdb=# CREATE TABLE tab1 (c1 INT, c2 INT);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'c1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE

-- Recompile view objects in the tesch2 namespace.
gaussdb=# CALL pg_catalog.gs_compile_schema('tesch2', true, 1);
INFO:  successful
CONTEXT:  referenced column: gs_compile_schema
SQL statement "select pkg_util.gs_compile_schema(:schema_name, :compile_all, :retry_times);"
PL/pgSQL function gs_compile_schema(character varying,boolean,integer) line 24 at EXECUTE
statement
INFO:  The number of executions is 1
INFO:  successful
 gs_compile_schema
-------------------
 t
(1 row)

-- Check the view's validity again.
gaussdb=# SELECT c.relname, o.valid FROM pg_catalog.pg_object o JOIN pg_catalog.pg_class c ON
c.oid = o.object_oid WHERE c.relname = 'v_tab1' or c.relname = 'v_tab2' or c.relname = 'v_tab3';
 relname | valid
---------+-------
 v_tab1  | t
 v_tab2  | t
 v_tab3  | t
(3 rows)
-- Drop views.
gaussdb=# DROP VIEW v_tab3;
gaussdb=# DROP VIEW v_tab2;
gaussdb=# DROP VIEW v_tab1;
-- Drop the table.
gaussdb=# DROP TABLE tab1;
gaussdb=# RESET current_schema;
-- Drop the schema.
gaussdb=# DROP SCHEMA tesch2 CASCADE;
```

1.6.60 Other System Functions

Other system functions are classified into PG-compatible functions and internal functions. These functions are not recommended. If you need to use them, contact Huawei technical support.

PG-compatible Functions and Operators

- The following table lists the built-in functions and operators of GaussDB that are compatible with PostgreSQL. For details about the functions, see the PostgreSQL official documentation.

_pg_char_max_length	_pg_char_octet_length	_pg_datetime_precision	_pg_expandarray	_pg_index_position	_pg_interval_type	_pg_numeric_precision
_pg_numeric_precision_radix	_pg_numeric_scale	_pg_truetypid	_pg_truetypmod	q	abs	abstime
abstimeeq	abstimege	abstimegt	abstimein	abstimele	abstimelt	abstimene

abstimeout	abstimerecv	abstimesend	aclcontains	acldefault	aclexplode	aclinsert
aclitemeq	aclitemin	aclitemout	aclremove	acos	age	akeys
any_in	any_out	anyarray_in	anyarray_out	anyarray_recv	anyarray_send	anyelement_in
anyelement_out	anyenum_in	anyenum_out	anynonarray_in	anynonarray_out	anyrange_in	anyrange_out
anytextcat	area	areajoinsel	areasel	array_agg	array_agg_finalfn	array_agg_transfn
array_append	array_cat	array_dims	array_eq	array_fill	array_ge	array_gt
array_in	array_larger	array_le	array_length	array_lower	array_lt	array_ndims
array_ne	array_out	array_prepend	array_recv	array_send	array_smaller	array_to_json
array_to_string	array_typanalyze	array_upper	arraycontained	arraycontains	arraycontjoinsel	arraycontsel
arrayoverlap	ascii	asin	atan	atan2	avals	avg
big5_to_euc_tw	big5_to_mic	big5_to_utf8	bit	bit_and	bit_in	bit_length
bit_or	bit_out	bit_recv	bit_send	bitand	bitcat	bitcmp
biteq	bitge	bitgt	bitle	bitlt	bitne	bitnot
bitor	bitshiftleft	bitshiftright	bittypmodin	bittypmodout	bitxor	bool
bool_and	bool_or	booland_statefunc	booleq	boolge	boolgt	boolin
boolle	boollt	boolne	boolor_statefunc	boolout	boolrecv	boolsend
box	box_above	box_above_eq	box_add	box_below	box_below_eq	box_center

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

box_contain	box_contain_pt	box_contained	box_distance	box_div	box_eq	box_ge
box_gt	box_in	box_intersect	box_le	box_left	box_lt	box_mul
box_out	box_overabove	box_overbelow	box_overlap	box_overleft	box_overright	box_recv
box_right	box_same	box_send	box_sub	bpchar	bpchar_larger	bpchar_pattern_ge
bpchar_pattern_gt	bpchar_pattern_le	bpchar_pattern_lt	bpchar_smaller	bpchar_sortsupport	bpcharcmp	bpchareq
bpcharge	bpchargt	bpcharic like	bpcharicnlike	bpcharicregexeq	bpcharicregexne	bpcharin
bpcharle	bpcharlike	bpcharlt	bpcharne	bpcharnlike	bpcharout	bpcharrecv
bpcharregexeq	bpcharregexne	bpcharsend	bpchartypmodin	bpchartypmodout	broadcast	btabstimecmp
btarraycmp	btbeginscan	btboolcmp	btbpchar_pattern_cmp	btbuild	btbuildempty	btbulkdelete
btcanreturn	btcharcmp	btcostestimate	btendscan	btfloat48cmp	btfloat4cmp	btfloat4sortsupport
btfloat84cmp	btfloat8cmp	btfloat8sortsupport	btgetbitmap	btgettuple	btinsert	btint24cmp
btint28cmp	btint2cmp	btint2sortsupport	btint42cmp	btint48cmp	btint4cmp	btint4sortsupport
btint82cmp	btint84cmp	btint8cmp	btint8sortsupport	btmarkpos	btnamecmp	btnamesortsupport
btoidcmp	btoidsortsupport	btoidvectorcmp	btoptions	btrecordcmp	btreltimecmp	btrescan
btrestrpos	btrim	bttext_pattern_cmp	bttextcmp	bttextsortsupport	bttidcmp	bttintervalcmp

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

btvacuumcleanup	bytea_sortsupport	bytea_string_agg_finalfn	bytea_string_agg_transfn	byteacat	byteacmp	byteaeq
byteage	byteagt	byteain	byteale	bytealike	bytealt	byteane
byteanlike	byteaout	byterecv	byteasend	cash_cmp	cash_div_cash	cash_div_flt4
cash_div_flt8	cash_div_int2	cash_div_int4	cash_div_int8	cash_eq	cash_ge	cash_gt
cash_in	cash_le	cash_lt	cash_mi	cash_mul_flt4	cash_mul_flt8	cash_mul_int2
cash_mul_int4	cash_mul_int8	cash_ne	cash_out	cash_pl	cash_recv	cash_send
cashlarger	cashsmaller	cbrt	ceil	ceiling	center	char
char_length	character_length	chareq	charge	chargt	charin	charle
charlt	charne	charout	charrecv	charsend	chr	cideq
cidin	cidout	cidr	cidr_in	cidr_out	cidr_recv	cidr_send
cidrecv	cidsend	circle	circle_above	circle_add_pt	circle_below	circle_center
circle_contain	circle_contain_pt	circle_contained	circle_distance	circle_div_pt	circle_eq	circle_ge
circle_gt	circle_in	circle_le	circle_left	circle_lt	circle_mul_pt	circle_ne
circle_out	circle_overabove	circle_overbelow	circle_overlap	circle_overleft	circle_overright	circle_recv
circle_right	circle_same	circle_send	circle_sub_pt	clock_timestamp	close_lb	close_ls
close_lseg	close_pb	close_pl	close_ps	close_sb	close_sl	col_description

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

concat	concat_ws	contjoinsel	contsel	convert	convert_from	convert_to
corr	cos	cot	count	covar_pop	covar_samp	cstring_in
cstring_out	cstring_recv	cstring_send	cume_dist	current_database	current_query	current_schema
xpath_exists	current_setting	current_user	currtid	currtid2	currval	cursor_to_xml
cursor_to_xmlschema	database_to_xml	database_to_xml_and_xmlschema	database_to_xmlschema	date	date_cmp	date_cmp_timestamp
date_cmp_timestamptz	date_eq	date_eq_timestamp	date_eq_timestamptz	date_ge	date_ge_timestamp	date_ge_timestamptz
date_gt	date_gt_timestamp	date_gt_timestamptz	date_in	date_larger	date_le	date_le_timestamp
date_le_timestamptz	date_lt	date_lt_timestamp	date_lt_timestamptz	date_mi	date_mi_interval	date_mii
date_ne	date_ne_timestamp	date_ne_timestamptz	date_out	date_pl_interval	date_pli	date_recv
date_send	date_smaller	date_sortsupport	daterange_canonical	daterange_subdiff	datetime_pl	datetimetz_pl
dcbrt	decode	defined	degrees	delete	dense_rank	dexp
diagonal	diameter	dispell_init	dispell_lexize	dist_cpoly	dist_lb	dist_pb
dist_pc	dist_pl	dist_ppath	dist_ps	dist_sb	dist_sl	div
dlog1	dlog10	domain_in	domain_recv	dpow	dround	dsimple_init

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

dsimple_lexize	dsnowball_init	dsnowball_lexize	dsqrt	dsynonym_init	dsynonym_lexize	dtrunc
each	enum_ne	enum_out	enum_range	enum_recv	enum_send	enum_smaller
eqjoinsel	eqsel	euc_cn_to_mic	euc_cn_to_utf8	euc_jis_2004_to_shift_jis_2004	euc_jis_2004_to_utf8	euc_jp_to_mic
euc_jp_to_sjis	euc_jp_to_utf8	euc_kr_to_mic	euc_kr_to_utf8	euc_tw_to_big5	euc_tw_to_mic	euc_tw_to_utf8
every	exist	exists_all	exists_any	exp	factorial	family
fdw_handler_in	fdw_handler_out	fetchval	first_value	float4	float4_accum	float48div
float48eq	float48ge	float48gt	float48le	float48lt	float48mi	float48mul
float48ne	float48pl	float4abs	float4div	float4eq	float4ge	float4gt
float4in	float4larger	float4le	float4lt	float4mi	float4mul	float4ne
float4out	float4pl	float4recv	float4send	float4smaller	float4um	float4up
float8	float8_accum	float8_avg	float8_collect	float8_corr	float8_covar_pop	float8_covar_samp
float8_regr_accum	float8_regr_avgx	float8_regr_avgy	float8_regr_collect	float8_regr_intercept	float8_regr_r2	float8_regr_slope
float8_regr_sxx	float8_regr_sxy	float8_regr_syy	float8_stddev_pop	float8_stddev_samp	float8_var_pop	float8_var_samp
float84div	float84eq	float84ge	float84gt	float84le	float84lt	float84mi
float84mul	float84ne	float84pl	float8abs	float8div	float8eq	float8ge
float8gt	float8in	float8larger	float8le	float8lt	float8mi	float8mul

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

float8ne	float8out	float8pl	float8recv	float8send	float8smaller	float8um
float8up	floor	flt4_mul_cash	flt8_mul_cash	fmgr_c_validator	fmgr_internal_validator	fmgr_sql_validator
format	format_type	gb18030_to_utf8	gbk_to_utf8	generate_series	generate_subscripts	get_bit
get_byte	get_current_ts_config	get_global_gs_asp	get_large_table_name	-	-	-
gtsquery_compress	gtsquery_consistent	gtsquery_decompress	gtsquery_penalty	gtsquery_picksplit	gtsquery_same	gtsquery_union
gtsvector_compress	gtsvector_consistent	gtsvector_decompress	gtsvector_penalty	gtsvector_picksplit	gtsvector_same	gtsvector_union
gtsvectorin	gtsvectorout	has_tablespace_privilege	has_type_privilege	hash_aclitem	hashbeginscan	hashbuild
hashbuildempty	hashbulkdelete	hashcostestimate	hashendscan	hashgetbitmap	hashgettuple	hashinsert
hashint2vector	hashint4	hashint8	hashmacaddr	hashmarkpos	hashname	hashoid
hashoidvector	hashoptions	hashrescan	hashrestrpos	hashtext	hashvacuumcleanup	hashvarlena
host	hostmask	iclikejoinsel	iclikesel	icnlikejoinsel	icnlikesel	icregexeqjoinsel
icregexeqsel	icregexnejoinsel	icregexnesel	inet_client_addr	inet_client_port	inet_in	inet_out
inet_recv	inet_send	inet_server_addr	inet_server_port	inetand	inetmi	inetmi_int8

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

inetnot	inetor	inetpl	initcap	int2_accum	int2_avg_accum	int2_mul_cash
int2_sum	int24div	int24eq	int24ge	int24gt	int24le	int24lt
int24mi	int24mul	int24ne	int24pl	int28div	int28eq	int28ge
int28gt	int28le	int28lt	int28mi	int28mul	int28ne	int28pl
int2abs	int2and	int2div	int2eq	int2ge	int2gt	int2in
int2larger	int2le	int2lt	int2mi	int2mod	int2mul	int2ne
int2not	int2or	int2out	int2pl	int2recv	int2send	int2shl
int2shr	int2smaller	int2um	int2up	int2vectoreq	int2vectorin	int2vectorout
int2vectorrecv	int2vectorsend	int2xor	int4_accum	int4_avg_accum	int4_mul_cash	int4_sum
int42div	int42eq	int42ge	int42gt	int42le	int42lt	int42mi
int42mul	int42ne	int42pl	int48div	int48eq	int48ge	int48gt
int48le	int48lt	int48mi	int48mul	int48ne	int48pl	int4abs
int4and	int4div	int4eq	int4ge	int4gt	int4in	int4inc
int4larger	int4le	int4lt	int4mi	int4mod	int4mul	int4ne
int4not	int4or	int4out	int4pl	int4range	int4range_canonical	int4range_subdiff
int4recv	int4send	int4shl	int4shr	int4smaller	int4um	int4up
int4xor	int8	int8_avg	int8_avg_accum	int8_avg_collect	int8_mul_cash	int8_sum
int8_sum_to_int8	int8+1635:1668_accum	int82div	int82eq	int82ge	int82gt	int82le

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

int82lt	int82mi	int82mul	int82ne	int82pl	int84div	int84eq
int84ge	int84gt	int84le	int84lt	int84mi	int84mul	int84ne
int84pl	int8abs	int8and	int8div	int8eq	int8ge	int8gt
int8in	int8inc	int8inc_any	int8inc_float8_float8	int8larger	int8le	int8lt
int8mi	int8mod	int8mul	int8ne	int8not	int8or	int8out
int8pl	int8pl_inet	int8range	int8range_canonical	int8range_subdiff	int8recv	int8send
int8shl	int8shr	int8smaller	int8um	int8up	int8xor	integer_pl_date
inter_lb	inter_sb	inter_sl	internal_in	internal_out	interval	interval_accum
interval_avg	interval_cmp	interval_collect	interval_div	interval_eq	interval_ge	interval_gt
interval_hash	interval_in	interval_larger	interval_le	interval_lt	interval_mi	interval_mul
interval_ne	interval_out	interval_pl	interval_pl_date	interval_pl_time	interval_pl_timestamp	interval_pl_timestamptz
interval_pl_timetz	interval_recv	interval_send	interval_smaller	interval_transform	interval_um	intervaltypmodin
intervaltypmodout	intinterval	isexists	ishorizontal	iso_to_koi8r	iso_to_mic	iso_to_win1251
iso_to_win866	iso8859_1_to_utf8	iso8859_to_utf8	isparallel	isperp	isvertical	johab_to_utf8
jsonb_in	jsonb_out	jsonb_recv	jsonb_send	-	-	-
json_in	json_out	json_recv	json_send	justify_days	justify_hours	justify_interval
koi8r_to_iso	koi8r_to_mic	koi8r_to_utf8	koi8r_to_win1251	koi8r_to_win866	koi8u_to_utf8	language_handler_in

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

language_handler_out	latin1_to_mic	latin2_to_mic	latin2_to_win1250	latin3_to_mic	latin4_to_mic	like_escape
likejoinsel	likesel	line	line_distance	line_eq	line_horizontal	line_in
line_interpt	line_intersect	line_out	line_parallel	line_perp	line_recv	line_send
line_vertical	ln	lo_close	lo_creat	lo_create	lo_export	lo_import
lo_lseek	lo_open	lo_tell	lo_truncate	lo_unlink	log	loread
lower	lower_inc	lower_inf	lowrite	lpad	lseg	lseg_center
lseg_distance	lseg_eq	lseg_ge	lseg_gt	lseg_horizontal	lseg_in	lseg_interpt
lseg_intersect	lseg_le	lseg_length	lseg_lt	lseg_ne	lseg_out	lseg_parallel
lseg_perp	lseg_recv	lseg_send	lseg_vertical	ltrim	macaddr_and	macaddr_cmp
macaddr_eq	macaddr_ge	macaddr_gt	macaddr_in	macaddr_le	macaddr_lt	macaddr_ne
macaddr_not	macaddr_or	macaddr_out	macaddr_recv	macaddr_send	makeaclitem	masklen

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

max	md5 The MD5 algorithm has lower security and poses security risks. Therefore, you are advised to use a more secure algorithm.	mic_to_big5	mic_to_euc_cn	mic_to_euc_jp	mic_to_euc_kr	mic_to_euc_tw
mic_to_iso	mic_to_koi8r	mic_to_latin1	mic_to_latin2	mic_to_latin3	mic_to_latin4	mic_to_sjis
mic_to_win1250	mic_to_win1251	mic_to_win866	min	mktinterval	money	mul_d_interval
name	nameeq	namege	namegt	nameiclike	nameicnlike	nameicregexeq
nameicregexne	namein	namele	namelike	namelt	namene	namenlike
nameout	namerecv	nameregexeq	nameregexne	namesend	neqjoinsel	neqsel
network_cmp	network_eq	network_ge	network_gt	network_le	network_lt	network_ne
network_sub	network_subeq	network_sup	network_supeq	nlikejoinsel	nlikesel	numeric
numeric_abs	numeric_accum	numeric_add	numeric_avg	numeric_avg_accum	numeric_avg_collect	numeric_cmp
numeric_collect	numeric_div	numeric_div_trunc	numeric_eq	numeric_exp	numeric_fac	numeric_ge

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

numeric_gt	numeric_in	numeric_inc	numeric_larger	numeric_le	numeric_ln	numeric_log
numeric_lt	numeric_mod	numeric_mul	numeric_ne	numeric_out	numeric_power	numeric_recv
numeric_send	numeric_smaller	numeric_sortsupport	numeric_sqrt	numeric_stddev_pop	numeric_stddev_samp	numeric_sub
numeric_transform	numeric_uminus	numeric_uplus	numeric_var_pop	numeric_var_samp	numerictypmodin	numerictypmodout
numrange_subdiff	oid	oideq	oidge	oidgt	oidin	oidlarger
oidle	oidlt	oidne	oidout	oidrecv	oidsend	oidsmaller
oidvectoreq	oidvectorge	oidvectorgt	oidvectorin	oidvectorle	oidvectorlt	oidvectorne
oidvectorout	oidvectorrecv	oidvectorsend	oidvectortypes	on_pb	on_pl	on_ppath
on_ps	on_sb	on_sl	opaque_in	opaque_out	ordered_set_transition	overlaps
overlay	path	path_add	path_add_pt	path_center	path_contain_pt	path_distance
path_div_pt	path_in	path_inter	path_length	path_mul_pt	path_n_eq	path_n_ge
path_n_gt	path_n_le	path_n_lt	path_npoints	path_out	path_recv	path_send
path_sub_pt	percentile_cont	percentile_cont_float8_final	percentile_cont_interval_final	pg_char_to_encoding	pg_cursor	pg_encoding_max_length
pg_encoding_to_char	pg_extension_config_dump	-	-	pg_node_tree_in	pg_node_tree_out	pg_node_tree_recv

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

pg_node_tree_send	pg_prepared_statement	pg_prepared_xact	pg_notify	pg_stat_get_wal_receiver	pg_show_all_settings	pg_stat_get_bgwriter_stat_reset_time
pg_stat_get_buf_fsync_backend	pg_stat_get_checkpoint_sync_time	pg_stat_get_checkpoint_write_time	pg_stat_get_db_blk_read_time	pg_stat_get_db_blk_write_time	pg_stat_get_db_conflict_all	pg_stat_get_db_conflict_bufferpin
pg_stat_get_db_conflict_snapshot	pg_stat_get_db_conflict_startup_deadlock	pg_switch_xlog	xpath	pg_timezone_abbrevs	pg_timezone_names	pgxc_node_str
plpgsql_call_handler	plpgsql_inline_handler	plpgsql_validator	point_above	point_add	point_below	point_distance
point_div	point_eq	point_horiz	point_in	point_left	point_mul	point_ne
point_out	point_recv	point_right	point_send	point_sub	point_vert	poly_above
poly_below	poly_center	poly_contain	poly_contain_pt	poly_contained	poly_distance	poly_in
poly_left	poly_npoints	poly_out	poly_overabove	poly_overbelow	poly_overlap	poly_overleft
poly_overright	poly_recv	poly_right	poly_same	poly_send	polygon	position
positionjoinsel	positionsel	postgresql_fdw_validator	pow	power	prsd_end	prsd_headline
prsd_lextype	prsd_nexttoken	prsd_start	pt_contained_circle	pt_contained_poly	query_to_xml	query_to_xml_and_xmlschema
query_to_xmlschema	quote_ident	quote_literal	quote_nullable	radians	radius	random
range_adjacent	range_after	range_before	range_cmp	range_contained_by	range_contains	range_contains_elem

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

range_ge	range_eq	range_gt	range_in	range_intersect	range_le	range_lt
range_minus	range_ne	range_out	range_overlaps	range_overleft	range_overright	range_recv
range_send	range_typanalyze	range_union	rank	record_eq	record_ge	record_gt
record_in	record_le	record_lt	record_ne	record_out	record_recv	record_send
regclass	regclassin	regclassout	regclassrecv	regclasssend	regconfigin	regconfigout
regconfigrecv	regconfigsend	regdictionaryin	regdictionaryout	regdictionaryrecv	regdictionarysend	regexeqjoinsel
regexeqsel	regexnejoinsel	regexnesel	regexp_matches	regexp_replace	regexp_split_to_array	regexp_split_to_table
regoperatorin	regoperatorout	regoperatorrecv	regoperatorsend	regoperin	regoperout	regoperrecv
regopersend	regprocedurein	regprocedureout	regprocedurerecv	regproceduresend	regprocin	regprocout
regprocrecv	regprocsend	regr_avgx	regr_avgy	regr_count	regr_intercept	regr_r2
regr_slope	regr_sxx	regr_sxy	regr_syy	regtypein	regtypeout	regtyperecv
regtypesend	reltime	reltimeeq	reltimege	reltimegt	reltimein	reltimele
reltimelt	reltimene	reltimeout	reltimerecv	reltimesend	repeat	replace
reverse	RI_FKey_cascade_del	RI_FKey_cascade_upd	RI_FKey_check_ins	RI_FKey_check_upd	RI_FKey_noaction_del	RI_FKey_noaction_upd

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

RI_FKey_restrict_del	RI_FKey_restrict_upd	RI_FKey_setdefault_del	RI_FKey_setdefault_upd	RI_FKey_setnull_del	RI_FKey_setnull_upd	right
round	row_number	row_to_json	rpad	rtrim	scalargtjoinsel	scalargtsel
scalarltjoinsel	scalarltsel	schema_to_xml	schema_to_xml_and_xmlschema	schema_to_xmlschema	session_user	set_bit
set_byte	set_config	set_masklen	shift_jis_2004_to_euc_jis_2004	shift_jis_2004_to_utf8	sjis_to_euc_jp	sjis_to_mic
sjis_to_utf8	smgrin	smgrout	spg_kd_choose	spg_kd_config	spg_kd_inner_consistent	spg_kd_picksplit
spg_quad_choose	spg_quad_config	spg_quad_inner_consistent	spg_quad_leaf_consistent	spg_quad_picksplit	spg_text_choose	spg_text_config
spg_text_inner_consistent	spg_text_leaf_consistent	spg_text_picksplit	spgbeginscan	spgbuild	spgbuildempty	spgbulkdelete
spgcanreturn	spgcostestimate	spgendscan	spggetbitmap	spggettuple	spginsert	spgmarkpos
spgoptions	spgrescan	spgrestrpos	spgvacuumcleanup	stddev	stddev_pop	stddev_samp
string_agg	string_agg_finalfn	string_agg_transfn	strip	sum	suppress_redundant_updates_trigger	table_to_xml
table_to_xml_and_xmlschema	table_to_xmlschema	tan	text	text_ge	text_gt	text_larger

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

text_le	text_lt	text_pattern_ge	text_pattern_gt	text_pattern_le	text_pattern_lt	text_smaller
textanycat	textcat	texteq	texticlike	texticnlike	texticregexeq	texticregexne
textin	textlike	textne	textnlike	textout	textrecv	textregexeq
textregexne	textsend	thesaurus_init	thesaurus_lexize	tideq	tidge	tidgt
tidin	tidlarger	tidle	tidlt	tidne	tidout	tidrecv
tidsend	tidsmaller	time	time_cmp	time_eq	time_ge	time_gt
time_hash	time_in	time_larger	time_le	time_lt	time_mi_interval	time_mi_time
time_ne	time_out	time_pl_interval	time_recv	time_send	time_smaller	time_transform
timedate_pl	timemi	timepl	timestamp	timestamp_cmp	timestamp_cmp_date	timestamp_cmp_timestamptz
timestamp_eq	timestamp_eq_date	timestamp_eq_timestamptz	timestamp_ge	timestamp_ge_date	timestamp_ge_timestamptz	timestamp_gt
timestamp_gt_date	timestamp_gt_timestamptz	timestamp_hash	timestamp_in	timestamp_larger	timestamp_le	timestamp_le_date
timestamp_le_timestamptz	timestamp_lt	timestamp_lt_date	timestamp_lt_timestamptz	timestamp_mi	timestamp_mi_interval	timestamp_ne
timestamp_ne_date	timestamp_ne_timestamptz	timestamp_out	timestamp_pl_interval	timestamp_recv	timestamp_send	timestamp_smaller

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

timestamp_sortsupport	timestamp_transform	timestamptypmodin	timestamptypmodout	timestamptz	timestamptz_cmp	timestamptz_cmp_date
timestamptz_cmp_timestamp	timestamptz_eq	timestamptz_eq_date	timestamptz_eq_timestamp	timestamptz_ge	timestamptz_ge_date	timestamptz_ge_timestamp
timestamptz_gt	timestamptz_gt_date	timestamptz_gt_timestamp	timestamptz_in	timestamptz_larger	timestamptz_le	timestamptz_le_date
timestamptz_le_timestamp	timestamptz_lt	timestamptz_lt_date	timestamptz_lt_timestamp	timestamptz_mi	timestamptz_mi_interval	timestamptz_ne
timestamptz_ne_date	timestamptz_ne_timestamp	timestamptz_out	timestamptz_pl_interval	timestamptz_recv	timestamptz_send	timestamptz_smaller
timestamptztypmodin	timestamptztypmodout	timetypmodin	timetypmodout	timetz	timetz_cmp	timetz_eq
timetz_ge	timetz_gt	timetz_hash	timetz_in	timetz_larger	timetz_le	timetz_lt
timetz_mi_interval	timetz_ne	timetz_out	timetz_pl_interval	timetz_recv	timetz_send	timetz_smaller
timetzdate_pl	timetztypmodin	timetztypmodout	timezone (2069)	timezone (1159)	timezone (2037)	timezone (2070)
timezone (1026)	timezone (2038)	tintervalct	tintervaleq	tintervalge	tintervalgt	tintervalin
tintervalle	tintervalleneq	tintervallenge	tintervallengt	tintervallenle	tintervallenlt	tintervallenne
tintervallt	tintervalne	tintervalout	tintervalov	tintervalrecv	tintervalsame	tintervalsend

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

tintervalstart	to_ascii (1845)	to_ascii (1847)	to_ascii (1846)	trigger_in	trigger_out	ts_match_qv
ts_match_tq	ts_match_tt	ts_match_vq	ts_rank	ts_rank_cd	ts_rewrite	ts_stat
ts_token_type	ts_typanalyze	tsmatchjoinsel	tsmatchsel	tsq_mcontained	tsq_mcontains	tsquery_and
tsquery_cmp	tsquery_eq	tsquery_ge	tsquery_gt	tsquery_le	tsquery_lt	tsquery_ne
tsquery_not	tsquery_or	tsqueryin	tsqueryout	tsqueryrecv	tsquerysend	tsrange
tsrange_subdiff	tstzrange	tstzrange_subdiff	tsvector_cmp	tsvector_concat	tsvector_eq	tsvector_ge
tsvector_gt	tsvector_le	tsvector_lt	tsvector_ne	tsvector_update_trigger	tsvector_update_trigger_column	tsvectorin
tsvectorout	tsvectorrecv	tsvectorsend	txid_current	txid_current_snapshot	txid_snapshot_in	txid_snapshot_out
txid_snapshot_recv	txid_snapshot_send	txid_snapshot_xip	txid_snapshot_xmax	txid_snapshot_xmin	txid_visible_in_snapshot	uhc_to_utf8
unique_key_recheck	unknownin	unknownout	unknownrecv	unknownsend	unnest	utf8_to_big5
utf8_to_euc_cn	utf8_to_euc_jis_2004	utf8_to_euc_jp	utf8_to_euc_kr	utf8_to_euc_tw	utf8_to_gb18030	utf8_to_gbk
utf8_to_iso8859	utf8_to_iso8859_1	utf8_to_johab	utf8_to_koi8r	utf8_to_koi8u	utf8_to_shift_jis_2004	utf8_to_sjis
utf8_to_uhc	utf8_to_win	uuid_cmp	uuid_eq	uuid_ge	uuid_gt	uuid_hash
uuid_in	uuid_le	uuid_lt	uuid_ne	uuid_out	uuid_recv	uuid_send

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

var_pop	var_samp	varbit	varbit_in	varbit_out	varbit_recv	varbit_send
varbit_transform	varbitcmp	varbiteq	varbitge	varbitgt	varbitle	varbitlt
varbitne	varbittypmodin	varbittypmodout	varchar	varchar_transform	varcharin	varcharout
varcharrecv	varcharsend	varchartypmodin	varchartypmodout	variance	void_in	void_out
void_recv	void_send	win_to_utf8	win1250_to_latin2	win1250_to_mic	win1251_to_iso	win1251_to_koi8r
win1251_to_mic	win1251_to_win866	win866_to_iso	win866_to_koi8r	win866_to_mic	win866_to_win1251	xideq
xideqint4	xidin	xidout	xidrecv	xidsend	xml	xml_in
xml_is_well_formed	xml_is_well_formed_content	xml_is_well_formed_document	xml_out	xml_recv	xml_send	xmlagg
xmlcomment	xmlconcat2	xmlexists	xmlvalidate	-	year_in	year_out
year_recv	year_send	yeartypmodin	yeartypmodout	year_eq	year_ne	year_lt
year_le	year_gt	year_ge	year_cmp	year_hash	year_larger	year_smaller
year_mi	year_mi_int4	int4_mi_year	year_pl	year_pl_int4	int4_pl_year	int4_year
year_int4	date_year	numeric_year	text_year	time_year	timestamp_year	timestamptz_year
bpcharlikebpchar	bpcharnlikebpchar	-	-	-	-	-

NOTE

In upgrade mode, system functions, for example, concat, with variable-length parameters cannot be called.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Internal Functions

The following lists the functions used by GaussDB to implement internal system functions. You are advised not to use these functions. If you need to use them, contact Huawei technical support.

- spread_collect()

  Description: Calculates the difference between the maximum and minimum values in a certain period of time. This function is used for data collection of aggregate functions.

  Parameters: s real[], v real[]

  Return type: real[]

- spread_final()

  Description: Calculates the difference between the maximum and minimum values in a certain period of time. This function is used for the final data processing of the aggregate function.

  Parameter: s real[]

  Return type: real

- spread_internal()

  Description: Calculates the difference between the maximum and minimum values in a certain period of time. This function is used for the intermediate data processing of aggregate functions.

  Parameters: s real[], v real

  Return type: real[]

- xidin4()

  Description: Inputs a 4-byte xid.

  Parameter: cstring

  Return type: xid32

- gap_fill_internal()

  Description: Returns the first non-NULL value in the parameter list.

  Parameter: s anyelement, v anyelement

  Return type: anyelement

- int1send()

  Description: Packs unsigned 1-byte integers into the internal data buffer stream.

  Parameter: tinyint

  Return type: bytea

- is_contain_namespace()

  Description: Searches for the table name and namespace split location. If no namespace exists, 0 is returned.

  Parameter: relationname name

  Return type: integer

- is_oid_in_group_members()

  Description: Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter: node_oid oid, group_members oidvector_extend

Return type: Boolean

- isubmit_on_nodes_internal()

  Description: Not supported

  Parameter: job bigint, node_name name, database name, what text, next_date timestamp without time zone, job_interval text

  Return type: integer

- listagg()

  Description: Specifies aggregate functions of the list type.

  Parameters: smallint, text

  Return type: text

- log_fdw_validator()

  Description: Specifies validation functions.

  Parameters: text[], oid

  Return type: void

- nvarchar2typmodin()

  Description: Obtains the typmod information of the varchar type.

  Parameter: cstring[]

  Return type: integer

- nvarchar2typmodout()

  Description: Obtains the typmod information of the varchar type, constructs a character string, and returns the character string.

  Parameter: integer

  Return type: cstring

- pg_nodes_memmon()

  Description: Not supported.

  Parameter: nan

  Return type: innernname text, innerusedmem bigint, innertopctxt bigint, nname text, usedmem text, sharedbuffercache text, topcontext text

- read_disable_conn_file()

  Description: Reads forbidden connection files.

  Parameter: nan

  Return type: disconn_mode text, disconn_host text, disconn_port text, local_host text, local_port text, redo_finished text

- regex_like_m()

  Description: Specifies the regular expression match, which is used to determine whether a character string complies with a specified regular expression.

  Parameters: text, text

  Return type: Boolean

- update_pgjob()

  Description: Updates a job.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: bigint, "char", bigint, timestamp without time zone, timestamp without time zone, timestamp without time zone, timestamp without time zone, timestamp without time zone, smallint, text

Return type: void

- enum_cmp()

  Description: Specifies the enumeration comparison function, which is used to determine whether two enumeration classes are equal and determine their relative sizes.

  Parameters: anyenum, anyenum

  Return type: integer

- enum_eq()

  Description: Specifies the enumeration comparison function, which is used to implement the equal sign (=).

  Parameters: anyenum, anyenum

  Return type: Boolean

- enum_first()

  Description: Returns the first element in the enumeration class.

  Parameter: anyenum

  Return type: anyenum

- enum_ge()

  Description: Specifies the enumeration comparison function, which is used to implement the greater-than sign (>) and equal sign (=).

  Parameters: anyenum, anyenum

  Return type: Boolean

- enum_gt()

  Description: Specifies the enumeration comparison function, which is used to implement the greater-than sign (>).

  Parameters: anyenum, anyenum

  Return type: Boolean

- enum_in()

  Description: Specifies the enumeration comparison function, which is used to determine whether an element is in an enumeration class.

  Parameters: cstring, OID

  Return type: anyenum

- enum_larger()

  Description: Specifies the enumeration comparison function, which is used to implement the greater-than sign (>).

  Parameters: anyenum, anyenum

  Return type: anyenum

- enum_last()

  Description: Returns the last element in the enumeration class.

  Parameter: anyenum

  Return type: anyenum

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- enum_le()

  Description: Specifies the enumeration comparison function, which is used to implement the less-than sign (<) and equal sign (=).

  Parameters: anyenum, anyenum

  Return type: Boolean

- enum_lt()

  Description: Specifies the enumeration comparison function, which is used to implement the less-than sign (<).

  Parameters: anyenum, anyenum

  Return type: Boolean

- enum_smaller()

  Description: Specifies the enumeration comparison function, which is used to implement the less-than sign (<).

  Parameters: anyenum, anyenum

  Return type: Boolean

- node_oid_name()

  Description: Not supported

  Parameter: oid

  Return type: cstring

- pg_buffercache_pages()

  Description: Reads data from the shared buffer.

  Parameter: nan

  Return type: bufferid integer, relfilenode oid, bucketid smallint, storage_type oid, reltablespace oid, reldatabase oid, relforknumber smallint, relblocknumber bigint, isdirty Boolean, usage_count smallint

- pg_check_xidlimit()

  Description: Checks whether nextxid is greater than or equal to xidwarnlimit.

  Parameter: nan

  Return type: Boolean

- pg_comm_delay()

  Description: Displays the delay status of the communications library of a single DN.

  Parameter: nan

  Return type: text, text, integer, integer, integer, integer

- pg_comm_recv_stream()

  Description: Displays the receiving stream status of all communication libraries on a single DN.

  Parameter: nan

  Return type: text, bigint, text, bigint, integer, integer, integer, text, bigint, integer, integer, integer, bigint, bigint, bigint, bigint, bigint

- pg_comm_send_stream()

  Description: Displays the sending stream status of all communication libraries on a single DN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter: nan

Return type: text, bigint, text, bigint, integer, integer, integer, text, bigint, integer, integer, integer, bigint, bigint, bigint, bigint, bigint

- pg_comm_status()

  Description: Displays the communication status of a single DN.

  Parameter: nan

  Return type: text, integer, integer, bigint, bigint, bigint, bigint, bigint, integer, integer, integer, integer, integer

- pg_log_comm_status()

  Description: Prints some logs on the DN.

  Parameter: nan

  Return type: Boolean

- pg_parse_clog()

  Description: Parses Clog to obtain the XID status of an ordinary table.

  Parameter: nan

  Return type: xid xid, status text

- pg_parse_clog(bucketid)

  Description: Parses Clog to obtain the XID status of a specified bucket ID.

  Parameter: **bucketid** indicates the physical bucket ID of the hash bucket table. The value **–1** indicates an ordinary table.

  Return type: xid xid, status text

- pg_pool_ping()

  Description: Sets PoolerPing.

  Parameter: Boolean

  Return type: SETOF Boolean

- pg_pool_validate()

  Description: Compares fields in the **pgxc_node** system catalog to check whether a connection is available.

  Parameter: clear Boolean, co_node_name cstring

  Return type: pid bigint, node_name text

- pg_resume_bkp_flag()

  Description: Obtains the delay Xlog flag for backup and restoration.

  Parameter: slot_name name

  Return type: start_backup_flag Boolean, to_delay Boolean, ddl_delay_recycle_ptr text, rewind_time text

  NOTE

  To call this function, users must possess the SYSADMIN or OPRADMIN permission. In the case of OPRADMIN, **operation_mode** must be enabled.

- gs_static_threadpool_ctrl_status()

  Description: Queries statistics about the static pool threads in the thread pool. When a CN thread pool is enabled, a distributed CN supports the query of related static pool information, and a DN returns a blank line. When the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

thread pool is disabled, a blank line is returned. If **static_thread_pool_num** is set to **0** or less than the number of groups in the thread pool, the default value of the static thread pool is **0**.

Parameter: nan

Return value: text node_name, int group_id, and text worker_info. For details about the return value, see **Table 1-318**.

Table 1-318 GS_STATIC_THREADPOOL_CTRL_STATUS columns

Name	Type	Description
node_name	OUT text	Instance name.
group_id	OUT int	Group ID of the thread pool.
worker_info	OUT text	Dynamic statistics about the running thread pool of the current group. The information includes: **default** and **default_s**: number of threads, including default value of the dynamic thread pool and default value of the static thread pool. **expect** and **expect_s**: number of threads, including the expected value of the dynamic thread pool and expected value of the static thread pool. **actual**: actual number of running threads, including dynamic pool threads and static pool threads. **static threads limit**: number of static pool threads configured for the current group. **has static threads**: specifies whether a static pool is created for the current group. The default value is **0**, indicating no creation. **idle static threads**: number of idle static pool threads. **wait session num**: number of waiting sessions.

- gs_thrdstream_worker_info()

  Description: Queries statistics about the stream thread pool. The query can be performed only on DNs running in thread pool mode.

  Parameter: nan

  Return values: **text node_name**, **int group_id**, **text db_info**, and **text stream_info**. For details about the returned columns, see **Table 1-319**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-319 GS_THRDSTREAM_WORKER_INFO columns

Name	Type	Description
node_name	OUT text	Instance name.
group_id	OUT integer	Group ID of the thread pool.
db_info	OUT text	Dynamic database statistics about the running stream thread pool of the current group. The information includes: • **total stream reusing db**: number of databases that are being used or reused in the stream thread pool. • **free session db**: number of databases that do not have running sessions in the stream thread pool.
stream_info	OUT text	Dynamic stream thread statistics about the running stream thread pool of the current group. The information includes: • **actual**: number of actually started stream threads. • **idle**: number of idle stream threads. • **using**: number of stream threads in use. • **init session threads**: number of stream threads for initializing a session. • **reuse session threads**: number of stream threads that reuse sessions.

- pg_stat_get_pooler_status()

  Description: Queries the cache connection status in the pooler.

  Parameter: nan

  Return type: See **Table 1-320**.

  Table 1-320 Return values of pg_stat_get_pooler_status

Name	Type	Description
database_name	OUT text	Database name.
user_name	OUT text	Username.
tid	OUT bigint	In non-thread pool logic, this parameter indicates the ID of the thread connected to the CN. In thread pool logic, this parameter indicates the ID of the session connected to the CN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
pgoptions	OUT text	Database connection options. For details, see the **options** column in "Application Development Guide > Appendix > libpq > Connection Parameters" in *Developer Guide*.
node_oid	OUT bigint	OID of the node connected.
in_use	OUT boolean	Specifies whether the connection is currently used. • **t** (true): The connection is in use. • **f** (false): The connection is not in use.
session_params	OUT text	GUC session parameter delivered by the connection.
fdsock	OUT bigint	Local socket.
local_host	OUT text	Local IP address.
local_port	OUT bigint	Local port number.
remote_host	OUT text	Peer IP address.
remote_port	OUT bigint	Peer port number.
remote_pid	OUT bigint	If the peer site is not in the thread pool logic, the value is the thread ID of the peer site. If the peer site is in the thread pool logic, the value is the session ID of the peer site.
used_count	OUT bigint	Number of reuse times of a connection.
idx	OUT bigint	Peer DN ID in the local CN.
streamid	OUT bigint	Stream ID in the physical connection.

- gs_stat_get_pooler_agent()

  Description: Queries information about the pooler agent on the connected CN to help locate connection pool problems. Only system administrators can access this function. If a DN is connected for query, an empty table is returned.

  Parameter: nan

  Return type: See **Table 1-321**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-321 Return values of gs_stat_get_pooler_agent()

Name	Type	Description
index	OUT bigint	Index of the current agent in the global agents.
database_name	OUT text	Name of the database in the current session.
user_name	OUT text	Username of the current session.
session_id	OUT bigint	• ID of the thread connected to the CN for non-thread pool logic. • Session ID for connecting to the CN for thread pool logic.
dn_oids	OUT oid[]	List of recorded DN OIDs.
cn_oids	OUT oid[]	List of recorded CN OIDs.
agent_init	OUT timestamp with time zone	Time when the agent is initialized. The agent is initialized when a session is created or the pgxc catalog information is reloaded.

- get_node_modulo()
  Description: Computes indexes of a node in a group.
  Parameters: See **Table 1-322**.

  Table 1-322 get_node_modulo input parameters

Parameter	Type	Description	Value Range
node_oid	IN oid	Node OID.	[0, 2^32-1]
group_oid	IN oid	OID of a node group.	[0, 2^32-1]

  Return type: uint4
- check_murmurhash_route_node()
  Description: Checks whether the node where the key is distributed is correct.
  Parameters: See **Table 1-323**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-323 check_murmurhash_route_node input parameters

Parameter	Type	Description	Value Range
key	IN text	Character string of the hash key.	-
node_modulo	IN uint4	Index of a node in a node group.	[0, 2^32-1]
node_num	IN uint4	Number of nodes in a node group.	[0, 2^32-1]

Return type: uint4

- gs_libcomm_fd_info()

  Description: Queries the socket persistent connection information of the libcomm communications library.

  Parameter: nan

  Return value: See **Table 1-324**.

Table 1-324 Return values of gs_libcomm_fd_info

Name	Type	Description
node_name	OUT text	DN name.
ip	OUT text	IP address of the DN.
ctrl_port	OUT bigint	Control channel port.
data_port	OUT bigint	Data channel port.
remote_name	OUT text	Name of the peer DN.
remote_ip	OUT text	Peer IP address.
remote_port	OUT bigint	Peer port number.
local_ip	OUT text	Local IP address.
local_port	OUT bigint	Local port number.
fd	OUT bigint	Socket connection.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
type	OUT text	Type of the TCP connection channel. The options are as follows: • **DATA_SEND_FD**: socket of the sender of the data channel • **DATA_RECV_FD**: socket of the receiver of the data channel. • **CTL_SEND_FD**: socket of the sender of the control channel. • **CTL_RECV_FD**: socket of the receiver of the control channel.

NOTE

This function can be used only on the DN. If no information is returned, the libcomm persistent connection is not established for the current DN instance.

- gs_libcomm_memory_info()

  Description: Queries the basic configuration information used by the internal memory of the current libcomm communications library.

  Parameter: nan

  Return value: See **Table 1-325**.

  Table 1-325 Return values of gs_libcomm_memory_info

Name	Type	Description
node_name	OUT text	Instance name of the DN.
current_used_memory	OUT bigint	Memory used by the libcomm communications library, in bytes.
current_data_item_num	OUT bigint	Number of data blocks in the libcomm lock-free queue.
init_mailbox_memory	OUT bigint	Memory used by libcomm to initialize the mailbox, in bytes.
max_datanode	OUT bigint	Maximum number of DNs supported by the libcomm communications library.
max_stream	OUT bigint	Maximum number of data streams supported by a single TCP connection of the libcomm.
max_quota_size	OUT bigint	Total buffer size of the libcomm logical connection, in bytes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
max_usable_memory	OUT bigint	Maximum memory that can be used by the libcomm communications library cache, in bytes.
max_memory_pool	OUT bigint	Total size of the memory resource pool that can be used by the libcomm communications library on each DN. The unit is byte.

- gs_validate_ext_listen_ip()

  Description: Connects to DNs, queries invalid worker threads on DNs connected to the original extended IP address, and clears the threads.

  Parameters: See **Table 1-326**.

  Return value: bigint pid, text node_name

  Note: This function is used only when old connections still exist on the original extended IP address after the extended IP address is reloaded. This function clears the threads where the original IP connections are located. Executing this function will clear communication listening on the extended IP address. Exercise caution when performing O&M operations. Currently, this command can be executed only by connecting to DNs and delivered by the administrator.

  Table 1-326 GS_VALIDATE_EXT_LISTEN_IP columns

Name	Type	Description
clear	IN cstring	Specifies whether to clear data. The options are as follows: **on**: clear; **off**: not clear.
validate_node_name	IN cstring	Name of the DN where the extended IP address connections to be cleared are located.
validate_ip	IN cstring	Extended IP address whose connections are to be cleared.
pid	OUT bigint	ID of the worker thread where the extended IP address connections to be cleared are located.
node_name	OUT text	Name of the instance to which the extended IP address connections to be cleared belong.

- gs_comm_listen_address_ext_info()

  Description: Displays the DFX information about the extended IP address configured for **listen_address_ext** connected to the current instance.

  Parameter: nan

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: text node_name, text app, bigint tid, integer lwtid, bigint query_id, integer socket, text remote_ip, text remote_port, text local_ip or text local_port For details about the returned columns, see **Table 1-327**.

Table 1-327 GS_COMM_LISTEN_ADDRESS_EXT_INFO columns

Name	Type	Description
node_name	OUT text	Name of the current instance.
app	OUT text	Client connected to the DN.
tid	OUT bigint	Thread ID of the current thread.
lwtid	OUT integer	Lightweight thread ID of the current thread.
query_id	OUT bigint	Query ID of the current thread.
socket	OUT integer	Socket FD of the current physical connection.
remote_ip	OUT text	Peer IP address of the current connection.
remote_port	OUT text	Peer port of the current connection.
local_ip	OUT text	Local IP address of the current connection.
local_port	OUT text	Local port of the current connection.

- gs_get_global_listen_address_ext_info

  Description: Queries the global extended IP address configuration on the CN.

  Parameters: See **Table 1-328**.

  Return type: text node_name, text host, text port, text ext_listen_ip

  Note: If the input parameter of the function is **all**, the function depends on the **ext_ip_info** static configuration file. If the file is abnormal, use the CM/OM tool to rectify the fault. In this case, the configuration cannot be updated in real time. You can set the input parameter to **primary** to obtain the configuration information about all primary DNs.

  Table 1-328 GS_GET_GLOBAL_LISTEN_ADDRESS_EXT_INFO columns

Name	Type	Description
dn_mode	IN cstring	Specifies the range of DNs to be displayed. If this parameter is set to **null**, all DNs are queried by default.
node_name	OUT text	DN name.
host	OUT text	Listening IP address of the DN.
port	OUT text	Listening port of the DN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
ext_listen_ip	OUT text	Extended listening IP address of the DN.

- gs_get_listen_address_ext_info()

  Description: Queries the extended IP address configuration of the current instance.

  Parameter: nan

  Return type: text node_name, text host, bigint port, text ext_listen_ip For details about the returned columns, see **Table 1-329**.

  Table 1-329 GS_GET_LISTEN_ADDRESS_EXT_INFO

Name	Type	Description
node_name	OUT text	DN name.
host	OUT text	Listening IP address of the DN.
port	OUT bigint	Listening port of the DN.
ext_listen_ip	OUT text	Extended listening IP address of the DN.

- psortoptions()

  Description: Returns the psort attribute.

  Parameters: text[], Boolean

  Return type: bytea
- remove_job_class_depend()

  Description: Removes the job dependency.

  Parameter: oid

  Return type: void
- xideq4()

  Description: Compares two xids to check whether they are the same.

  Parameters: xid32, xid32

  Return type: Boolean
- xideqint8()

  Description: Compares two xids to check whether they are the same.

  Parameters: xid, bigint

  Return type: Boolean
- xidlt()

  Description: Returns whether xid1 < xid2 is true.

  Parameters: xid, xid

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- xidlt4()

  Description: Returns whether xid1 < xid2 is true.

  Parameters: xid32, xid32

  Return type: Boolean

- get_local_cont_query_stat()

  Description: Obtains the statistics of a specified continuous computing view on the local node.

  Parameter: cq_id oid

  Return type: cq oid, w_in_rows int8, w_in_bytes int8, w_out_rows int8, w_out_bytes int8, w_pendings int8, w_errors int8, r_in_rows int8, r_in_bytes int8, r_out_rows int8, r_out_bytes int8, r_errors int8, c_in_rows int8, c_in_bytes int8, c_out_rows int8, c_out_bytes int8, c_pendings int8, c_errors int8

- get_local_cont_query_stats()

  Description: Obtains all continuous computing view statistics of the local node.

  Parameter: nan

  Return type: cq oid, w_in_rows int8, w_in_bytes int8, w_out_rows int8, w_out_bytes int8, w_pendings int8, w_errors int8, r_in_rows int8, r_in_bytes int8, r_out_rows int8, r_out_bytes int8, r_errors int8, c_in_rows int8, c_in_bytes int8, c_out_rows int8, c_out_bytes int8, c_pendings int8, c_errors int8

- get_cont_query_stats()

  Description: Obtains statistics about all continuous computing views on each DN.

  Parameter: nan

  Return type: node name, cq oid, w_in_rows int8, w_in_bytes int8, w_out_rows int8, w_out_bytes int8, w_pendings int8, w_errors int8, r_in_rows int8, r_in_bytes int8, r_out_rows int8, r_out_bytes int8, r_errors int8, c_in_rows int8, c_in_bytes int8, c_out_rows int8, c_out_bytes int8, c_pendings int8, c_errors int8

- reset_local_cont_query_stat()

  Description: Resets the statistics of a specified continuous computation view on the local node.

  Parameter: cq_id oid

  Return type: Boolean

- reset_local_cont_query_stats()

  Description: Resets association statistics on the specified continuous computation view of the local node.

  Parameter: cq_id oid

  Return type: Boolean

- reset_cont_query_stats()

  Description: Resets the continuous computation view statistics corresponding to the STREAM object on each DN.

  Parameter: stream_id oid

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- check_cont_query_schema_changed()

  Description: Determines the schema change status of a specified continuous computation view.

  Parameter: cq_id oid

  Return type: Boolean

- gs_get_standby_cluster_barrier_status()

  Description: Queries the barrier log replay information of the standby CN or DN, including the latest received barrier, LSN of the latest received barrier, barrier replayed last time, and target barrier to be replayed.

  Parameter: nan

  Return type: barrier_id text, barrier_lsn text, recovery_id text, target_id text

  Note: To call this function, the user must have the SYSADMIN or OPRADMIN permission. For the O&M administrator role, **operate_mode** must be enabled.

- gs_set_standby_cluster_target_barrier_id()

  Description: Set the target barrier to be replayed.

  Parameter: barrier_id

  Return type: target_id text

  Note: To call this function, the user must have the SYSADMIN or OPRADMIN permission. For the O&M administrator role, **operate_mode** must be enabled.

- gs_query_standby_cluster_barrier_id_exist()

  Description: Queries whether the specified barrier is received by the standby node.

  Parameter: barrier_id

  Return type: Boolean

  Note: To call this function, the user must have the SYSADMIN or OPRADMIN permission. For the O&M administrator role, **operate_mode** must be enabled.

- standby_read_status()

  Description: Obtains the CSN-based snapshot information of all DNs in a cluster, the difference between the primary and standby DNs, and the CSN-based snapshot information used for the last read on standby. This is used for read on standby in the distributed GTM-free scenario.

  Parameter: nan

  Return type: text, text, text, integer, bigint, bigint, bigint For details about the returned columns, see **Table 1-330**.

  Table 1-330 standby_read_status columns

Name	Type	Description
name	text	DN name.
type	text	DN type (**primary** for the primary node and **standby** for the standby node).
host	text	IP address used by the DN.
port	integer	Port used by the DN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
collected_csn	bigint	CSN-based snapshot information of the DN (For the primary node, the value is **0**. For the standby node, the value is the collected CSN.)
delay	bigint	Difference between the primary and standby DNs. The unit is ms.
visited_csn	bigint	CSN information used for the last read query on the standby node. (This parameter is valid only for the standby node where the last query is executed.)

- gs_shutdown_cross_region_walsenders()

  Description: Interrupts cross-cluster streaming replication.

  Parameter: nan

  Return type: void

  Note: To call this function, the user must have the SYSADMIN or OPRADMIN permission. For the O&M administrator role, **operate_mode** must be enabled.

  The following stream functions exist but are not supported. You are advised not to use them.

  streaming_int8_avg_gather, streaming_numeric_avg_gather, streaming_float8_avg_gather, streaming_interval_avg_gather, streaming_int8_sum_gather, streaming_int2_int4_sum_gather
- is_dblink_in_transaction()

  Description: Checks whether a database link corresponding to an OID is used in the current transaction.

  Parameter: oid

  Return type: Boolean
- dblink_has_updatasent()

  Description: Checks whether DML statements are sent using a database link corresponding to an OID in the current transaction and are not committed.

  Parameter: oid

  Return type: Boolean
- get_last_xmin_by_oid()

  Description: Obtains the maximum **xmin** value of all columns in a table based on the table OID.

  Parameter: oid

  Return type: xid
- get_relid_by_relname()

  Description: Obtains the OID of a table based on the table name and relnamespace.

  Parameters: cstring, OID

  Return type: OID

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- btint12cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int1, int2

  Return type: integer

- btint14cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int1, int4

  Return type: integer

- btint18cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int1, int8

  Return type: integer

- btint116cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int1, int16

  Return type: integer

- btint1numericcmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int1, numeric

  Return type: integer

- btint21cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int2, int1

  Return type: integer

- btint216cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

another one, **0** is returned. If the value is less than another one, a negative number is returned.

Parameters: int2, int16

Return type: integer

- btint2numericcmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int2, numeric

  Return type: integer

- btint41cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int4, int1

  Return type: integer

- btint416cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int4, int16

  Return type: integer

- btint4numericcmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int4, numeric

  Return type: integer

- btint81cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int8, int1

  Return type: integer

- btint816cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int8, int16

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: integer

- btint8numericcmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int8, numeric

  Return type: integer

- btint161cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int16, int1

  Return type: integer

- btint162cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int16, int2

  Return type: integer

- btint164cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int16, int4

  Return type: integer

- btint168cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int16, int8

  Return type: integer

- btnumericint1cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: numeric, int1

  Return type: integer

- btnumericint2cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

another one, **0** is returned. If the value is less than another one, a negative number is returned.

Parameters: numeric, int2

Return type: integer

- btnumericint4cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: numeric, int4

  Return type: integer

- btnumericint8cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: numeric, int8

  Return type: integer

- btint16cmp()

  Description: Compares the values of two parameters. If the value is greater than another one, a positive number is returned. If the value is equal to another one, **0** is returned. If the value is less than another one, a negative number is returned.

  Parameters: int16, int16

  Return type: integer

- hashint16()

  Description: Calculates the hash value of an input parameter.

  Parameter: int16

  Return type: integer

- hashint1_numeric()

  Description: Calculates the hash value of an input parameter.

  Parameter: int1

  Return type: integer

- hashint2_numeric()

  Description: Calculates the hash value of an input parameter.

  Parameter: int2

  Return type: integer

- hashint4_numeric()

  Description: Calculates the hash value of an input parameter.

  Parameter: int4

  Return type: integer

- hashint8_numeric()

  Description: Calculates the hash value of an input parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter: int8

Return type: integer

- int12eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int2

  Return type: Boolean

- int14eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int4

  Return type: Boolean

- int18eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int8

  Return type: Boolean

- int116eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int16

  Return type: Boolean

- int1numericeq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, numeric

  Return type: Boolean

- int21eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, int1

  Return type: Boolean

- int216eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, int16

  Return type: Boolean

- int2numericeq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, numeric

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- int41eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, int1

  Return type: Boolean

- int416eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, int16

  Return type: Boolean

- int4numericeq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, numeric

  Return type: Boolean

- int81eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, int1

  Return type: Boolean

- int816eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, int16

  Return type: Boolean

- int8numericeq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, numeric

  Return type: Boolean

- int161eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int1

  Return type: Boolean

- int162eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int2

  Return type: Boolean

- int164eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: int16, int4

Return type: Boolean

- int168eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int8

  Return type: Boolean

- numericint1eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int1

  Return type: Boolean

- numericint2eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int2

  Return type: Boolean

- numericint4eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int4

  Return type: Boolean

- numericint8eq()

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int8

  Return type: Boolean

- int12ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int2

  Return type: Boolean

- int14ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int4

  Return type: Boolean

- int18ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int8

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- int116ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int16

  Return type: Boolean

- int1numericne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, numeric

  Return type: Boolean

- int21ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, int1

  Return type: Boolean

- int216ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, int16

  Return type: Boolean

- int2numericne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, numeric

  Return type: Boolean

- int41ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, int1

  Return type: Boolean

- int416ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, int16

  Return type: Boolean

- int4numericne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, numeric

  Return type: Boolean

- int81ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: int8, int1

Return type: Boolean

- int816ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, int16

  Return type: Boolean

- int8numericne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, numeric

  Return type: Boolean

- int161ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int1

  Return type: Boolean

- int162ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int2

  Return type: Boolean

- int164ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int4

  Return type: Boolean

- int168ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int8

  Return type: Boolean

- numericint1ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int1

  Return type: Boolean

- numericint2ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int2

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- numericint4ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int4

  Return type: Boolean

- numericint8ne()

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int8

  Return type: Boolean

- int12lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int2

  Return type: Boolean

- int14lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int4

  Return type: Boolean

- int18lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int8

  Return type: Boolean

- int116lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int16

  Return type: Boolean

- int1numericlt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, numeric

  Return type: Boolean

- int21lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: int2, int1

Return type: Boolean

- int216lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, int16

  Return type: Boolean

- int2numericlt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, numeric

  Return type: Boolean

- int41lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, int1

  Return type: Boolean

- int416lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, int16

  Return type: Boolean

- int4numericlt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, numeric

  Return type: Boolean

- int81lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, int1

  Return type: Boolean

- int816lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, int16

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- int8numericlt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, numeric

  Return type: Boolean

- int161lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int1

  Return type: Boolean

- int162lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int2

  Return type: Boolean

- int164lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int4

  Return type: Boolean

- int168lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int8

  Return type: Boolean

- numericint1lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int1

  Return type: Boolean

- numericint2lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int2

  Return type: Boolean

- numericint4lt()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

Parameters: numeric, int4

Return type: Boolean

- numericint8lt()

  Description: Compares whether a parameter is less than another one. If the parameter is less than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int8

  Return type: Boolean

- int12gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int2

  Return type: Boolean

- int14gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int4

  Return type: Boolean

- int18gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int8

  Return type: Boolean

- int116gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int16

  Return type: Boolean

- int1numericgt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, numeric

  Return type: Boolean

- int21gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: int2, int1

Return type: Boolean

- int216gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, int16

  Return type: Boolean

- int2numericgt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, numeric

  Return type: Boolean

- int41gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, int1

  Return type: Boolean

- int416gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, int16

  Return type: Boolean

- int4numericgt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, numeric

  Return type: Boolean

- int81gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, int1

  Return type: Boolean

- int816gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, int16

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- int8numericgt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, numeric

  Return type: Boolean

- int161gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int1

  Return type: Boolean

- int162gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int2

  Return type: Boolean

- int164gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int4

  Return type: Boolean

- int168gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int8

  Return type: Boolean

- numericint1gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int1

  Return type: Boolean

- numericint2gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int2

  Return type: Boolean

- numericint4gt()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

Parameters: numeric, int4

Return type: Boolean

- numericint8gt()

  Description: Compares whether a parameter is greater than another one. If the parameter is greater than another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int8

  Return type: Boolean

- int12le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int2

  Return type: Boolean

- int14le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int4

  Return type: Boolean

- int18le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int8

  Return type: Boolean

- int116le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int16

  Return type: Boolean

- int1numericle()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, numeric

  Return type: Boolean

- int21le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: int2, int1

Return type: Boolean

- int216le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, int16

  Return type: Boolean

- int2numericle()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, numeric

  Return type: Boolean

- int41le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, int1

  Return type: Boolean

- int416le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, int16

  Return type: Boolean

- int4numericle()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, numeric

  Return type: Boolean

- int81le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, int1

  Return type: Boolean

- int816le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, int16

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- int8numericle()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, numeric

  Return type: Boolean

- int161le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int1

  Return type: Boolean

- int162le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int2

  Return type: Boolean

- int164le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int4

  Return type: Boolean

- int168le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int8

  Return type: Boolean

- numericint1le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int1

  Return type: Boolean

- numericint2le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int2

  Return type: Boolean

- numericint4le()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

Parameters: numeric, int4

Return type: Boolean

- numericint8le()

  Description: Compares whether a parameter is less than or equal to another one. If the parameter is less than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int8

  Return type: Boolean

- int12ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int2

  Return type: Boolean

- int14ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int4

  Return type: Boolean

- int18ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int8

  Return type: Boolean

- int116ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, int16

  Return type: Boolean

- int1numericge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int1, numeric

  Return type: Boolean

- int21ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: int2, int1

Return type: Boolean

- int216ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, int16

  Return type: Boolean

- int2numericge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int2, numeric

  Return type: Boolean

- int41ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, int1

  Return type: Boolean

- int416ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, int16

  Return type: Boolean

- int4numericge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int4, numeric

  Return type: Boolean

- int81ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, int1

  Return type: Boolean

- int816ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, int16

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- int8numericge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int8, numeric

  Return type: Boolean

- int161ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int1

  Return type: Boolean

- int162ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int2

  Return type: Boolean

- int164ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int4

  Return type: Boolean

- int168ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: int16, int8

  Return type: Boolean

- numericint1ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int1

  Return type: Boolean

- numericint2ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int2

  Return type: Boolean

- numericint4ge()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

Parameters: numeric, int4

Return type: Boolean

- numericint8ge()

  Description: Compares whether a parameter is greater than or equal to another one. If the parameter is greater than or equal to another one, **true** is returned. Otherwise, **false** is returned.

  Parameters: numeric, int8

  Return type: Boolean

- timestamp_in

  Description: Converts the input cstring type to the timestamp type.

  Parameter: cstring

  Return type: timestamp

- timestamp_out

  Description: Converts the timestamp type to the cstring type.

  Parameter: timestamp

  Return type: cstring

- datea_cmp

  Description: Compares two parameters. If the first parameter is greater than the second parameter, **1** is returned. If the first parameter is less than the second parameter, **–1** is returned. If the two parameters are equal, **0** is returned.

  Parameters: datea, datea

  Return type: int4

- datea_eq

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: datea, datea

  Return type: Boolean

- datea_ge

  Description: Returns **true** if the first parameter is greater than or equal to the second parameter; otherwise, returns **false**.

  Parameters: datea, datea

  Return type: Boolean

- datea_gt

  Description: Returns **true** if the first parameter is greater than the second parameter; otherwise, returns **false**.

  Parameters: datea, datea

  Return type: Boolean

- datea_le

  Description: Returns **true** if the first parameter is less than or equal to the second parameter; otherwise, returns **false**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: datea, datea

Return type: Boolean

- datea_lt

  Description: Returns **true** if the first parameter is less than the second parameter; otherwise, returns **false**.

  Parameters: datea, datea

  Return type: Boolean

- datea_neq

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: datea, datea

  Return type: Boolean

- datea_pl_interval

  Description: Returns the result of adding the datea type to the interval type.

  Parameters: datea, interval

  Return type: datea

- interval_pl_datea

  Description: Returns the result of adding the interval type to the datea type.

  Parameters: interval, datea

  Return type: datea

- datea_pl_float4

  Description: Adds a float4 type value to a datea type value and returns the result.

  Parameters: **datea** and **float4**.

  Return type: datea

- float4_pl_datea

  Description: Adds a datea type value to a float4 type value and returns the result.

  Parameters: **float4** and **datea**.

  Return type: datea

- datea_pl_float8

  Description: Adds a float8 type value to a datea type value and returns the result.

  Parameters: **datea** and **float8**.

  Return type: datea

- float8_pl_datea

  Description: Adds a datea type value to a float8 type value and returns the result.

  Parameters: **float8** and **datea**.

  Return type: datea

- datea_pl_int8

  Description: Adds an int8 type value to a datea type value and returns the result.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: **datea** and **int8**.

Return type: datea

- int8_pl_datea

  Description: Adds a datea type value to an int8 type value and returns the result.

  Parameters: **int8** and **datea**.

  Return type: datea

- datea_pl_text

  Description: Subtracts a text type value from a datea type value and returns the result.

  Parameters: **datea** and **text**.

  Return type: datea

- text_pl_datea

  Description: Adds a datea type value to a text type value and returns the result.

  Parameters: **text** and **datea**.

  Return type: datea

- numeric_pl_datea

  Description: Adds a datea type value to a numeric type value and returns the result.

  Parameters: **numeric** and **datea**.

  Return type: datea

- datea_mi_interval

  Description: Returns the result of subtracting the datea type from the interval type.

  Parameters: datea, interval

  Return type: datea

- datea_mi

  Description: Returns the subtraction result of two datea types.

  Parameters: datea, datea

  Return type: numeric

- datea_mi_timestamp

  Description: Returns the result of subtracting the datea type from the timestamp type.

  Parameters: datea, timestamp

  Return type: interval

- timestamp_mi_datea

  Description: Returns the result of subtracting the timestamp type from the datea type.

  Parameters: timestamp, datea

  Return type: interval

- datea_mi_float4

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Subtracts a float4 type value from a datea type value and returns the result.

Parameters: **datea** and **float4**.

Return type: datea

- datea_mi_float8

  Description: Subtracts a float8 type value from a datea type value and returns the result.

  Parameters: **datea** and **float8**.

  Return type: datea

- datea_mi_int8

  Description: Subtracts an int8 type value from a datea type value and returns the result.

  Parameters: **datea** and **int8**.

  Return type: datea

- datea_mi_numeric

  Description: Subtracts a numeric type value from a datea type value and returns the result.

  Parameters: **datea** and **numeric**.

  Return type: datea

- datea_mi_text

  Description: Subtracts a text type value from a datea type value and returns the result.

  Parameters: **datea** and **text**.

  Return type: datea

- datea_larger

  Description: Returns the larger value of two datea types.

  Parameters: datea, datea

  Return type: datea

- datea_smaller

  Description: Returns the smaller value of two datea types.

  Parameters: datea, datea

  Return type: datea

- datea_in

  Description: Converts the input cstring type to the datea type.

  Parameter: cstring

  Return type: datea

- datea_out

  Description: Converts the datea type to the cstring type.

  Parameter: datea

  Return type: cstring

- datea_recv

  Description: Converts binary data to the datea type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter: internal

Return type: datea

- datea_send

  Description: Converts the datea type to binary data.

  Parameter: datea

  Return type: bytea

- rowid_cmp(rowid, rowid)

  Description: Compares two parameters. If the first parameter is greater than the second parameter, **1** is returned. If the first parameter is less than the second parameter, **–1** is returned. If the two parameters are equal, **0** is returned.

  Parameters: rowid and rowid.

  Return type: int4

- rowid_eq(rowid, rowid)

  Description: Compares whether two parameter values are equal. If they are equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: rowid and rowid.

  Return type: Boolean

- rowid_ge(rowid, rowid)

  Description: Returns **true** if the first parameter is greater than or equal to the second parameter; otherwise, returns **false**.

  Parameters: rowid and rowid.

  Return type: Boolean

- rowid_gt(rowid, rowid)

  Description: Returns **true** if the first parameter is greater than the second parameter; otherwise, returns **false**.

  Parameters: rowid and rowid.

  Return type: Boolean

- rowid_le(rowid, rowid)

  Description: Returns **true** if the first parameter is less than or equal to the second parameter; otherwise, returns **false**.

  Parameters: rowid and rowid.

  Return type: Boolean

- rowid_lt(rowid, rowid)

  Description: Returns **true** if the first parameter is less than the second parameter; otherwise, returns **false**.

  Parameters: rowid and rowid.

  Return type: Boolean

- rowid_ne(rowid, rowid)

  Description: Compares whether two parameter values are equal. If they are not equal, **true** is returned. Otherwise, **false** is returned.

  Parameters: rowid and rowid.

  Return type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- rowid_larger(rowid, rowid)

  Description: Returns the larger value of two rowid types.

  Parameters: rowid and rowid.

  Return type: rowid.
- rowid_smaller(rowid, rowid)

  Description: Returns the smaller value of two rowid types.

  Parameters: rowid and rowid.

  Return type: rowid.
- rowid_in(cstring)

  Description: Converts the input cstring type to the rowid type.

  Parameter: cstring

  Return type: rowid.
- rowid_out(rowid)

  Description: Converts the rowid type to the cstring type.

  Parameter: rowid

  Return type: cstring
- rowid_recv(internal)

  Description: Converts binary data to the rowid type.

  Parameter: internal

  Return type: rowid.
- rowid_send(rowid)

  Description: Converts the rowid type to binary data.

  Parameter: rowid

  Return type: bytea
- gs_relation_is_updatable()

  Description: Returns whether a relationship can be inserted, updated, or deleted. If the value can be updated, **4** is returned. If the value can be inserted, **8** is returned. If the value can be deleted, **16** is returned. If the value meets the requirements, the value obtained by adding the corresponding values is returned.

  Parameters: OID, Boolean

  Return type: int32
- gs_column_is_updatable()

  Description: Returns whether a column can be updated. If the column can be updated, **t** is returned. Otherwise, **f** is returned.

  Parameters: OID, int16, Boolean

  Return type: Boolean
- hash_meet_partition(partition_num, hash_key_name1, hash_key_name2)

  Description: Calculates the hash partition where a tuple is located when an online DDL statement is used to exchange partitions. Returns the index of the partition where the tuple is located.

  Parameter: int32, text

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: int32

Parameter	Type	Description	Value Range
partition_num	int32	Total number of partitions in a partitioned table.	-
hash_key_name	text	Partition key name of the hash partitioned table. The number of variable parameters is the same as the number of partition keys.	-

Example:

```
-- Create a table.
gaussdb=# CREATE TABLE t1(c1 INT,c2 INT) PARTITION BY HASH(c2)
(
    PARTITION P1,
    PARTITION P2,
    PARTITION P3
);

gaussdb=# INSERT INTO t1 VALUES(generate_series(1,10), generate_series(1,10));

gaussdb=# SELECT * FROM t1 WHERE hash_meet_partition(3, c2) = 0;
 c1 | c2
----+----
  1 |  1
  4 |  4
  8 |  8
  9 |  9
 10 | 10
(5 rows)

gaussdb=# SELECT * FROM t1 partition (p1);
 c1 | c2
----+----
  1 |  1
  4 |  4
  8 |  8
  9 |  9
 10 | 10
(5 rows)
```

- online_ddl_cleanup(relschema, relname)

  Description: Clears and restores temporary tools when a fault occurs during online DDL or online VACUUM FULL execution.

  Parameters: text, text

  Return type: text

Parameter	Type	Description	Value Range
relschema	text	Schema of a table.	-
relname	text	Name of a table.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- **online_ddl_cleanup** is not supported on the standby node.
- After a fault occurs, the temporary status of the table is automatically cleared when the online DDL or online VACUUM FULL is executed next time.

Example:

```
-- Create a table.
gaussdb=# CREATE TABLE t1(c1 int, c2 int);
-- Execute the online DDL.
gaussdb=# ALTER TABLE ONLINE t1 ALTER c2 type BIGINT;

-- Clear a fault after it occurs.
gaussdb=# SELECT online_ddl_cleanup('public','t1');
 online_ddl_cleanup
--------------------
 cleanup success
(1 row)
```

- gs_stat_progress_online_ddl(relschema, relname)

  Description: Queries the status and progress of the ongoing online VACUUM FULL operation in the system.

  Parameters: Both parameters can be left empty, set to **NULL**, or not set to **NULL** at the same time.

  Return type: table. Each row represents a phase of a DDL operation.

Parameter	Type	Description	Value Range
relschema	cstring	Schema where online VACUUM FULL is executed.	-
relname	cstring	Name of the table where online VACUUM FULL is executed.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

1. In the current version, only the progress of online VACUUM FULL can be monitored. The progress of other online DDL operations cannot be monitored.
2. If the **relschema** and **relname** parameters are not empty, you can query the residual online DDL information of the table. If both **relschema** and **relname** are null or left empty, you can query information about all online VACUUM FULL operations that are being executed.
3. The temporary table of online DDL operations is located in the **online$$ddl$$ [***Hash value of (Schema name of the table + Table name)***]** schema.
4. If the GUC parameter **delete_cctmp_table** is set to **on**, the schema of the temporary table will be deleted after the online DDL operation is complete. If this parameter is set to **off**, the schema will be renamed to **online$$ddl$$[***Hash value of (Schema name of the table + Table name)***]_[***Date and time***]**.
5. If the current user is the initial user or a system administrator, the user can query the progress of all online VACUUM FULL operations that are being executed. If the current user is a common user, the user can only query the progress of the tasks executed by themselves.
6. The progress monitoring information is quasi-real-time. That is, the monitoring function returns the latest phase in real time, but the progress information in each phase is not updated in real time.

Example:

```
-- Session 1: Execute online VACUUM FULL.
gaussdb=# VACUUM FULL t1 online;
-- Session 2: Execute the progress monitoring function.
gaussdb=# \x -- Modify the output formatting.
gaussdb=# select * from gs_stat_progress_online_ddl('public', 't1') ;
-[ RECORD 1 ]------+--------------------------------------------------------------------------
pid                | 139717555775232
node_name          | coordinator2
phase_id           | 1
rel_schema         | public
rel_name           | t1
sql_command        | VACUUM FULL t1 online with (enable_cancel=true, parallel_threads=2);
started_time       | 2026-07-14 17:09:54.977
elapsed_time       | 10269
phase_name         | Initialization
phase_started_time | 2026-07-14 17:09:54.977
phase_elapsed_time | 716
lock_level         | 8
lock_try_times     | 1
lock_hold_time     | 705
phase_detail       |
-[ RECORD 2 ]------+--------------------------------------------------------------------------
pid                | 139717555775232
node_name          | coordinator2
phase_id           | 2
rel_schema         | public
rel_name           | t1
sql_command        | VACUUM FULL t1 online with (enable_cancel=true, parallel_threads=2);
started_time       | 2026-07-14 17:09:54.977
elapsed_time       | 10269
phase_name         | Base data copy
phase_started_time | 2026-07-14 17:09:56.003
phase_elapsed_time | 2422
lock_level         | 4
lock_try_times     | 1
lock_hold_time     | 2479
phase_detail       | {
                   |         "total_pages":  4356,
                   |         "inserted_rows":        1000000,
                   |         "insert_speed": 447828.03403493058,
                   |         "parallel_threads":     2
                   | }
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-[ RECORD 3 ]------+--------------------------------------------------------------------------
pid                | 139717555775232
node_name          | coordinator2
phase_id           | 3
rel_schema         | public
rel_name           | t1
sql_command        | VACUUM FULL t1 online with (enable_cancel=true, parallel_threads=2);
started_time       | 2026-07-14 17:09:54.977
elapsed_time       | 10269
phase_name         | Reindexing
phase_started_time | 2026-07-14 17:09:58.428
phase_elapsed_time | 4656
lock_level         | 4
lock_try_times     | 1
lock_hold_time     | 4432
phase_detail       | {
                   |         "total_indexes":        2,
                   |         "indexes_rebuilt":      2,
                   |         "parallel_threads":     2,
                   |         "parallel_workers":     1
                   | }
-[ RECORD 4 ]------+--------------------------------------------------------------------------
pid                | 139717555775232
node_name          | coordinator2
phase_id           | 4
rel_schema         | public
rel_name           | t1
sql_command        | VACUUM FULL t1 online with (enable_cancel=true, parallel_threads=2);
started_time       | 2026-07-14 17:09:54.977
elapsed_time       | 10269
phase_name         | Catchup 1
phase_started_time | 2026-07-14 17:10:04.518
phase_elapsed_time | 275
lock_level         | 4
lock_try_times     | 1
lock_hold_time     | 23
phase_detail       | {
                   |         "short_lock_level":     7,
                   |         "short_lock_try_times": 1,
                   |         "short_lock_hold_time": 252,
                   |         "total_delete_pages":   0,
                   |         "total_insert_pages":   0,
                   |         "deleted_rows": 0,
                   |         "delete_speed": 0,
                   |         "inserted_rows":        0,
                   |         "insert_speed": 0,
                   |         "last_catchup_threshold":       5000,
                   |         "max_catchup_times":    50
                   | }
-[ RECORD 5 ]------+--------------------------------------------------------------------------
pid                | 139717555775232
node_name          | coordinator2
phase_id           | 5
rel_schema         | public
rel_name           | t1
sql_command        | VACUUM FULL t1 online with (enable_cancel=true, parallel_threads=2);
started_time       | 2026-07-14 17:09:54.977
elapsed_time       | 10269
phase_name         | Last catchup
phase_started_time | 2026-07-14 17:10:04.926
phase_elapsed_time | 299
lock_level         | 8
lock_try_times     | 1
lock_hold_time     | 74
phase_detail       | {
                   |         "total_delete_pages":   0,
                   |         "total_insert_pages":   0,
                   |         "deleted_rows": 0,
                   |         "delete_speed": 0,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
|       "inserted_rows":        0,
|       "insert_speed": 0,
|       "last_catchup_threshold":       5000,
|       "max_catchup_times":    50
| }
```

The meaning of each field is as follows.

Parameter	Description
pid	Process ID.
node_name	Name of the node where the statement is executed.
phase_id	Phase number, which increases in ascending order.
rel_schema	Schema of a table.
rel_name	Table name.
sql_command	Executed SQL statement.
started_time	Time when the SQL statement starts to be executed.
elapsed_time	End time of the query.
phase_name	Name of the online DDL execution phase, including **Initialization**, **Base data copy**, **Reindexing**, **Catchup** *<N>*, and **Last catchup**.
phase_started_time	Start time of the phase.
phase_elapsed_time	Actual execution time of the phase.
lock_level	Lock level.
lock_try_times	Number of attempts to obtain the lock. Value range: • Greater than or equal to 1: The lock has been obtained. • **0**: The lock has not been obtained, or the lock has been obtained but the progress information is not updated in real time.
lock_hold_time	Duration of holding the lock.
phase_detail	Detailed information about the current phase. The information displayed in each phase is in JSON format and may vary.

- The **phase_detail** information in the **Initialization** phase is empty.
- The **phase_detail** information in the **Base data copy** phase contains the following fields:

Parameter	Description
total_pages	Estimated number of pages to be inserted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
inserted_rows	Number of rows that have been inserted. This value is updated after the phase ends.
insert_speed	Actual insertion rate. This value is updated after the phase ends.
parallel_threads	Value of the **parallel_threads** parameter to be used or actually used in this phase. If a table contains an expression index, indexes cannot be built in parallel, and the value of **parallel_threads** is always **1**.

- The **phase_detail** column in the **Reindexing** phase contains the following fields:

Parameter	Description
total_indexes	Total number of indexes to be rebuilt.
indexes_rebuilt	Number of indexes that have been rebuilt.
parallel_threads	Value of **parallel_threads** to be used or actually used in this phase. If a table contains an expression index, indexes cannot be built in parallel, and the value of **parallel_threads** is always **1**.
parallel_workers	**parallel_workers** option of the source table. For details, see the description of **parallel_workers** in **ALTER TABLE**.

NOTE

During online DDL execution, two additional indexes are created for the generated temporary table. (The temporary table is located in the **online$$ddl$$[***Hash value of (Schema of the table + Table name)***]** schema. Therefore, the value of **total_index** is equal to the number of indexes created by users plus 2.

When indexes are built in parallel, the value of **indexes_rebuilt** may not be updated in real time.

- The **phase_detail** information in the **Catchup** *<N>* phase contains the following fields:

Parameter	Description
short_lock_level	Level of the strong lock that is held for a short period of time when the catchup phase starts.
short_lock_try_times	Number of retries to obtain a strong lock when the catchup phase starts.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
short_lock_hold_time	Duration of holding a strong lock for a short period of time when the catchup phase starts.
total_delete_pages	Estimated number of pages to be deleted.
total_insert_pages	Estimated number of pages to be inserted.
deleted_rows	Number of rows that are actually deleted. The value is updated after the phase ends.
delete_speed	Actual deletion speed. The value is updated after the phase ends.
inserted_rows	Number of rows that have been inserted. This value is updated after the phase ends.
insert_speed	Actual insertion speed. The value is updated after the phase ends.
last_catchup_threshold	Value of **last_catchup_threshold** to be used or actually used when this phase is executed.
max_catchup_times	Value of **max_catchup_times** to be used or actually used when this phase is executed.

- The **phase_detail** information in the **Last catchup** phase contains the following fields:

Parameter	Description
total_delete_pages	Estimated number of pages to be deleted.
total_insert_pages	Estimated number of pages to be inserted.
deleted_rows	Number of rows that are actually deleted. The value is updated after the phase ends.
delete_speed	Actual deletion speed. The value is updated after the phase ends.
inserted_rows	Number of rows that have been inserted. This value is updated after the phase ends.
insert_speed	Actual insertion speed. The value is updated after the phase ends.
last_catchup_threshold	Value of **last_catchup_threshold** to be used or actually used when this phase is executed.
max_catchup_times	Value of **max_catchup_times** to be used or actually used when this phase is executed.
total_gpis	Number of GPIs to be rebuilt. This parameter is displayed only when the DDL operation is partition-level VACUUM FULL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
gpis_rebuilt	Number of GPIs that have been rebuilt. In the current version, the value of this parameter is not updated in real time. It is updated to the value of **total_GPIs** only after all GPIs are rebuilt. This field is displayed only when the online VACUUM FULL operation is performed at the partition level.
parallel_workers	**parallel_workers** option of the source table. For details, see the description of **parallel_workers** in **ALTER TABLE**.

NOTE

In the index rebuild phase, the number of parallel workers is affected by both the online DDL parameter **parallel_threads** and the table size. However, in the last catchup process, the number of parallel workers is not affected by the online DDL parameter **parallel_threads**.

- gs_online_ddl_set_parameters(relschema, relname, params)

  Description: Dynamically modifies parameters during the execution of online DDL operations or online VACUUM FULL.

  Parameters:

  - **relschema**: schema name of the table. The type is cstring and the value cannot be empty.
  - **relname**: table name. The type is cstring and the value cannot be empty.
  - **params**: optional parameters. The type is variable-length text[]. It indicates the key-value pairs of the parameters to be modified. The format is 'key', 'value'. Multiple parameters can be updated at a time.
    - If **params** is specified, the parameters are modified based on the key-value pairs. A maximum of seven key-value pairs are supported. The key must be a parameter supported by the system, and the value must be a valid value within the supported range of the parameter.
    - If **params** is not specified, the query is executed and all parameter values supported by the function are returned.
    - If a parameter is entered for multiple times, the parameter value is not within the supported range, or the number of key-value pairs exceeds seven, an error is reported.

  NOTE

  - Parameter verification rule: Only the parameters in the whitelist and their valid values are processed. Otherwise, an error is reported.
  - Permission control rule: Only the system administrators, initial user, and users who have the VACUUM FULL permission can modify and query parameter information.

  Return type: multiple columns of text in one row. All parameter values of the online DDL operation being executed on the specified table are returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# CREATE TABLE t1(c1 int, c2 int);

-- Session 1: Perform online VACUUM FULL.
gaussdb=# VACUUM FULL t1 online;
-- Session 2:
gaussdb=# SELECT * from gs_online_ddl_set_parameters('public', 't1', 'last_catchup_threshold', '10000',
'max_catchup_times', '30','parallel_threads', '2','online_ddl_lockwait_timeout', '10', 'lockwait_interval',
'10', 'trylock_threshold', '1', 'enable_cancel', 'true');
 parallel_threads | last_catchup_threshold | max_catchup_times | online_ddl_lockwait_timeout |
lockwait_interval | trylock_threshold | enable_cancel
------------------+------------------------+-------------------+-----------------------------+-------------------
+-------------------+---------------
 2                | 10000                  | 30                | 10                          | 10                | 1                 | true
(1 row)
```

The supported parameters are as follows:

- **parallel_threads**: controls multi-thread parallel operations to achieve higher performance when resources are sufficient. The value is an integer ranging from 1 to 32. The default value is **1**.
  - When resource control is enabled (for example, **io_limits** set to a value greater than 0 or **io_priority** set to **high**, **medium**, or **low**), setting **parallel_threads** will forcibly enable multi-threading, ignoring the current user- or session-level resource control and displaying a message indicating invalid resource control. The online VACUUM FULL operation in multi-thread mode complies with the resource control of the initial user.
  - This parameter can be set together with the **parallel_workers** parameter that controls the degree of parallelism for index build during VACUUM FULL. The **parallel_threads** parameter controls the degree of parallelism for the catchup phase during VACUUM FULL.
- **max_catchup_times**: controls the maximum number of catchups. This parameter is used to prevent the execution of VACUUM FULL from taking a long time. The value is an integer ranging from 1 to *INT_MAX*. The default value is **50**.
  - Online VACUUM FULL execution records the IUD incremental data generated during data rebuild to a temporary table and rebuilds the incremental data in multiple rounds of catchups. To prevent VACUUM FULL catchup failures due to a large number of concurrent IUD operations, you can set this parameter to limit the number of catchups.
- **last_catchup_threshold**: specifies the upper limit of the time for triggering the last catchup. After a catchup round is complete, if the duration of this round is less than the threshold, the last round of catchup and switchover is triggered. Value range: [1000,1200000] (unit: ms). The default value is **5000**.
- **online_ddl_lockwait_timeout**: specifies the lock wait timeout interval for online DDL operations. If the thread waiting for the lock fails to obtain the lock within the specified time, a lock timeout error is reported. When the **CANCEL** mode is triggered to obtain a lock, the actual timeout interval is twice the value of this parameter. Value range: [1,1200000] (unit: ms). The default value is **10000**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - **lockwait_interval**: specifies the lock wait increment. That is, the retry interval when the lock wait times out consecutively. If the lock wait times out in the current round, the system waits for the period specified by this parameter and then attempts to obtain the lock in the next round. Value range: [0,300] (unit: seconds). The default value is **5**.
  - **trylock_threshold**: indicates the maximum number of lock wait attempts. It must be used together with **enable_cancel**. If the lock cannot be obtained due to lock wait timeout after the maximum number of attempts is reached, the **CANCEL** mode will be used for the next attempt. If **enable_cancel** is not enabled, the lock fails to be obtained and an error is reported after the maximum number of attempts is reached. Value range: [1,*INT_MAX*] or **-1** (indicating no upper limit). The default value is **-1**.
  - **enable_cancel**: specifies whether services can be canceled. That is, when the number of lock attempts reaches the upper limit or the level-8 lock of the user table is obtained during the table switching transaction, user services are quickly terminated to ensure that the online DDL operation can successfully obtain the lock. The value can be **false** or **true**. The default value is **false**.

- anynesttable_out(anynesttable)

  Description: Converts anynesttable into cstring.

  Parameter: **anynesttable**.

  Return type: cstring.

- nesttable_count(a nesttable)

  Description: Returns the number of elements in **nesttable**.

  Parameter: **nesttable**.

  Return type: int.

- nesttable_delete(a nesttable)

  Description: Deletes all elements from **nesttable**.

  Parameter: **nesttable**.

  Return type: nesttable.

- nesttable_delete(a nesttable, index int)

  Description: Deletes the element at the specified index from **nesttable** if the index falls within the range of **nesttable**.

  Parameters: **nesttable** and **int**.

  Return type: nesttable.

- nesttable_delete(a nesttable, lower int, upper int)

  Description: Deletes elements within the specified index range *[lower,upper]* from **nesttable** if the indexes from *lower* to *upper* fall within the range of **nesttable**.

  Parameters: **nesttable**, **int**, and **int**.

  Return type: nesttable.

- nesttable_eq(a nesttable, b nesttable)

  Description: Checks whether two **nesttable** parameters are equal.

  Parameters: **nesttable** and **nesttable**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: Boolean.

- nesttable_except(a nesttable, b nesttable)

  Description: Obtains the difference set of two **nesttable** parameters.

  Parameters: **nesttable** and **nesttable**.

  Return type: nesttable.

- nesttable_except_distinct(a nesttable, b nesttable)

  Description: Obtains the deduplicated difference set of two **nesttable** parameters.

  Parameters: **nesttable** and **nesttable**.

  Return type: nesttable.

- nesttable_exists(a nesttable, index int)

  Description: Checks whether an element exists at the specified index in **nesttable**.

  Parameters: **nesttable** and **int**.

  Return type: nesttable.

- nesttable_extend(a nesttable, cnt int)

  Description: Extends **nesttable** by appending *cnt* elements to it.

  Parameters: **nesttable** and **int**.

  Return type: nesttable.

- nesttable_extend(a nesttable, cnt int, index int)

  Description: Extends **nesttable** by appending *cnt* elements to it and assigning the element value at the specified index to these new elements.

  Parameters: **nesttable**, **int**, and **int**.

  Return type: nesttable.

- nesttable_first(a nesttable)

  Description: Returns the index of the first valid element in **nesttable**, or returns NULL if none is found.

  Parameter: **nesttable**.

  Return type: int.

- nesttable_in(str cstring, oid Oid, typmod int)

  Description: Converts a character string into nesttable with the specified OID.

  Parameters: **cstring**, **Oid**, and **int**.

  Return type: nesttable.

- nesttable_intersect(a nesttable, b nesttable)

  Description: Obtains the intersection of two **nesttable** parameters.

  Parameters: **nesttable** and **nesttable**.

  Return type: nesttable.

- nesttable_intersect_distinct(a nesttable, b nesttable)

  Description: Obtains the deduplicated intersection of two **nesttable** parameters.

  Parameters: **nesttable** and **nesttable**.

  Return type: nesttable.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- nesttable_last(a nesttable)

  Description: Returns the index of the last valid element in **nesttable**, or returns NULL if none is found.

  Parameter: **nesttable**.

  Return type: int.
- nesttable_ne(a nesttable, b nesttable)

  Description: Checks whether two **nesttable** parameters are unequal.

  Parameters: **nesttable** and **nesttable**.

  Return type: Boolean.
- nesttable_next(a nesttable, index int)

  Description: Returns the index of the next valid element following the specified index in **nesttable**, or returns NULL if none is found.

  Parameters: **nesttable** and **int**.

  Return type: int.
- nesttable_out(a nesttable)

  Description: Converts nesttable into cstring.

  Parameter: **nesttable**.

  Return type: cstring.
- nesttable_prior(a nesttable, index int)

  Description: Returns the index of the previous valid element before the specified index in **nesttable**, or returns NULL if none is found.

  Parameters: **nesttable** and **int**.

  Return type: int.
- nesttable_trim(a nesttable, cnt int)

  Description: Deletes *cnt* elements from the end of **nesttable** to free up storage space.

  Parameters: **nesttable** and **int**.

  Return type: int.
- nesttable_union(a nesttable, b nesttable)

  Description: Obtains the union of two **nesttable** parameters.

  Parameters: **nesttable** and **nesttable**.

  Return type: nesttable.
- nesttable_union_distinct(a nesttable, b nesttable)

  Description: Obtains the deduplicated union of two **nesttable** parameters.

  Parameters: **nesttable** and **nesttable**.

  Return type: nesttable.
- anyindexbytable_out(anyindexbytable)

  Description: Converts anyindexbytable into cstring.

  Parameter: **anyindexbytable**.

  Return type: cstring.
- indexbytable_count(a indexbytable)

  Description: Returns the number of elements in **indexbytable**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter: **indexbytable**.

Return type: int.

- indexbytable_in(str cstring, oid Oid)

  Description: Converts a character string into indexbytable with the specified OID.

  Parameters: **cstring** and **Oid**.

  Return type: indexbytable.

- indexbytable_out(a indexbytable)

  Description: Converts indexbytable into cstring.

  Parameter: **indexbytable**.

  Return type: cstring.

- indexbytableint_delete(a indexbytable)

  Description: Deletes all elements from **indexbytable**.

  Parameter: **indexbytable**.

  Return type: indexbytable.

- indexbytableint_delete(a indexbytable, index int)

  Description: Deletes the element at the specified index from **indexbytable** if the index falls within the range of **indexbytable**.

  Parameters: **indexbytable** and **int**.

  Return type: indexbytable.

- indexbytableint_delete(a indexbytable, lower int, upper int)

  Description: Deletes elements within the specified index range *[lower,upper]* from **indexbytable**.

  Parameters: **indexbytable**, **int**, and **int**.

  Return type: indexbytable.

- indexbytableint_exists(a indexbytable, index int)

  Description: Checks whether an element exists at the specified index in **indexbytable**.

  Parameters: **indexbytable** and **int**.

  Return type: indexbytable.

- indexbytableint_first(a indexbytable)

  Description: Returns the index of the first valid element in **indexbytable**, or returns NULL if none is found.

  Parameter: **indexbytable**.

  Return type: int.

- indexbytableint_last(a indexbytable)

  Description: Returns the index of the last valid element in **indexbytable**, or returns NULL if none is found.

  Parameter: **indexbytable**.

  Return type: int.

- indexbytableint_next(a indexbytable, index int)

  Description: Returns the index of the next valid element following the specified index in **indexbytable**, or returns NULL if none is found.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters: **indexbytable** and **int**.

Return type: int.

- indexbytableint_prior(a indexbytable, index int)

  Description: Returns the index of the previous valid element before the specified index in **indexbytable**, or returns NULL if none is found.

  Parameters: **indexbytable** and **int**.

  Return type: int.

- indexbytablevarchar_delete(a indexbytable)

  Description: Deletes all elements from **indexbytable**.

  Parameter: **indexbytable**.

  Return type: indexbytable.

- indexbytablevarchar_delete(a indexbytable, index varchar)

  Description: Deletes the element at the specified index from **indexbytable** if the index falls within the range of **indexbytable**.

  Parameters: **indexbytable** and **varchar**.

  Return type: indexbytable.

- indexbytablevarchar_delete(a indexbytable, lower varchar, upper varchar)

  Description: Deletes elements within the specified index range *[lower,upper]* from **indexbytable**.

  Parameters: **indexbytable**, **varchar**, and **varchar**.

  Return type: indexbytable.

- indexbytablevarchar_exists(a indexbytable, index varchar)

  Description: Checks whether an element exists at the specified index in **indexbytable**.

  Parameters: **indexbytable** and **varchar**.

  Return type: indexbytable.

- indexbytablevarchar_first(a indexbytable)

  Description: Returns the index of the first valid element in **indexbytable**, or returns NULL if none is found.

  Parameter: **indexbytable**.

  Return type: varchar.

- indexbytablevarchar_last(a indexbytable)

  Description: Returns the index of the last valid element in **indexbytable**, or returns NULL if none is found.

  Parameter: **indexbytable**.

  Return type: varchar.

- indexbytablevarchar_next(a indexbytable, index varchar)

  Description: Returns the index of the next valid element following the specified index in **indexbytable**, or returns NULL if none is found.

  Parameters: **indexbytable** and **varchar**.

  Return type: varchar.

- indexbytablevarchar_prior(a indexbytable, index varchar)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description: Returns the index of the previous valid element before the specified index in **indexbytable**, or returns NULL if none is found.

Parameters: **indexbytable** and **varchar**.

Return type: varchar.

- gs_partitioned_temp_fix_filenode()

  Description: Creates physical files for the partitioned tables and local indexes present on the source version during a version upgrade. If the creation is successful, the function will return **true**. If the physical files for the original objects (partitioned tables and local indexes on the source version) already exist, the function will return **false**. This function can be executed only during upgrades and by system administrators.

  Parameter: regclass

  Return type: Boolean

- double_backslash(input)

  Description: Double writes the backslash character in the LIKE matching logic of a single backslash character. The input can be of the text, bpchar, bytea, raw, or byteawithoutorderwithequalcol type.

  Parameters: text | bpchar | bytea | raw | byteawithoutorderwithequalcol

  Return type: same as the input parameter type.

1.6.61 Deprecated Functions

The following GaussDB functions have been deprecated in the current version:

- gs_wlm_get_session_info
- gs_wlm_get_user_session_info
- check_engine_status
- encode_plan_node
- model_train_opt
- gs_stat_get_wlm_plan_operator_info
- track_model_train_opt
- array_extend
- dbe_perf.global_slow_query_info
- dbe_perf.global_slow_query_info_bytime
- dbe_perf.global_slow_query_history
- pg_reload_conf
- pg_rotate_logfile
- gs_stat_ustore
- local_segment_space_info
- global_segment_space_info
- remote_segment_space_info
- pg_stat_segment_extent_usage
- pgxc_get_wlm_current_instance_info(text, int default null)
- pgxc_get_wlm_history_instance_info(text, TIMESTAMP, TIMESTAMP, int default null)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- GS_ALL_NODEGROUP_CONTROL_GROUP_INFO(text)
- create_wlm_operator_info(int flag)
- create_wlm_session_info(int flag)
- pg_stat_get_wlm_session_info(int flag)
- gs_wlm_get_resource_pool_info(int)
- gs_wlm_get_all_user_resource_info()
- gs_wlm_get_workload_records()
- gs_wlm_persistent_user_resource_info()
- gs_wlm_session_respool(bigint)
- pgxc_get_wlm_operator_history()
- pgxc_get_wlm_operator_info()
- pgxc_get_wlm_operator_statistics()
- pgxc_get_wlm_session_history()
- pgxc_get_wlm_session_info()
- pgxc_get_wlm_session_info_bytime(tag text, begin timestamp, end timestamp, limit int)
- pgxc_get_wlm_session_statistics()
- pgxc_wlm_get_workload_records()
- DBE_PERF.get_wlm_controlgroup_ng_config()
- DBE_PERF.get_global_wlm_workload_runtime()
- gs_wlm_user_resource_info(name text)
- create_wlm_instance_statistics_info()
- pg_stat_get_session_wlmstat()
- pg_stat_get_wlm_instance_info()
- pg_stat_get_wlm_instance_info_with_cleanup()
- pg_stat_get_wlm_node_resource_info()
- pg_stat_get_wlm_operator_info()
- pg_stat_get_wlm_realtime_operator_info()
- pg_stat_get_wlm_realtime_session_info()
- pg_stat_get_wlm_session_info_internal()
- pg_stat_get_wlm_statistics()
- gs_redis_set_bucketxid(bigint)
- gs_check_logic_cluster_consistency()
- gs_upload_obs_file('slot_name', 'src_file', 'dest_file')
- gs_download_obs_file('slot_name', 'src_file', 'dest_file')
- gs_get_obs_file_context('file_name', 'slot_name')
- gs_set_obs_file_context('file_name', 'file_context','slot_name')
- gs_get_hadr_key_cn()
- gs_hadr_has_barrier_creator()
- gs_hadr_do_switchover()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- track_memory_context(context_list text)
- track_memory_context_detail()
- gs_stat_remain_segment_info()
- gs_local_stat_remain_segment_info()
- gs_free_remain_segment()
- gs_local_free_remain_segment()
- copy_error_log_add_custom_id
- populate_record
- cupointer_bigint
- pg_autovac_coordinator
- mot_global_memory_detail
- mot_jit_detail
- mot_jit_profile
- mot_local_memory_detail
- mot_session_memory_detail
- gs_imcv_status()
- gs_imcv_flush(reloid oid)
- gs_htap_tmu_data(rel_oid oid, rowgroup_id integer)
- gs_htap_tmu_chunk_meta(rel_oid oid, rowgroup_id integer)
- gs_imcu_meta(rel_name)
- gs_htap_file_status(db_oid oid)
- gs_htap_scan_hit_status(reloid oid)
- gs_htap_common_debug(type integer, info text)
- gs_imcv_taskbuffer_view()
- gs_imcu_slot_status()
- prepare_statement_status()
- DBE_PERF.global_prepare_statement_status()
- pg_replication_origin_create()
- pg_replication_origin_drop()
- pg_replication_origin_oid()
- pg_replication_origin_session_setup()
- pg_replication_origin_session_reset()
- pg_replication_origin_session_is_setup()
- pg_replication_origin_session_progress()
- pg_replication_origin_xact_setup()
- pg_replication_origin_xact_reset()
- pg_replication_origin_advance()
- pg_replication_origin_progress()
- pg_show_replication_origin_status()

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.6.62 Vector Data Functions and Operators

1.6.62.1 Vector Distance Calculation APIs

l2_distance

Description: Calculates the Euclidean distance between two vectors.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: float8

Code example:

```
gaussdb=# SELECT l2_distance(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# SELECT l2_distance('[1,2,3]',  '[5,-1,3.5]');
```

vector_l2_squared_distance

Description: Calculates the square of the Euclidean distance between two vectors.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: float8

Code example:

```
gaussdb=# SELECT vector_l2_squared_distance(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# SELECT vector_l2_squared_distance('[1,2,3]',  '[5,-1,3.5]');
```

cosine_distance

Description: Calculates the cosine distance between two vectors.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: float8

Code example:

```
gaussdb=# select cosine_distance(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# select cosine_distance('[1,2,3]', '[5,-1,3.5]');
```

vector_spherical_distance

Description: Calculates the spherical distance between two normalized vectors (represented by the radian system of the cosine angle).

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: float8

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Code example:

```
gaussdb=#  select vector_spherical_distance('[1,0,0,0]', '[0,0,0,1]');
```

NOTE

If the input vector is not normalized, the correct calculation result cannot be obtained.

1.6.62.2 Vector Operation Functions

The vector operation functions include: vector size comparison, vector addition, vector subtraction, and vector bitwise multiplication.

inner_product

Description: Calculates the inner product of two vectors.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: float8

Code example:

```
gaussdb=# SELECT inner_product(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# SELECT inner_product('[1,2,3]', '[5,-1,3.5]');
```

vector_negative_inner_product

Description: Calculates the negative inner product of two vectors.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: float8

Code example:

```
gaussdb=# SELECT vector_negative_inner_product(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# SELECT vector_negative_inner_product('[1,2,3]', '[5,-1,3.5]');
```

vector_dims

Description: Returns the dimension value of the floatvector vector.

Input parameter type: floatvector

Output parameter type: int4

Code example:

```
gaussdb=# SELECT vector_dims(floatvector('[1,2,3]'));
gaussdb=# SELECT vector_dims('[1,2,3]');
```

vector_norm

Description: Returns the L2 norm of a vector.

Input parameter type: floatvector

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Output parameter type: float8

Code example:

```
gaussdb=# SELECT vector_norm(floatvector('[1,2,3]'));
gaussdb=# SELECT vector_norm('[1,2,3]');
```

vector_add

Description: Calculates the sum of two vectors.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: floatvector

Code example:

```
gaussdb=# SELECT vector_add(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# SELECT vector_add('[1,2,3]', '[5,-1,3.5]');
```

vector_sub

Description: Calculates the subtraction between two vectors.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: floatvector

Code example:

```
gaussdb=# SELECT vector_sub(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# SELECT vector_sub('[1,2,3]', '[5,-1,3.5]');
```

vector_lt

Description: Compares vector sizes and checks whether vector 1 is less than vector 2.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: Boolean

Code example:

```
gaussdb=# SELECT vector_lt(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# SELECT vector_lt('[1,2,3]', '[5,-1,3.5]');
```

vector_le

Description: Compares vector sizes and checks whether vector 1 is less than or equal to vector 2.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Code example:

```
gaussdb=# SELECT vector_le(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# SELECT vector_le('[1,2,3]', '[5,-1,3.5]');
```

vector_eq

Description: Compares vectors and checks whether they are equal.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: Boolean

Code example:

```
gaussdb=# SELECT vector_eq(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# SELECT vector_eq('[1,2,3]', '[5,-1,3.5]');
```

vector_ne

Description: Compares whether two vectors are not equal.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: Boolean

Code example:

```
gaussdb=# SELECT vector_ne(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# SELECT vector_ne('[1,2,3]', '[5,-1,3.5]');
```

vector_ge

Description: Compares vector sizes and checks whether vector 1 is greater than or equal to vector 2.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: Boolean

Code example:

```
gaussdb=# SELECT vector_ge(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# SELECT vector_ge('[1,2,3]', '[5,-1,3.5]');
```

vector_gt

Description: Compares vector sizes and checks whether vector 1 is greater than vector 2.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: Boolean

Code example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT vector_gt(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# SELECT vector_gt('[1,2,3]', '[5,-1,3.5]');
```

vector_cmp

Description: Compares vector sizes.

Type of input parameter 1: floatvector

Type of input parameter 2: floatvector

Output parameter type: int4

Code example:

```
gaussdb=# SELECT vector_cmp(floatvector('[1,2,3]'), floatvector('[5,-1,3.5]'));
gaussdb=# SELECT vector_cmp('[1,2,3]', '[5,-1,3.5]');
```

vector_accum

Description: Returns the vector accumulation.

Type of input parameter 1: anyarray

Type of input parameter 2: floatvector

Output parameter type: anyarray

Code example:

```
-- This is a system function and is not recommended. If this function is required, the array element type must be float8.
gaussdb=# SELECT vector_accum(array[cast(3 as float8),1,2,3],  floatvector('[5,-1,3.5]'));
```

vector_combine

Description: Combines vectors.

Type of input parameter 1: anyarray

Type of input parameter 2: anyarray

Output parameter type: anyarray

Code example:

```
-- This is a system function and is not recommended. If this function is required, the array element type must be float8.
gaussdb=# SELECT vector_combine(array[cast(1 as float8),2,3], array[cast(1 as float8),2,3]);
```

vector_avg

Description: Averages vectors.

Input parameter type: anyarray

Output parameter type: floatvector

Code example:

```
-- This is a system function and is not recommended. If this function is required, the array element type must be float8.
gaussdb=# SELECT vector_avg(array[cast(1 as float8),2,3]);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

bool_vector_dims

Description: Returns the dimension value of the boolvector vector.

Input parameter type: boolvector

Output parameter type: int4

Code example:

```
gaussdb=# SELECT bool_vector_dims(boolvector('[1,1,1]'));
```

bool_vector_cmp

Description: Compares the sizes of Boolean vectors.

Type of input parameter 1: boolvector

Type of input parameter 2: boolvector

Output parameter type: int4

Code example:

```
gaussdb=# SELECT bool_vector_cmp('[1,0,0]','[1,1,0]');
gaussdb=# SELECT bool_vector_cmp('[1,1,0]','[1,1,0]');
```

bool_vector_eq

Description: Compares Boolean vectors and checks whether they are equal.

Type of input parameter 1: boolvector

Type of input parameter 2: boolvector

Output parameter type: Boolean

Code example:

```
gaussdb=# SELECT bool_vector_eq(boolvector('[1,1,1]'), boolvector('[1,1,1]'));
gaussdb=# SELECT bool_vector_eq('[1,1,1]', '[1,1,1]');
```

1.6.62.3 Vector Functions and Operators

The floatvector supports data conversion between vector and array types, as well as conversion of strings in specific formats into the vector type.

array<->floatvector: The vector data type can be freely converted to the corresponding array type. The member data type of the floatvector vector is floating point.

string ->floatvector: You need to pay attention to the character string format. You need to contain arrays using square brackets ([]) or braces ({}) and separate elements using commas (,).

floatvector

Description:

Scenario 1: Convert array data into vector data.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Input parameter type: anyarray

Output parameter type: floatvector

Code example:

```
gaussdb=# SELECT floatvector(ARRAY[1,2,9.3]);
```

Scenario 2: Convert floatvector and check the dimensions.

Type of input parameter 1: floatvector

Type of input parameter 2: integer

Output parameter type: floatvector

Code example:

```
gaussdb=# SELECT floatvector(floatvector('[1,2,9.3]'),3);
```

vector_to_array

Description: Converts vector data to array data.

Input parameter type: floatvector

Output parameter type: real[]

Code example:

```
gaussdb=# SELECT vector_to_array(floatvector('[1,2,3]'));
```

text_to_vector

Description: Converts character data to vector data.

Input parameter type: cstring

Output parameter type: floatvector

Code example:

```
gaussdb=# SELECT text_to_vector('[1,2,9.3]');
```

vector_in

Description: Converts data in text format to the internal floatvector type.

Type of input parameter 1: cstring

Type of input parameter 2: OID

Type of input parameter 3: int4

Output parameter type: floatvector

Code example:

```
gaussdb=# SELECT vector_in('[1,2,3]',701,3);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

vector_out

Description: Converts vector data of the internal floatvector type to the text format.

Input parameter type: floatvector

Output parameter type: cstring

Code example:

```
gaussdb=# SELECT vector_out(floatvector('[1,2,3]'));
```

vector_send

Description: Converts vector data of the internal floatvector type to the bytea type.

Input parameter type: floatvector

Output parameter type: bytea

Code example:

```
gaussdb=# SELECT vector_send(floatvector('[1,2,3]'));
```

vector_recv

Description: Converts data in binary format to the internal floatvector type.

Type of input parameter 1: integer

Type of input parameter 2: OID

Type of input parameter 3: int4

Output parameter type: floatvector

Code example:

```
-- This is a system function and cannot be called by SQL statements.
```

vector_typmod_in

Description: Specifies the floatvector input type modifier function.

Input parameter type: cstring[]

Output parameter type: integer

Code example:

```
gaussdb=# SELECT vector_typmod_in( '{1}' );
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- Vector data type members support only single precision.
- Inter-vector calculation supports only the same dimension. If the dimensions are different, an error is reported.
- The floatvector supports vector addition and subtraction. The dot product operation is performed by the inner_product function.
- When creating a table, you must specify the dimension for the vector type. If the dimension of the inserted data is inconsistent with the specified dimension, the system reports an error.
- Note that if data has been inserted into the data table, the data in the table must comply with the dimension setting when the vector dimension is changed. Otherwise, the database reports an error. If a vector index has been created for a vector attribute, you cannot change the data type of the attribute. Otherwise, the database reports an error.

boolvector

Scenario 1:

Description: Converts array data to vector data.

Input parameter type: anyarray

Output parameter type: boolvector

Code example:

```
gaussdb=# SELECT boolvector(ARRAY[1,0,1]);
```

Scenario 2:

Description: Converts data to boolvector and detects dimensions.

Type of input parameter 1: boolvector

Type of input parameter 2: integer

Output parameter type: boolvector

Code example:

```
gaussdb=# SELECT boolvector(boolvector('[1,0,1]'),3);
```

boolvector_to_array

Description: Converts vector data to array data.

Input parameter type: boolvector

Output parameter type: anyarray

Code example:

```
gaussdb=# SELECT boolvector_to_array(boolvector('[1,0,1]'));
```

text_to_boolvector

Description: Converts character data to vector data.

Input parameter type: cstring

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Output parameter type: boolvector

Code example:

```
gaussdb=# SELECT text_to_boolvector('[1,1,1]');
```

bool_vector_in

Description: Converts data in text format to the internal boolvector type.

Type of input parameter 1: cstring

Type of input parameter 2: OID

Type of input parameter 3: int4

Output parameter type: boolvector

Code example:

```
gaussdb=# SELECT bool_vector_in('[1,1,1]',701,3);
```

bool_vector_out

Description: Converts vector data of the internal boolvector type to the text format.

Input parameter type: boolvector

Output parameter type: cstring

Code example:

```
gaussdb=# SELECT bool_vector_out(boolvector('[1,1,1]'));
```

bool_vector_send

Description: Converts vector data of the internal boolvector type to the bytea type.

Input parameter type: boolvector

Output parameter type: bytea

Code example:

```
gaussdb=# SELECT bool_vector_send(boolvector('[1,1,1]'));
```

bool_vector_recv

Description: Converts data in binary format to the internal boolvector type.

Type of input parameter 1: internal

Type of input parameter 2: OID

Type of input parameter 3: int4

Output parameter type: boolvector

Code example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- This is a system function and cannot be called by SQL statements.
```

bool_vector_typmod_in

Description: Specifies the boolvector input type modifier function.

Input parameter type: cstring[]

Output parameter type: integer

Code example:

```
gaussdb=# SELECT bool_vector_typmod_in( '{1}' );
```

NOTE

- The vector data type does not support NULL, NaN, or Inf as elements. If a vector contains NULL values, the database reports an error.
- When a vector data type is inserted, NULL cannot be used as the inserted value. If NULL is inserted as vector data, the database reports an error.
- Elements of the boolvector type can express Boolean data using **t(T)**/**f(F)**, **y(Y)**/**n(N)**, **1**/**0**, **true**/**false**, **yes**/**no**, or **on**/**off**.
- The boolvector type supports only the equal-to operation and does not support comparison.

gs_vector_index_options

Description: Displays the hyperparameter values of related vector indexes.

Input parameter type: text

Output parameter type: text

Code example:

```
-- Create a table.
gaussdb=# CREATE TABLE t1 (id int unique,repr floatvector(960));
Insert data.
gaussdb=# CREATE OR REPLACE FUNCTION float_random_array(dims int,range int) RETURNS float[] AS $
$BEGIN RETURN ARRAY(SELECT (random() * range)::int FROM generate_series(1, dims));END;$$
LANGUAGE plpgsql;
gaussdb=# INSERT INTO t1 SELECT i,floatvector(float_random_array(960,100)) FROM
generate_series(1,1000) as i;
Create an index.
gaussdb=# CREATE INDEX test1v on t1 using gsdiskann (repr l2) with
(pq_nseg=120,pq_nclus=64,queue_size=120,num_parallels=30,enable_pq=true,using_clustering_for_parallel=f
alse);

gaussdb=# SELECT gs_vector_index_options('test1v');
```

gs_diskann_inspect

Description: Queries the index health status, including the index disk usage, PQ compression accuracy, index graph connection health, and vacuum status.

Scenario 1:

Input parameter type: text

Output parameter type: record

Code example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a table.
gaussdb=# CREATE TABLE t1 (id int unique,repr floatvector(960));
-- Insert data.
gaussdb=# CREATE OR REPLACE FUNCTION float_random_array(dims int,range int) RETURNS float[] AS $
$BEGIN RETURN ARRAY(SELECT (random() * range)::int FROM generate_series(1, dims));END;$$
LANGUAGE plpgsql;
gaussdb=# INSERT INTO t1 SELECT i,floatvector(float_random_array(960,100)) FROM
generate_series(1,1000) as i;
-- Create a vector index.
gaussdb=# CREATE INDEX test1v on t1 using gsdiskann (repr l2) with
(pq_nseg=120,pq_nclus=64,queue_size=120,num_parallels=30,enable_pq=true,using_clustering_for_parallel=f
alse);
-- Display the index status on a specified DN.
gaussdb=# EXECUTE DIRECT ON (datanode1) 'SELECT * FROM gs_diskann_inspect(''test1v'');';
-- Display the index status on all DNs.
gaussdb=# SELECT * FROM gs_diskann_inspect('test1v');
```

Scenario 2:

Type of input parameter 1: text

Type of input parameter 2: text

Output parameter type: record

Code example:

```
-- Create a partitioned table.
gaussdb=# CREATE TABLE t1 (id int unique,repr floatvector(960)) partition by hash(id) (partition p1,
partition p2, partition p3);
-- Insert data.
gaussdb=# CREATE OR REPLACE FUNCTION float_random_array(dims int,range int) RETURNS float[] AS $
$BEGIN RETURN ARRAY(SELECT (random() * range)::int FROM generate_series(1, dims));END;$$
LANGUAGE plpgsql;
gaussdb=# INSERT INTO t1 SELECT i,floatvector(float_random_array(960,100)) FROM
generate_series(1,1000) as i;
-- Create a partitioned vector index.
gaussdb=# CREATE INDEX test1v on t1 using gsdiskann (repr l2) local with
(pq_nseg=120,pq_nclus=64,queue_size=120,num_parallels=30,enable_pq=true,using_clustering_for_parallel=f
alse);
-- Query a partition name and a partitioned index name.
SELECT oid,relname,parttype,parentid,boundaries FROM pg_partition WHERE parentid = 't1'::regclass::oid;
SELECT oid,relname,parttype,parentid,boundaries,indextblid FROM pg_partition WHERE parentid =
'test1v'::regclass::oid;
-- Display the index status of a specified partition on a specified DN.
gaussdb=# EXECUTE DIRECT ON (datanode1) 'SELECT * FROM gs_diskann_inspect(''test1v'', ''p1_repr_idx'');';
-- Display the index status of a specified partition on all DNs.
gaussdb=# SELECT * FROM gs_diskann_inspect('test1v', 'p1_repr_idx');
```

Scenario 3:

Type of input parameter 1: text

Type of input parameter 2: text

Type of input parameter 3: text

Output parameter type: record

Code example:

```
-- Create a partitioned table.
gaussdb=# CREATE TABLE t1 (id int unique,repr floatvector(960)) partition by hash(id) (partition p1,
partition p2, partition p3);
-- Insert data.
gaussdb=# CREATE OR REPLACE FUNCTION float_random_array(dims int,range int) RETURNS float[] AS $
$BEGIN RETURN ARRAY(SELECT (random() * range)::int FROM generate_series(1, dims));END;$$
LANGUAGE plpgsql;
gaussdb=# INSERT INTO t1 SELECT i,floatvector(float_random_array(960,100)) FROM
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
generate_series(1,1000) as i;
-- Create a partitioned vector index.
gaussdb=# CREATE INDEX test1v on t1 using gsdiskann (repr l2) local with
(pq_nseg=120,pq_nclus=64,queue_size=120,num_parallels=30,enable_pq=true,using_clustering_for_parallel=f
alse);
-- Query a partition name, partitioned index name, and DN name.
SELECT oid,relname,parttype,parentid,boundaries FROM pg_partition WHERE parentid = 't1'::regclass::oid;
SELECT oid,relname,parttype,parentid,boundaries,indextblid FROM pg_partition WHERE parentid =
'test1v'::regclass::oid;
SELECT oid,node_name, nodeis_preferred FROM pgxc_node WHERE node_type = 'D' ORDER BY 1;
-- Display the index status of a specified partition on a specified DN.
gaussdb=# SELECT * FROM gs_diskann_inspect('test1v', 'p1_repr_idx', 'datanode1');
```

gs_ivfflat_inspect

Description: Queries the GsIvfflat index status, including the fixed page number, index disk usage, and vector data distribution in the cluster. You need to connect to a DN or use the EXECUTE DIRECT ON syntax.

Scenario 1:

Input parameter type: text

Output parameter type: record

Code example:

```
-- Create a table.
gaussdb=# CREATE TABLE t1 (id int unique,repr floatvector(960));
-- Insert data.
gaussdb=# CREATE OR REPLACE FUNCTION float_random_array(dims int,range int) RETURNS float[] AS $
$BEGIN RETURN ARRAY(SELECT (random() * range)::int FROM generate_series(1, dims));END;$$
LANGUAGE plpgsql;
gaussdb=# INSERT INTO t1 SELECT i,floatvector(float_random_array(960,100)) FROM
generate_series(1,1000) as i;
-- Create a vector index.
gaussdb=# SET maintenance_work_mem='128MB';
gaussdb=# CREATE INDEX test1v on t1 using gsivfflat (repr l2) with (ivf_nlist=10);
-- Display the index status on a specified DN.
gaussdb=# EXECUTE DIRECT ON (datanode1) 'SELECT * FROM gs_ivfflat_inspect(''test1v'');';
```

Scenario 2:

Type of input parameter 1: text

Type of input parameter 2: text

Output parameter type: record

Code example:

```
-- Create a partitioned table.
gaussdb=# CREATE TABLE t1 (id int unique,repr floatvector(960)) partition by hash(id) (partition p1,
partition p2, partition p3);
-- Insert data.
gaussdb=# CREATE OR REPLACE FUNCTION float_random_array(dims int,range int) RETURNS float[] AS $
$BEGIN RETURN ARRAY(SELECT (random() * range)::int FROM generate_series(1, dims));END;$$
LANGUAGE plpgsql;
gaussdb=# INSERT INTO t1 SELECT i,floatvector(float_random_array(960,100)) FROM
generate_series(1,1000) as i;
-- Create a partitioned vector index.
gaussdb=# SET maintenance_work_mem='128MB';
gaussdb=# CREATE INDEX test1v on t1 using gsivfflat (repr l2) LOCAL with (ivf_nlist=10);
-- Query a partition name and a partitioned index name.
gaussdb=# SELECT oid,relname,parttype,parentid,boundaries FROM pg_partition WHERE parentid =
't1'::regclass::oid;
gaussdb=# SELECT oid,relname,parttype,parentid,boundaries,indextblid FROM pg_partition WHERE parentid
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
= 'test1v'::regclass::oid;
-- Display the index status of a specified partition on a specified DN.
gaussdb=# EXECUTE DIRECT ON (datanode1) 'SELECT * FROM gs_ivfflat_inspect(''test1v'', ''p1_repr_idx'');';
```

Operators for vector calculation focus on measuring similarity between vectors and binary operators of vectors.

<->

Description: Calculates the Euclidean distance (L2) between two floatvectors.

Left parameter type: floatvector

Right parameter type: floatvector

Return type: double precision

Code example:

```
gaussdb=# SELECT floatvector('[1,1,3,2]') <->  floatvector('[1,1,3,2]');
gaussdb=# SELECT '[1,2,3,2]'<-> floatvector('[1,1,3,2]');
```

<+>

Description: Calculates the cosine distance between two floatvectors.

Left parameter type: floatvector

Right parameter type: floatvector

Return type: double precision

Code example:

```
gaussdb=# SELECT floatvector('[1,1,3,2]') <+>  floatvector('[1,1,3,2]');
gaussdb=# SELECT '[1,2,3,2]'<+> floatvector('[1,1,3,2]');
```

<#>

Description: Calculates the Hamming distance between two boolvectors.

Left parameter type: boolvector

Right parameter type: boolvector

Return type: double precision

Code example:

```
gaussdb=# SELECT boolvector('[1,0,1,0]') <#>  boolvector('[1,1,1,0]');
gaussdb=# SELECT '[1,0,1,0]'<#> boolvector('[1,1,1,0]');
```

+

Description: Calculates the bitwise addition of two vectors with the same dimension.

Left parameter type: floatvector

Right parameter type: floatvector

Return type: floatvector

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Code example:

```
gaussdb=# SELECT floatvector('[1,1,3,2]') + floatvector('[1,1,3,2]');
gaussdb=# SELECT '[1,2,3,2]'+ floatvector('[1,1,3,2]');
```

-

Description: Calculates the bitwise subtraction of two vectors with the same dimension.

Left parameter type: floatvector

Right parameter type: floatvector

Return type: floatvector

Code example:

```
gaussdb=# SELECT floatvector('[1,1,3,2]') - floatvector('[1,1,3,2]');
gaussdb=# SELECT '[1,2,3,2]'- floatvector('[1,1,3,2]');
```

<

Description: Compares the lexicographical order of two vectors with the same dimension.

Left parameter type: floatvector

Right parameter type: floatvector

Return type: Boolean

Code example:

```
gaussdb=# SELECT floatvector('[1,1,3,2]') < floatvector('[1,1,3,2]');
gaussdb=# SELECT '[1,2,3,2]'< floatvector('[1,1,3,2]');
```

<=

Description: Compares the lexicographical order of two vectors with the same dimension.

Left parameter type: floatvector

Right parameter type: floatvector

Return type: Boolean

Code example:

```
gaussdb=# SELECT floatvector('[1,1,3,2]') <= floatvector('[1,1,3,2]');
gaussdb=# SELECT '[1,2,3,2]'<= floatvector('[1,1,3,2]');
```

>

Description: Compares the lexicographical order of two vectors with the same dimension.

Left parameter type: floatvector

Right parameter type: floatvector

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: Boolean

Code example:

```
gaussdb=# SELECT floatvector('[1,1,3,2]') > floatvector('[1,1,3,2]');
gaussdb=# SELECT '[1,2,3,2]'> floatvector('[1,1,3,2]');
```

>=

Description: Compares the lexicographical order of two vectors with the same dimension.

Left parameter type: floatvector

Right parameter type: floatvector

Return type: Boolean

Code example:

```
gaussdb=# SELECT floatvector('[1,1,3,2]') >= floatvector('[1,1,3,2]');
gaussdb=# SELECT '[1,2,3,2]'>= floatvector('[1,1,3,2]');
```

=

Scenario 1:

Description: Determines whether two vectors with the same dimension are equal.

Left parameter type: floatvector

Right parameter type: floatvector

Return type: Boolean

Code example:

```
gaussdb=# SELECT floatvector('[1,1,3,2]') = floatvector('[1,1,3,2]');
gaussdb=# SELECT '[1,2,3,2]'= floatvector('[1,1,3,2]');
```

Scenario 2:

Description: Determines whether two boolvectors with the same dimension are consistent.

Left parameter type: boolvector

Right parameter type: boolvector

Return type: Boolean

Code example:

```
gaussdb=# SELECT boolvector('[1,0,1,0]') =  boolvector('[1,1,1,0]');
gaussdb=# SELECT '[1,0,1,0]' =  boolvector('[1,1,1,0]');
```

<>

Description: Determines whether two vectors with the same dimension are not equal.

Left parameter type: floatvector

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Right parameter type: floatvector

Return type: Boolean

Code example:

```
gaussdb=# SELECT floatvector('[1,1,3,2]') <> floatvector('[1,1,3,2]');
gaussdb=# SELECT '[1,2,3,2]'<> floatvector('[1,1,3,2]');
```

NOTE

If the vector element value is too large, the intermediate calculation result may overflow the single-precision range during distance calculation. As a result, the distance result is NaN or INF.

1.6.62.4 Functions and Operators for Similar Document Retrieval and Ranking

###

Scenario 1:

Description: Calculates the similarity between two texts based on the BM25 algorithm family. This is valid only for queries that use the BM25 index.

Left parameter type: text

Right parameter type: text

Return type: double precision

Note: This function is unavailable in the current version.

Scenario 2:

Description: Calculates the similarity between two tokenized text arrays based on the BM25 algorithm family. This is valid only for queries that use the BM25 index.

Left parameter type: text[]

Right parameter type: text[]

Return type: double precision

Note: This function is unavailable in the current version.

gs_ts_add_definition

Description: Incrementally adds dictionary content.

Input parameter type: regdictionary, "char", text[]

Output parameter type: Boolean

Note: This function is unavailable in the current version.

gs_bm25_tokenize

Description: Uses a specified dictionary to tokenize the input text and returns tokenized text arrays.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Input parameter type: text, cstring (default value: **"pg_catalog.simple_cn_segmentation"**)

Output parameter type: text[]

Note: This function is unavailable in the current version.

gs_bm25_score

Description: Returns the similarity score of the BM25 document. This parameter is valid only when the BM25 index is used for retrieval.

Input parameter type: ctid

Output parameter type: double precision

Note: This function is unavailable in the current version.

gs_bm25_distance_text

Description: Returns the similarity score of the BM25 document. This parameter is valid only when the BM25 index is used for retrieval.

Input parameter type: text, text

Output parameter type: double precision

Note: This function is unavailable in the current version.

gs_bm25_distance_textarr

Description: Returns the similarity score of the BM25 document. This parameter is valid only when the BM25 index is used for retrieval.

Input parameter type: text[], text[]

Output parameter type: double precision

Note: This function is unavailable in the current version.

gs_bm25_docid_info

Description: Returns the **docid** information corresponding to a specified CTID in a specified BM25 index.

Input parameter type: oid, uint4, uint2

Output parameter type: record

Note: This function is unavailable in the current version.

gs_bm25_document_info

Description: Returns the forward index information corresponding to a specified **docid** in a specified BM25 index.

Input parameter type: oid, uint4

Output parameter type: record

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Note: This function is unavailable in the current version.

gs_bm25_tokenhash_info

Description: Returns the hash container information of a specified BM25 index.

Input parameter type: OID

Output parameter type: record

Note: This function is unavailable in the current version.

gs_bm25_token_info

Description: Returns the token information of a specified document in a specified BM25 index.

Input parameter type: OID, text

Output parameter type: record

Note: This function is unavailable in the current version.

gs_bm25_posting_info

Description: Returns the inverted index information corresponding to a specified **tokenid** in a specified BM25 index.

Input parameter type: OID, text

Output parameter type: record

Note: This function is unavailable in the current version.

1.7 Expressions

1.7.1 Simple Expressions

Logical Expressions

Logical Operators lists the operators and calculation rules of logical expressions.

Comparative Expressions

Comparison Operators lists the common comparative operators.

In addition to comparative operators, you can also use the following sentence structure:

- BETWEEN operators:
  a BETWEEN x AND y is equivalent to **a >= x AND a <= y**.
  a NOT BETWEEN x AND y is equivalent to **a < x OR a > y**.
- To check whether a value is null, use:
  expression IS NULL

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

expression IS NOT NULL

or an equivalent (non-standard) sentence structure:

expression ISNULL

expression NOTNULL

NOTICE

- Do not write **expression=NULL** or **expression<>(!=)NULL** because **NULL** represents an unknown value, and these expressions cannot determine whether two unknown values are equal.
- Only the comparative expressions IS NULL and IS NOT NULL support data of XML type.

- is distinct from/is not distinct from
  - is distinct from

    If the data types and values of A and B are different, the value is **true**.

    If the data types and values of A and B are the same, the value is **false**.

    Empty values are considered the same.
  - is not distinct from

    If the data types and values of A and B are different, the value is **false**.

    If the data types and values of A and B are the same, the value is **true**.

    Empty values are considered the same.
- <=> NULL-safe equal operator

  The comparison of NULL values is added on the basis of the comparison of '='. If neither the left nor right value of the operator is NULL, the result is the same as that of '='.

  If the data types and values of A and B are different, the value is **false**.

  If the data types and values of A and B are the same, the value is **true**.

  Empty values are considered the same.

  NOTE

  - The usage of the <=> operator is the same as that of IS NOT DISTINCT FROM.
  - This operator is valid only when the database is compatible with the MySQL type (that is, **sql_compatibility** is set to **'MYSQL'**). Other types do not support this operator.

Examples

```
gaussdb=# SELECT 2 BETWEEN 1 AND 3 AS RESULT;
 result
----------
 t
(1 row)

gaussdb=# SELECT 2 >= 1 AND 2 <= 3 AS RESULT;
 result
----------
 t
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT 2 NOT BETWEEN 1 AND 3 AS RESULT;
 result
----------
 f
(1 row)

gaussdb=# SELECT 2 < 1 OR 2 > 3 AS RESULT;
 result
----------
 f
(1 row)

gaussdb=# SELECT 2+2 IS NULL AS RESULT;
 result
----------
 f
(1 row)

gaussdb=# SELECT 2+2 IS NOT NULL AS RESULT;
 result
----------
 t
(1 row)

gaussdb=# SELECT 2+2 ISNULL AS RESULT;
 result
----------
 f
(1 row)

gaussdb=# SELECT 2+2 NOTNULL AS RESULT;
 result
----------
 t
(1 row)

gaussdb=# SELECT 2+2 IS DISTINCT FROM NULL AS RESULT;
 result
----------
 t
(1 row)

gaussdb=# SELECT 2+2 IS NOT DISTINCT FROM NULL AS RESULT;
 result
----------
 f
(1 row)

gaussdb=# SELECT 1 <=> 1 AS RESULT;
 result
--------
 t
(1 row)

gaussdb=# SELECT NULL <=> 1 AS RESULT;
 result
--------
 f
(1 row)

gaussdb=# SELECT NULL <=> NULL AS RESULT;
 result
--------
 t
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.7.2 Condition Expressions

Data that meets the requirements specified by conditional expressions are filtered during SQL statement execution.

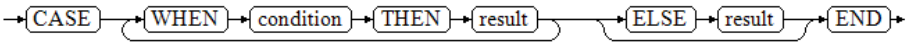
Conditional expressions include the following types:

- CASE

  CASE expressions are similar to the **CASE** statements in other coding languages.

  Figure 1-3 shows the syntax of a CASE expression.

  Figure 1-3 case::=

  A CASE clause can be used in a valid expression. The condition is an expression that returns a value of Boolean type.

  - If the result is **true**, the result of the CASE expression is the required result.
  - If the result is false, the following WHEN or ELSE clauses are processed in the same way.
  - If every WHEN condition is not true, the result of the expression is the result of the ELSE clause. If the ELSE clause is omitted and has no match condition, the result is NULL.
  - Operations on XML data are supported.
  - When the GUC parameter **enable_case_intervaltonumeric** is set to **true**, INTERVAL can be converted to NUMERIC without reporting an error.

  Example:

```
gaussdb=# CREATE TABLE case_when_t1(CW_COL1 INT) DISTRIBUTE BY HASH (CW_COL1);

gaussdb=# INSERT INTO case_when_t1 VALUES (1), (2), (3);

gaussdb=# SELECT * FROM case_when_t1;
 cw_col1
---
 1
 2
 3
(3 rows)

gaussdb=# SELECT CW_COL1, CASE WHEN CW_COL1=1 THEN 'one' WHEN CW_COL1=2 THEN 'two'
ELSE 'other' END FROM case_when_t1 ORDER BY 1;
 cw_col1 | case
---------+-------
       1 | one
       2 | two
       3 | other
(3 rows)

gaussdb=# DROP TABLE case_when_t1;

-- The default value of the GUC parameter is false. In this case, an error is reported.
gaussdb=# SHOW enable_case_intervaltonumeric;
 enable_case_intervaltonumeric
-------------------------------
 off
```

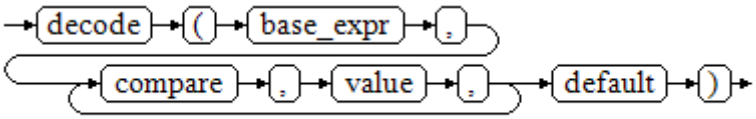
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)

gaussdb=# SELECT CASE
gaussdb-#     WHEN 1=1 THEN
gaussdb-#         to_date('20240118','yyyymmdd')-to_date('20240116','yyyymmdd')
gaussdb-#     ELSE
gaussdb-#         1
gaussdb-# end;
ERROR:  CASE types integer and interval cannot be matched
LINE 3:         to_date('20240118','yyyymmdd')-to_date('20240116','y...
                ^
CONTEXT:  referenced column: case

-- Enable the GUC parameter. The result can be correctly returned.
gaussdb=# SET enable_case_intervaltonumeric=true;
SET
gaussdb=# SELECT CASE
gaussdb-#     WHEN 1=1 THEN
gaussdb-#         to_date('20240118','yyyymmdd')-to_date('20240116','yyyymmdd')
gaussdb-#     ELSE
gaussdb-#         1::int1
gaussdb-# END;
 case
------
    2
(1 row)
```

- DECODE

  Figure 1-4 shows the syntax of DECODE.

  Figure 1-4 decode::=

  Compare each following compare(n) with base_expr, **value(n)** is returned if a compare(n) matches the base_expr expression. If base_expr does not match each compare(n), the default value is returned.

  Operations on XML data are supported.

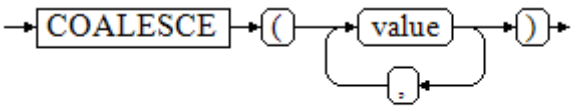
  Conditional Expression Functions describes the examples.

```
gaussdb=# SELECT DECODE('A','A',1,'B',2,0);
 case
------
    1
(1 row)
```

- COALESCE

  Figure 1-5 shows the syntax of COALESCE.

  Figure 1-5 coalesce::=

  COALESCE returns its first not-NULL value. If all the parameters are **NULL**, **NULL** is returned. This value is replaced by the default value when data is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

displayed. Like a CASE expression, COALESCE only evaluates the parameters that are needed to determine the result. That is, parameters to the right of the first not-NULL parameter are not evaluated.

Operations on XML data are supported.

Example:

```
gaussdb=# CREATE TABLE c_tabl(description varchar(10), short_description varchar(10), last_value
varchar(10))
DISTRIBUTE BY HASH (last_value);

gaussdb=# INSERT INTO c_tabl VALUES('abc', 'efg', '123');
gaussdb=# INSERT INTO c_tabl VALUES(NULL, 'efg', '123');

gaussdb=# INSERT INTO c_tabl VALUES(NULL, NULL, '123');

gaussdb=# SELECT description, short_description, last_value, COALESCE(description, short_description,
last_value) FROM c_tabl ORDER BY 1, 2, 3, 4;
 description | short_description | last_value | coalesce
-------------+-------------------+------------+----------
 abc         | efg               | 123        | abc
             | efg               | 123        | efg
             |                   | 123        | 123
(3 rows)

gaussdb=# DROP TABLE c_tabl;
```

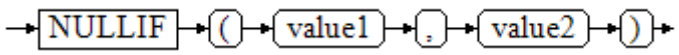
If **description** is not **NULL**, the value of **description** is returned. Otherwise, **short_description** is calculated. If **short_description** is not **NULL**, the value of **short_description** is returned. Otherwise, **last_value** is calculated. If **last_value** is not **NULL**, the value of **last_value** is returned. Otherwise, **none** is returned.

```
gaussdb=# SELECT COALESCE(NULL,'Hello World');
   coalesce
---------------
 Hello World
(1 row)
```

- NULLIF

  Figure 1-6 shows the syntax of NULLIF.

  Figure 1-6 nullif::=

  Only if **value1** is equal to **value2** can **NULLIF** return the **NULL** value. Otherwise, **value1** is returned. Operations on XML data are supported.

  Example:

```
gaussdb=# CREATE TABLE null_if_t1 (
    NI_VALUE1 VARCHAR(10),
    NI_VALUE2 VARCHAR(10)
)DISTRIBUTE BY HASH (NI_VALUE1);

gaussdb=# INSERT INTO null_if_t1 VALUES('abc', 'abc');
gaussdb=# INSERT INTO null_if_t1 VALUES('abc', 'efg');

gaussdb=# SELECT NI_VALUE1, NI_VALUE2, NULLIF(NI_VALUE1, NI_VALUE2) FROM null_if_t1 ORDER
BY 1, 2, 3;

 ni_value1 | ni_value2 | nullif
-----------+-----------+--------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 abc      | abc      |
 abc      | efg      | abc
(2 rows)
gaussdb=# DROP TABLE null_if_t1;
```

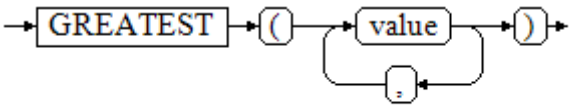
If **value1** is equal to **value2**, **NULL** is returned. Otherwise, **value1** is returned.

```
gaussdb=# SELECT NULLIF('Hello','Hello World');
 nullif
--------
 Hello
(1 row)
```

- GREATEST (maximum value) and LEAST (minimum value)

  Figure 1-7 shows the syntax of GREATEST.

  Figure 1-7 greatest::=

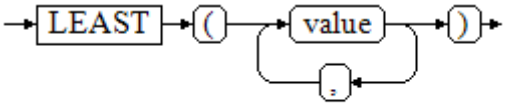
  

  You can select the maximum value from any numerical expression list. Operations on XML data are supported.

```
gaussdb=# SELECT greatest(9000,155555,2.01);
 greatest
----------
   155555
(1 row)
```

  Figure 1-8 shows the syntax of LEAST.

  Figure 1-8 least::=

  You can select the minimum value from any numerical expression list.

  Each of the preceding numeric expressions can be converted into a common data type, which will be the data type of the result.

  The NULL values in the list will be ignored. The result is **NULL** only if the results of all expressions are **NULL**.

  Operations on XML data are supported.

```
gaussdb=# SELECT least(9000,2);
 least
-------
     2
(1 row)
```

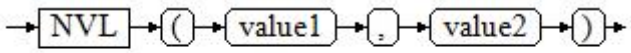
  Conditional Expression Functions describes the examples.

- NVL

  Figure 1-9 shows the syntax of NVL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 1-9 nvl::=



If the value of **value1** is **NULL**, **value2** is returned. Otherwise, **value1** is returned. Operations on XML data are supported.

Example:

```
gaussdb=# SELECT nvl(null,1);
nvl
-----
 1
(1 row)

gaussdb=# SELECT nvl('Hello World',1);
      nvl
---------------
 Hello World
(1 row)
```

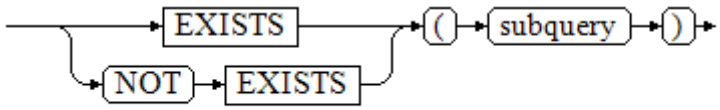
1.7.3 Subquery Expressions

Subquery expressions include the following types:

- EXISTS/NOT EXISTS

  Figure 1-10 shows the syntax of EXISTS/NOT EXISTS.

  Figure 1-10 EXISTS/NOT EXISTS::=

  The parameter of an EXISTS expression is an arbitrary SELECT statement, that is, subquery. The subquery is evaluated to determine whether it returns any rows. If it returns at least one row, the result of EXISTS is true. If the subquery returns no rows, the result of EXISTS is false.

  The subquery will generally only be executed long enough to determine whether at least one row is returned, not all the way to completion.

  Operations on XML data are not supported.

  Example:

```
gaussdb=# CREATE TABLE exists_t1(a int, b int);
gaussdb=# INSERT INTO exists_t1 VALUES(1, 2),(2, 3),(3, 4),(4, 5);

gaussdb=# CREATE TABLE exists_t2(a int, c int);
gaussdb=# INSERT INTO exists_t2 VALUES(3, 4),(4, 5),(5, 6),(6, 7);

gaussdb=# SELECT * FROM exists_t1 t1 WHERE EXISTS (SELECT * FROM exists_t2 t2 WHERE t2.a =
t1.a);
 a | b
---+---
 3 | 4
 4 | 5
(2 rows)
```

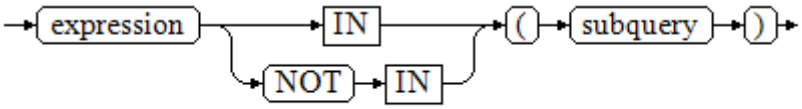
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# DROP TABLE exists_t1, exists_t2;
```

- IN/NOT IN

  Figure 1-11 shows the syntax of IN/NOT IN.

  Figure 1-11 IN/NOT IN::=

  The right-hand side is a parenthesized subquery, which must return exactly one column. The left-hand expression is evaluated and compared to each row of the subquery result. The result of IN is true if any equal subquery row is found. The result is false if no equal row is found (including the case where the subquery returns no rows).

  This is in accordance with SQL normal rules for Boolean combinations of null values. If the columns corresponding to two rows equal and are not empty, the two rows are equal to each other. If any columns corresponding to the two rows do not equal and are not empty, the two rows are not equal to each other. Otherwise, the result is **NULL**. If the results in each row are either unequal or NULL, with at least one NULL, the result of IN is null.

  Operations on XML data are not supported.

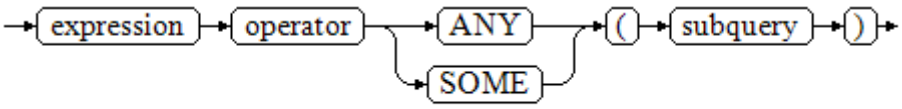
  Example:

  ```
  gaussdb=# CREATE TABLE in_t1(a int, b int);
  gaussdb=# INSERT INTO in_t1 VALUES(1, 2),(2, 3),(3, 4),(4, 5);

  gaussdb=# CREATE TABLE in_t2(a int, c int);
  gaussdb=# INSERT INTO in_t2 VALUES(3, 4),(4, 5),(5, 6),(6, 7);

  gaussdb=# SELECT * FROM in_t1 t1 WHERE t1.a IN (SELECT t2.a FROM in_t2 t2);
   a | b
  ---+---
   3 | 4
   4 | 5
  (2 rows)

  gaussdb=# DROP TABLE in_t1, in_t2;
  ```

- ANY/SOME

  Figure 1-12 shows the syntax of ANY/SOME.

  Figure 1-12 any/some::=

  The right-hand side is a parenthesized subquery, which must return exactly one column. The left-hand expression is evaluated and compared to each row of the subquery result using the given operator, which must yield a Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

result. The result of ANY is true if any true result is obtained. The result is false if no true result is found (including the case where the subquery returns no rows). SOME is a synonym of ANY. IN can be equivalently replaced with ANY.

Operations on XML data are not supported.

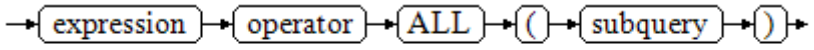
Example:

```
gaussdb=# CREATE TABLE any_t1(a int, b int);
gaussdb=# INSERT INTO any_t1 VALUES(1, 2),(2, 3),(3, 4),(4, 5);

gaussdb=# CREATE TABLE any_t2(a int, c int);
gaussdb=# INSERT INTO any_t2 VALUES(3, 4),(4, 5),(5, 6),(6, 7);

gaussdb=# SELECT * FROM any_t1 t1 WHERE t1.a < ANY(SELECT t2.a FROM any_t2 t2 where t2.a = 3
or t2.a = 4);
 a | b
---+---
 1 | 2
 2 | 3
 3 | 4
(3 rows)

gaussdb=# DROP TABLE any_t1, any_t2;
```

- ALL

  Figure 1-13 shows the syntax of ALL.

  Figure 1-13 all::=

  The right-hand side is a parenthesized subquery, which must return exactly one column. The left-hand expression is evaluated and compared to each row of the subquery result using the given operator, which must yield a Boolean result. The result of ALL is true if all values are true (including the case where the subquery returns no rows). The result is false if any false result is found.

  Operations on XML data are not supported.

  Example:

```
gaussdb=# CREATE TABLE all_t1(a int, b int);
gaussdb=# INSERT INTO all_t1 VALUES(1, 2),(2, 3),(3, 4),(4, 5);

gaussdb=# CREATE TABLE all_t2(a int, c int);
gaussdb=# INSERT INTO all_t2 VALUES(3, 4),(4, 5),(5, 6),(6, 7);

gaussdb=# SELECT * FROM all_t1 t1 WHERE t1.a < ALL(SELECT t2.a FROM all_t2 t2 where t2.a = 3 or
t2.a = 4);
 a | b
---+---
 1 | 2
 2 | 3
(2 rows)

gaussdb=# DROP TABLE all_t1, all_t2;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.7.4 Array Expressions

IN

expression **IN** *(value [, ...])*

The parentheses on the right contain an expression list. The expression result on the left is compared with the content in the expression list. If the content in the list meets the expression result on the left, the result of IN is true. If no result meets the requirements, the result of IN is false.

Example:

```
gaussdb=# SELECT 8000+500 IN (10000, 9000) AS RESULT;
 result
----------
 f
(1 row)
```

NOTE

- If the expression result is null or the expression list does not meet the expression conditions and at least one empty value is returned for the expression list on the right, the result of IN is null rather than false. This method is consistent with the Boolean rules used when SQL statements return empty values.
- Operations on XML data are not supported.

NOT IN

expression **NOT IN** *(value [, ...])*

The parentheses on the right contain an expression list. The expression result on the left is compared with the content in the expression list. If the content in the list does not meet the expression result on the left, the result of NOT IN is true. If any content meets the expression result, the result of NOT IN is false.

Example:

```
gaussdb=# SELECT 8000+500 NOT IN (10000, 9000) AS RESULT;
 result
----------
 t
(1 row)
```

NOTE

- If the query statement result is null or the expression list does not meet the expression conditions and at least one empty value is returned for the expression list on the right, the result of NOT IN is null rather than false. This method is consistent with the Boolean rules used when SQL statements return empty values.
- In all situations, X NOT IN Y equals to NOT(X IN Y).
- Operations on XML data are not supported.

ANY/SOME (array)

expression operator **ANY** *(array expression)*

expression operator **SOME** *(array expression)*

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The right side is a parenthesized expression, which must yield an array value. The result of the expression on the left uses operators to compute and compare the results in each row of the array expression. The comparison result must be a Boolean value.

Example:

```
gaussdb=# SELECT 8000+500 < SOME (array[10000,9000]) AS RESULT;
 result
----------
 t
(1 row)

gaussdb=# SELECT 8000+500 < ANY (array[10000,9000]) AS RESULT;
 result
----------
 t
(1 row)
```

NOTE

- If at least one comparison result is true, the result of ANY is true.
- If no comparison result is true, the result of ANY is false.
- If no comparison result is true and the array expression generates at least one null value, the value of **ANY** is **NULL** rather than **false**. This method is consistent with the Boolean rules used when SQL statements return empty values.
- SOME is a synonym of ANY.
- Operations on XML data are not supported.

ALL (array)

expression operator **ALL** *(array expression)*

The right-hand side is a parenthesized expression, which must yield an array value. The result of the expression on the left uses operators to compute and compare the results in each row of the array expression. The comparison result must be a Boolean value.

- The result of ALL is true if all comparisons yield true (including the case where the array has zero elements).
- If one or more comparison results are false, the result of ALL is false.
- If the array expression yields a null array, the result of ALL will be NULL. If the expression on the left yields NULL, the result of ALL is generally NULL (though a non-strict comparison operator could possibly yield a different result). If the array on the right contains any null elements and no false comparison result is found, the result of ALL is NULL, not true (again, assuming a strict comparison operator). This method is consistent with the Boolean rules used when SQL statements return empty values.
- Operations on XML data are not supported.

Example:

```
gaussdb=# SELECT 8000+500 < ALL (array[10000,9000]) AS RESULT;
 result
----------
 t
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.7.5 Row Expressions

The syntax is as follows:

```
row_constructor operator row_constructor
```

Both sides of the row expression are row constructors. The values of both rows must have the same number of fields and they are compared with each other. The row comparison allows operators including =, <>, <, <=, and >= or a similar operator.

For operators <, <=, >, and >=, the columns in rows are compared from left to right until a pair of columns that are not equal or are empty are detected. If the pair of columns contains at least one null value, the comparison result is null. Otherwise, the comparison result of this pair of columns is the final result. If no unequal or empty column is found, the values in the two rows are equal. The final result is determined based on the operator meaning.

Operations on XML data are not supported.

Example:

```
gaussdb=# SELECT ROW(1,2,NULL) < ROW(1,3,0) AS RESULT;
 result
----------
 t
(1 row)

gaussdb=# SELECT (4,5,6) > (3,2,1) AS result;
 result
--------
 t
(1 row)

gaussdb=# SELECT (4,1,1) > (3,2,1) AS result;
 result
--------
 t
(1 row)

gaussdb=# SELECT ('test','data') > ('data','data') AS result;
 result
--------
 t
(1 row)

gaussdb=# SELECT (4,1,1) > (3,2,null) AS result;
 result
--------
 t
(1 row)

gaussdb=# SELECT (null,1,1) > (3,2,1) AS result;
 result
--------

(1 row)

gaussdb=# SELECT (null,5,6) > (null,5,6) AS result;
 result
--------

(1 row)

gaussdb=# SELECT (4,5,6) > (4,5,6) AS result;
 result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------
 f
(1 row)

gaussdb=# SELECT (2,2,5) >= (2,2,3) AS result;
 result
--------
 t
(1 row)

gaussdb=# SELECT (2,2,1) <= (2,2,3) AS result;
 result
--------
 t
(1 row)
```

The use of operators = and <> is slightly different from other operators. If all columns in the two rows are not empty and meet the operator condition, the two rows meet the operator condition. If any column in the two rows is not empty and does not meet the operator condition, the two rows do not meet the operator condition. If any column in the two rows is empty, the comparison result is null.

Example:

```
gaussdb=# SELECT (1,2,3) = (1,2,3) AS result;
 result
--------
 t
(1 row)

gaussdb=# SELECT (1,2,3) <> (2,2,3) AS result;
 result
--------
 t
(1 row)

gaussdb=# SELECT (2,2,3) <> (2,2,null) AS result;
 result
--------

(1 row)

gaussdb=# SELECT (null,5,6) <> (null,5,6) AS result;
 result
--------

(1 row)
```

1.7.6 Time Interval Expressions

Syntax: INTERVAL EXPR UNIT

Description: **EXPR** indicates a value, and the **UNIT** specifier indicates a unit of the value, such as HOUR, DAY, and WEEK. The keyword INTERVAL and specifier are case-insensitive.

Table 1-331 describes the value range of **UNIT** in a time interval expression. Any punctuation-separated EXPR format is allowed. The recommended separators are displayed in **Table 1-331**.

NOTE

The INTERVAL expression supports the preceding functions only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **5.7**, and **b_format_dev_version** is set to **s1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-331 Value range of UNIT in a time interval expression

UNIT Value Range	Expected EXPR Format
MICROSECOND	MICROSECONDS
SECOND	SECONDS
MINUTE	MINUTES
HOUR	HOURS
DAY	DAYS
WEEK	WEEKS
MONTH	MONTHS
QUARTER	QUARTERS
YEAR	YEARS
SECOND_MICROSECOND	'SECOND_MICROSECOND'
MINUTE_MICROSECOND	'MINUTES:SECONDS.MICROSECONDS'
MINUTE_SECOND	'MINUTES:SECONDS'
HOUR_MICROSECOND	'HOURS:MINUTES:SECONDS.MICROSECONDS'
HOUR_SECOND	'HOURS:MINUTES:SECONDS'
HOUR_MINUTE	'HOURS:MINUTES'
DAY_MICROSECOND	'DAYS HOURS:MINUTES:SECONDS.MICROSECONDS'
DAY_SECOND	'DAYS HOURS:MINUTES:SECONDS'
DAY_MINUTE	'DAYS HOURS:MINUTES'
DAY_HOUR	'DAYS HOURS'
YEAR_MONTH	'YEAR_MONTH'

Example:

```
gaussdb=# SELECT DATE_ADD('2018-05-01', INTERVAL 1 DAY);
  date_add
------------
 2018-05-02
(1 row)

gaussdb=# SELECT DATE_SUB('2018-05-01', INTERVAL 1 YEAR);
  date_sub
------------
 2017-05-01
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT DATE'2023-01-10' - INTERVAL 1 DAY;
     ?column?
---------------------
 2023-01-09 00:00:00
(1 row)

gaussdb=# SELECT DATE'2023-01-10' + INTERVAL 1 MONTH;
     ?column?
---------------------
 2023-02-10 00:00:00
(1 row)
```

1.8 Pseudocolumn

ROWNUM is a pseudocolumn that returns a number indicating the row number of the obtained query result. The value of **ROWNUM** in the first row is **1**, the value of **ROWNUM** in the second row is **2**, and so forth. **ROWNUM** is used to limit the number of rows returned by a query.

Here is a breakdown of the ROWNUM mechanism:

- Timing of value assignment: **ROWNUM** is dynamically assigned during the generation of the query result set. It is a pseudocolumn added to the result set of every query. This means that sequence numbers are determined after the data is filtered by the WHERE clause but before any ORDER BY.
- Comparison operator: The **ROWNUM** <= *n* (*n* > 1) operator is valid and commonly used to limit the number of returned rows, such as retrieving the first *N* records. For example:

  ```
  gaussdb=# CREATE TABLE Students (name varchar(20), id int) with (STORAGE_TYPE = USTORE);
  gaussdb=# INSERT INTO Students VALUES ('Jack', 35);
  gaussdb=# INSERT INTO Students VALUES ('Leon', 15);
  gaussdb=# INSERT INTO Students VALUES ('James', 24);
  gaussdb=# INSERT INTO Students VALUES ('Taker', 81);
  gaussdb=# INSERT INTO Students VALUES ('Mary', 25);
  gaussdb=# INSERT INTO Students VALUES ('Rose', 64);
  gaussdb=# INSERT INTO Students VALUES ('Perl', 18);
  gaussdb=# INSERT INTO Students VALUES ('Under', 57);
  gaussdb=# INSERT INTO Students VALUES ('Angel', 101);
  gaussdb=# INSERT INTO Students VALUES ('Frank', 20);
  gaussdb=# INSERT INTO Students VALUES ('Charlie', 40);

  -- Output the first 10 rows of data records in the Students table.
  gaussdb=# SELECT * FROM Students WHERE rownum <= 10;
   name  | id
  -------+-----
   Jack  |  35
   Leon  |  15
   James |  24
   Taker |  81
   Mary  |  25
   Rose  |  64
   Perl  |  18
   Under |  57
   Angel | 101
   Frank |  20
  (10 rows)
  ```

- **ROWNUM = 1** is valid and can be used to retrieve the first record.

  ```
  gaussdb=# SELECT * FROM Students WHERE rownum = 1;
   name | id
  ------+----
   Jack | 35
  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The conditions **ROWNUM** = *n* (*n* > 1), **ROWNUM** > *n* (*n* >= 1), and **ROWNUM** >= *n* (*n* > 1) always yield a FALSE result, leading to no rows being returned. This is because the first row retrieved is assigned **ROWNUM=1**. If it fails to meet the specified condition (such as **ROWNUM > 1**), it will be discarded. The next record is then processed, becoming the new "first" record, and **ROWNUM** is reset to **1**. This cycle continues, with none of the rows meeting the specified condition.

```
gaussdb=# SELECT * FROM Students WHERE rownum > 1;
 name | id
------+----
(0 rows)
```

[Basic Usage of **ROWNUM**]

- Limiting the number of returned rows

```
gaussdb=# SELECT * FROM Students WHERE rownum <= 2;
 name | id
------+----
 Jack | 35
 Leon | 15
(2 rows)
```

- Implementing paginated queries: Since **ROWNUM** > *n* cannot be used directly, paginated queries need to be implemented by creating an alias (such as **rn**) for **ROWNUM** through a subquery. This turns **ROWNUM** into a "fixed" column, allowing outer queries to filter ranges accordingly.

```
gaussdb=# SELECT *
FROM (SELECT a.*, ROWNUM AS rn
      FROM
      (SELECT * FROM Students ORDER BY id DESC) a -- The inner subquery sorts data first.
      WHERE ROWNUM <= 10) -- The intermediate subquery defines the upper limit and generates an
alias for ROWNUM.
WHERE rn >= 5;
  name   | id | rn
---------+----+----
 Charlie | 40 |  5
 Jack    | 35 |  6
 Mary    | 25 |  7
 James   | 24 |  8
 Frank   | 20 |  9
 Perl    | 18 | 10
(6 rows)
```

- Obtaining the first *N* rows after sorting: When WHERE ROWNUM <= *N* and ORDER BY are simultaneously used at the same level of query, incorrect results may occur because **ROWNUM** is assigned before sorting takes place. The correct method is to use a subquery to sort rows and then assign **ROWNUM**.

```
gaussdb=# SELECT rownum, * FROM (SELECT * FROM Students ORDER BY 1) WHERE rownum <= 2;
 rownum |  name   | id
--------+---------+-----
      1 | Angel   | 101
      2 | Charlie |  40
(2 rows)
```

- Using **ROWNUM** to assign a value to each row within a specific range of the table

```
gaussdb=# SELECT * FROM Students;
  name   | id
---------+-----
 Jack    |  35
 Leon    |  15
 James   |  24
 Taker   |  81
 Mary    |  25
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 Rose    | 64
 Perl    | 18
 Under   | 57
 Angel   | 101
 Frank   | 20
 Charlie | 40
(11 rows)

gaussdb=# UPDATE Students SET id = id + 5 WHERE rownum < 4;
UPDATE 3
gaussdb=# SELECT * FROM Students;
  name   | id
---------+-----
 Jack    | 40
 Leon    | 20
 James   | 29
 Taker   | 81
 Mary    | 25
 Rose    | 64
 Perl    | 18
 Under   | 57
 Angel   | 101
 Frank   | 20
 Charlie | 40
(11 rows)

gaussdb=# DROP TABLE Students;
DROP TABLE
```

The restrictions on using **ROWNUM** are as follows:

- Do not use ROWNUM as an alias to avoid ambiguity in SQL statements.
- Do not use ROWNUM when creating an index.
- Do not use ROWNUM as the default value when creating a table.
- Do not use ROWNUM as an alias in the WHERE clause.
- Do not use ROWNUM when inserting data.
- Do not use ROWNUM in a tableless query.
- Do not use ROWNUM in the LIMIT clause.
- **ROWNUM** cannot be used as a parameter of the EXECUTE statement.
- Do not use ROWNUM to update a clause in the UPSERT statement.
- Do not use ROWNUM as a projection column or WHERE condition in the SELECT ... FOR UPDATE statement.
- If the HAVING clause contains **ROWNUM** (not in an aggregate function), the GROUP BY clause must also contain **ROWNUM** (not in an aggregate function), unless the GROUP BY clause contains an expression, such as **SELECT a + a FROM t group by a + a having rownum < 5**.
- If the **ROWNUM** condition exists in the HAVING clause, the HAVING clause cannot be pushed down to any scan node.

```
gaussdb=# CREATE TABLE test (a int, b int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# INSERT INTO test SELECT generate_series, generate_series FROM generate_series(1, 10);
INSERT 0 10

-- The rownum condition cannot be pushed down to seqscan.
gaussdb=# EXPLAIN SELECT a,rownum FROM test group by a,rownum having rownum < 5;
                            QUERY PLAN
------------------------------------------------------------------------------
 HashAggregate  (cost=3.45..3.49 rows=3 width=4)
   Group By Key: a, ROWNUM
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  Filter: ((ROWNUM) < 5::numeric)
  ->  Rownum  (cost=0.19..3.40 rows=10 width=4)
        ->  Streaming (type: GATHER)  (cost=0.19..3.40 rows=10 width=4)
              Node/s: All datanodes
              ->  Seq Scan on test  (cost=0.00..3.03 rows=10 width=4)
(7 rows)
```

- If a subquery contains the ROWNUM condition, the predicate cannot be pushed down to any scan node.

```
-- b<5 cannot be pushed down to seqscan.
gaussdb=# EXPLAIN SELECT * FROM (SELECT * FROM test WHERE rownum < 5) WHERE b < 5;
                                QUERY PLAN
-------------------------------------------------------------------------------------
 Streaming (type: GATHER)  (cost=0.06..0.63 rows=2 width=8)
   Node/s: All datanodes
   ->  Subquery Scan on __unnamed_subquery__  (cost=0.00..0.50 rows=2 width=8)
         Filter: (__unnamed_subquery__.b < 5)
         ->  Rownum  (cost=0.00..0.45 rows=4 width=8)
               StopKey: (ROWNUM < 5::numeric)
               ->  Streaming(type: BROADCAST)  (cost=0.00..1.36 rows=12 width=8)
                     Spawn on: All datanodes
                     ->  Rownum  (cost=0.00..1.35 rows=4 width=8)
                           StopKey: (ROWNUM < 5::numeric)
                           ->  Seq Scan on test  (cost=0.00..3.03 rows=10 width=8)
(11 rows)

gaussdb=# DROP TABLE test;
DROP TABLE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The query result of **ROWNUM** may be unstable due to the sequence in which the CN receives DN data.

```
gaussdb=# CREATE TABLE test(a int, b int);
gaussdb=# INSERT INTO test VALUES(generate_series(1,10),generate_series(1,10));

-- The CN receives data from DN1 and then from DN2.
gaussdb=# SELECT rownum,* FROM test;
 rownum | a  | b
--------+----+----
      1 |  1 |  1
      2 |  2 |  2
      3 |  5 |  5
      4 |  6 |  6
      5 |  8 |  8
      6 |  9 |  9
      7 |  3 |  3
      8 |  4 |  4
      9 |  7 |  7
     10 | 10 | 10
(10 rows)

-- The CN receives data from DN2 and then from DN1.
gaussdb=# SELECT rownum,* FROM test;
 rownum | a  | b
--------+----+----
      1 |  3 |  3
      2 |  4 |  4
      3 |  7 |  7
      4 | 10 | 10
      5 |  1 |  1
      6 |  2 |  2
      7 |  5 |  5
      8 |  6 |  6
      9 |  8 |  8
     10 |  9 |  9
(10 rows)
```

- It is not recommended that the ROWNUM condition be used in the JOIN ON clause. In GaussDB, when the ROWNUM condition is used in the JOIN ON clause, the behavior in the LEFT JOIN, RIGHT JOIN, FULL JOIN, and MERGE INTO scenarios is different from that in other databases, causing risks in service migration.

If the parent query contains the **rownum** restriction and the projection column of the subquery contains **rownum**, the restriction is pushed down to the subquery. The constraints are as follows:

- The parent query can be pushed down only when the **rownum** restriction condition is <, <=, or = and the subquery directly uses **rownum** as the pseudocolumn.
- If the parent query has multiple filter criteria for **rownum** in the subquery and the pushdown requirements are met, only the first filter criterion is pushed down.
- If a subquery contains volatile functions and stored procedures, the pushdown is not supported.

1.9 Type Conversion

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.9.1 Overview

Context

SQL is a typed language. That is, every data item has an associated data type which determines its behavior and allowed usage. GaussDB has an extensible type system that is more general and flexible than other SQL implementations. Hence, most type conversion behaviors in GaussDB are governed by general rules. This allows the use of mixed-type expressions.

The GaussDB scanner/parser divides lexical elements into five fundamental categories: integers, floating-point numbers, strings, identifiers, and keywords. Constants of most non-numeric types are first classified as strings. The SQL language definition allows specifying type names with constant strings. Example:

```
gaussdb=# SELECT text 'Origin' AS "label", point '(0,0)' AS "value";
 label  | value
--------+-------
 Origin | (0,0)
(1 row)
```

In this example, there are two literal constants of the text and point types, respectively. If a type is not specified for a string literal, then the placeholder type unknown is assigned initially.

There are four fundamental SQL constructs requiring distinct type conversion rules in GaussDB parser:

- Function calls

  Much of the SQL type system is built around a rich set of functions. Functions can have one or more arguments. Since SQL permits function overloading, the function name alone does not uniquely identify the function to be called. The parser must select the right function based on the data types of the supplied arguments.

- Operators

  SQL allows expressions with prefix and postfix unary (one-argument) operators, as well as binary (two-argument) operators. Like functions, operators can be overloaded, so the same problem of selecting the right operator exists.

- Value storage

  The INSERT and UPDATE statements place the results of expressions into a table. The expressions in the statement must be matched up with, and perhaps converted to, the types of the target columns.

- UNION, CASE, and Related Constructs

  Since all query results from a unionized SELECT statement must appear in a single set of columns, the types of the results of each SELECT clause must be matched up and converted to a uniform set. Similarly, the result expressions of a CASE construct must be converted to a common type so that the **CASE** expression as a whole has a known output type. The same holds for ARRAY constructs, and for the GREATEST and LEAST functions.

The system catalog pg_cast stores information about which conversions, or casts, exist between which data types, and how to perform those conversions. For details, see **PG_CAST**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The return type and conversion behavior of an expression are determined during semantic analysis. Data types are divided into several basic type categories, including Boolean, numeric, string, bitstring, datetime, timespan, geometric, and network. Within each category there can be one or more preferred types, which are preferred when there is a choice of possible types. With careful selection of preferred types and available implicit casts, it is possible to ensure that ambiguous expressions (those with multiple candidate parsing solutions) can be resolved in a useful way.

All type conversion rules are designed based on the following principles:

- Implicit conversions should never have surprising or unpredictable outcomes.
- There should be no extra overhead in the parser or executor if a query does not need implicit type conversion. That is, if a query is well-formed and the types already match, then the query should execute without spending extra time in the parser and without introducing unnecessary implicit conversion calls in the query.
- Additionally, if a query usually requires an implicit conversion for a function, and if then the user defines a new function with the correct argument types, the parser should use this new function.
- XML data does not support implicit type conversion, including implicit conversion between strings and XML types.
- In ORA-compatible mode, the DATEA type can be converted only to the text, bpchar, varchar, clob, nvarchar2, timestamp, and timestamp with time zone types.

1.9.2 Operators

Operator Type Resolution

1. Select the operators to be considered from the pg_operator system catalog. Considered operators are those with the matching name and argument count. If the search path finds multiple available operators, only the most suitable one is considered.
2. Look for the best match.
   a. Discard candidate operators for which the input types do not match and cannot be converted (using an implicit conversion) to match. Unknown text can be converted to anything for this purpose. If only one candidate remains, use it; else continue to the next step.
   b. Run through all candidates and keep those with the most exact matches on input types. In this case, the domain type is considered to be the same as its base type. Keep all candidates if none survives these tests. If only one candidate remains, use it; else continue to the next step.
   c. Run through all candidates and keep those that accept preferred types (of the input data type's type category) at the most positions where type conversion will be required. Keep all candidates if none accepts preferred types. If only one candidate remains, use it; else continue to the next step.
   d. If any input arguments are of unknown types, check the type categories accepted at those argument positions by the remaining candidates. At each position, select the string category if any candidate accepts that

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

category. (This bias towards string is appropriate since an unknown-type literal looks like a string.) Otherwise, if all the remaining candidates accept the same type category, select that category; otherwise fail because the correct choice cannot be deduced without more clues. Discard candidates that do not accept the selected type category. Furthermore, if any candidate accepts a preferred type in that category, discard candidates that accept non-preferred types for that argument. Keep all candidates if none survives these tests. If only one candidate remains, use it; else continue to the next step.

e. If there are both unknown and known arguments, and all the known arguments have the same type, assume that the unknown arguments are also of that type, and check which candidates can accept that type at the unknown-argument positions. If exactly one candidate passes this test, use it. Otherwise, an error occurs.

> NOTE
>
> After an operator is found, if the type of the input parameter is different from that of the operator, implicit type conversion may occur. After the conversion, unpredictable behavior may occur. If the behavior is incorrect after implicit conversion, you can use explicit type conversion to avoid this problem. For example, after the fixed-length bpchar type is converted to the variable-length text type, spaces at the end of the character string are deleted. If the character string is compared with other character strings, errors may occur.

Examples

Example 1: Use factorial operator type resolution. There is only one factorial operator (postfix !) defined in the system catalog, and it takes an argument of bigint type. The scanner assigns bigint as an initial type to the argument in this query expression:

```
gaussdb=# SELECT 40 ! AS "40 factorial";

                  40 factorial
-------------------------------------------------
 815915283247897734345611269596115894272000000000
(1 row)
```

The parser converts the parameter type. The query is equivalent to:

```
gaussdb=# SELECT CAST(40 AS bigint) ! AS "40 factorial";
```

Example 2: String concatenation operator type resolution. A string-like syntax is used for working with string types and for working with complex extension types. Strings with unspecified type are matched with likely operator candidates. An example with one unspecified argument:

```
gaussdb=# SELECT text 'abc' || 'def' AS "text and unknown";
 text and unknown
------------------
 abcdef
(1 row)
```

In this example, the parser looks for an operator whose parameters are of the text type. Such an operator is found.

Here is a concatenation of two values of unspecified types:

```
gaussdb=# SELECT 'abc' || 'def' AS "unspecified";
 unspecified
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-------------
 abcdef
(1 row)
```

NOTE

In this case there is no initial hint for which type to use, since no types are specified in the query. So, the parser looks for all candidate operators and finds that there are candidates accepting both string-category and bit-string-category inputs. Since string category is preferred when available, that category is selected, and then the preferred type for strings, text, is used as the specific type to resolve the unknown-type literals as.

Example 3: Absolute-value and negation operator type resolution. The GaussDB operator catalog has several entries for the prefix operator @. All the entries implement absolute-value operations for various numeric data types. One of these entries is for the float8 type, which is the preferred type in the numeric category. Therefore, GaussDB will use that entry when faced with an unknown input:

```
gaussdb=# SELECT @ '-4.5' AS "abs";
 abs
-----
 4.5
(1 row)
```

Here the system has implicitly resolved the unknown-type literal as the float8 type before applying the chosen operator.

Example 4: Use the array inclusion operator type resolution as an example. Here is another example of resolving an operator with one known and one unknown input:

```
gaussdb=# SELECT array[1,2] <@ '{1,2,3}' as "is subset";
 is subset
-----------
 t
(1 row)
```

NOTE

The GaussDB operator catalog has several entries for the infix operator <@, but the only two that could possibly accept an integer array on the left side are array inclusion (anyarray <@ anyarray) and range inclusion (anyelement <@ anyrange). Since none of these polymorphic pseudo-types (see **Pseudo-Types**) is considered preferred, the parser cannot resolve the ambiguity on that basis. However, the last resolution rule tells it to assume that the unknown-type literal is of the same type as the other input, that is, integer array. Now only one of the two operators can match, so array inclusion is selected. (Had range inclusion been selected, we would have gotten an error because the string does not have the right format to be a range literal.)

1.9.3 Functions

Function Type Resolution

- Select the functions to be considered from the pg_proc system catalog. If a non-schema-qualified function name was used, the functions in the current search path are considered. If a qualified function name was given, only functions in the specified schema are considered.

  If the search path finds multiple functions of different argument types, a proper function in the path is considered.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Check for a function accepting exactly the input argument types. If the function exists, use it. Cases involving unknown types will never find a match at this step.
- If no exact match is found, see if the function call appears to be a special type conversion request.
- Look for the best match.
  - Discard candidate functions for which the input types do not match and cannot be converted (using an implicit conversion) to match. Unknown text can be converted to anything for this purpose. If only one candidate remains, use it; else continue to the next step.
  - Run through all candidates and keep those with the most exact matches on input types. Domains are considered the same as their base type for this purpose. Keep all candidates if none has exact matches. If only one candidate remains, use it; else continue to the next step.
  - Run through all candidates and keep those that accept preferred types at the most positions where type conversion will be required. Keep all candidates if none accepts preferred types. If only one candidate remains, use it; else continue to the next step.
  - If any input arguments are of unknown types, check the type categories accepted at those argument positions by the remaining candidates. At each position, select the string category if any candidate accepts that category. (This bias towards string is appropriate since an unknown-type literal looks like a string.) Otherwise, if all the remaining candidates accept the same type category, select that category; otherwise fail because the correct choice cannot be deduced without more clues. Now discard candidates that do not accept the selected type category. Furthermore, if any candidate accepts a preferred type in that category, discard candidates that accept non-preferred types for that argument. Keep all candidates if none survives these tests. If only one candidate remains, use it; else continue to the next step.
  - If there are both unknown and known arguments, and all the known arguments have the same type, assume that the unknown arguments are also of that type, and check which candidates can accept that type at the unknown-argument positions. If exactly one candidate passes this test, use it. Otherwise, an error occurs.

Examples

Example 1: Use the rounding function argument type resolution as the first example. There is only one round function that takes two arguments; it takes a first argument of type numeric and a second argument of type integer. In this case, the following query automatically converts the first argument of type integer to numeric:

```
gaussdb=# SELECT round(4, 4);
 round
--------
 4.0000
(1 row)
```

That query is actually transformed by the parser to:

```
gaussdb=# SELECT round(CAST (4 AS numeric), 4);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Since numeric constants with decimal points are initially assigned the type numeric, the following query will require no type conversion and therefore might be slightly more efficient:

```
gaussdb=# SELECT round(4.0, 4);
```

Example 2: Use the substring function type resolution as the second example. There are several substr functions, one of which takes types text and integer. If the function is called with a string constant of unspecified type, the system chooses the candidate function that accepts an argument of the preferred category string (namely of type text).

```
gaussdb=# SELECT substr('1234', 3);
 substr
--------
    34
(1 row)
```

If the string is declared to be of type varchar, as might be the case if it comes from a table, then the parser will try to convert it to text:

```
gaussdb=# SELECT substr(varchar '1234', 3);
 substr
--------
    34
(1 row)
```

This is transformed by the parser to effectively become:

```
gaussdb=# SELECT substr(CAST (varchar '1234' AS text), 3);
```

NOTE

The parser learns from the pg_cast catalog that text and varchar are binary-compatible, meaning that one can be passed to a function that accepts the other without doing any physical conversion. Therefore, no type conversion is really inserted in this case.

And, if the function is called with an argument of type integer, the parser will try to convert that to text:

```
gaussdb=# SELECT substr(1234, 3);
substr
--------
 34
(1 row)
```

This is transformed by the parser to effectively become:

```
gaussdb=# SELECT substr(CAST (1234 AS text), 3);
 substr
--------
    34
(1 row)
```

1.9.4 Value Storage

Value Storage Type Resolution

1. Search for an exact match with the target column.
2. Try to directly convert the expression to the target type. If a registered conversion function exists between the two types, call the conversion function directly. If the expression is a text string of an unknown or unspecified type, the content of the text string is converted to the target type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3. Check whether the target type supports length conversion (a conversion from a type to itself). If a corresponding length conversion rule is found in the pg_cast catalog, apply it to the expression before storing into the destination column. This type of conversion functions receive a required parameter of the integer type to identify the **atttypmod** value of the target column (that is, the declared length of the column; the interpretation of **atttypmod** varies depending on the data type). In addition, this type of conversion function can also receive an optional parameter of the Boolean type to identify whether the conversion is explicit or implicit. The conversion function is responsible for implementing length-related semantic logic, such as length checks and content truncation.

Type Conversion Supported by Common Types in ORA-compatible Mode

Table 1-332 Common type conversion support

Source Type	Supported Target Type
int1	int2, int4, int8, float4, float8, numeric, bpchar, varchar, nvarchar2, clob
int2	int1, int4, int8, float4, float8, numeric, bpchar, varchar, nvarchar2, clob
int4	int1, int2, int8, float4, float8, numeric, bpchar, varchar, nvarchar2, clob
int8	int1, int2, int4, float4, float8, numeric, bpchar, varchar, nvarchar2, clob
float4	int1, int2, int4, int8, float8, numeric, bpchar, varchar, nvarchar2, clob
float8	int1, int2, int4, int8, float4, numeric, bpchar, varchar, nvarchar2, clob
numeric	int1, int2, int4, int8, float4, float8, bpchar, varchar, nvarchar2, clob
date	timestamp, timestamptz, bpchar, varchar
datea	timestamp, timestamptz, bpchar, varchar, nvarchar2, clob
timestamp	datea, timestamptz, varchar, nvarchar2, clob
timestamptz	timestamp, datea, clob
bpchar	int4, int8, float4, float8, numeric, date, datea, timestamp, varchar, nvarchar2, clob

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Source Type	Supported Target Type
varchar	int4, int8, float4, float8, numeric, date, datea, timestamp, bpchar, nvarchar2, clob, raw, blob
nvarchar2	int4, numeric, datea, timestamp, bpchar, varchar, clob
clob	int1, int2, int4, int8, float4, float8, numeric, datea, timestamp, bpchar, varchar, nvarchar2
raw	varchar, blob
blob	raw
xml	bpchar, varchar, nvarchar2, clob
xmltype	-

Examples

Use the character storage type conversion as an example. For a target column declared as character(20), the following statement shows that the stored value is sized correctly:

```
gaussdb=# CREATE SCHEMA tpcds;
gaussdb=# CREATE TABLE tpcds.value_storage_t1 (
    VS_COL1 CHARACTER(20)
)DISTRIBUTE BY HASH (VS_COL1);
gaussdb=# INSERT INTO tpcds.value_storage_t1 VALUES('abcdef');
gaussdb=# SELECT VS_COL1, octet_length(VS_COL1) FROM tpcds.value_storage_t1;
       vs_col1        | octet_length
----------------------+--------------
 abcdef               |           20
(1 row)
)

gaussdb=# DROP TABLE tpcds.value_storage_t1;
gaussdb=# DROP SCHEMA tpcds;
```

NOTE

What has happened here is that the two unknown literals are resolved to text by default, allowing the **||** operator to be resolved as text concatenation. Then the text result of the operator is converted to bpchar ("blank-padded char", the internal name of the character data type) to match the target column type. Since the conversion from text to bpchar is binary-coercible, this conversion does not insert any real function call. Finally, the sizing function **bpchar(bpchar, integer, Boolean)** is found in the system catalog and used for the operator's result and the stored column length. This type-specific function performs the required length check and addition of padding spaces.

1.9.5 UNION, CASE, and Related Constructs

SQL UNION constructs must match up possibly dissimilar types to become a single result set. The resolution algorithm is applied separately to each output column of a union query. The INTERSECT and EXCEPT constructs resolve dissimilar types in the same way as UNION. The CASE, ARRAY, VALUES, GREATEST and LEAST

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

constructs use the identical algorithm to match up their component expressions and select a result data type.

Type Resolution for UNION, CASE, and Related Constructs

- If all inputs are of the same type and are not unknown, resolve them as the unknown type.
- If all inputs are of the unknown type, resolve them as the text type (the preferred type of the string category). Otherwise, unknown inputs are ignored.
- If the inputs are not all of the same type category, a failure will be resulted. (The unknown type is not included in this case.)
- If the inputs are all of the same type category, choose the top preferred type in that category. (Exception: The UNION operation regards the type of the first branch as the selected type.)

  NOTE

  typcategory in the pg_type system catalog indicates the data type category. **typispreferred** specifies whether a type is preferred in **typcategory**.
- Convert all inputs to the selected type. (Retain the original lengths of strings). Fail if there is not an implicit conversion from a given input to the selected type.
- If the input contains the xml, xmltype, json, txid_snapshot, sys_refcursor, or geometry type, UNION cannot be performed.

The following describes the derivation rules of common types UNION and CASE in ORA-compatible mode. For details about common types, see **Data Types**.

NOTE

The derivation of some UNION and CASE types is related to the parsing sequence of expressions. UNION type derivation processes the output expression of each clause from left to right. CASE type derivation starts with the ELSE clause (default result expression) and then processes the result expression of each WHEN clause from left to right.

Table 1-333 Numeric types

Type (vertical: type 1; horizontal: type 2)	int1	int2	int4	int8	float4	float8	numeric
int1	int1	int2	int4	int8	float4	float8	numeric
int2	int2	int2	int4	int8	float4	float8	numeric
int4	int4	int4	int4	int8	float4	float8	numeric
int8	int8	int8	int8	int8	float4	float8	numeric

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

float4	float4	float4	float4	float4	float4	float8	numeric
float8	float8	float8	float8	float8	float8	float8	float8
numeric	numeric	numeric	numeric	numeric	numeric	numeric	numeric

Table 1-334 Datetime types

Type (vertical: type 1; horizontal: type 2)	datea	timestamp	timestamptz
datea	datea	timestamp	timestamptz
timestamp	timestamp	timestamp	timestamptz
timestamptz	timestamptz	timestamptz	timestamptz

Table 1-335 Character types

Type (vertical: type 1; horizontal: type 2)	bpchar	varchar	nvarchar2	clob
bpchar	bpchar	bpchar	bpchar	bpchar
varchar	varchar	varchar	varchar	varchar
nvarchar2	nvarchar2	nvarchar2	nvarchar2	nvarchar2
clob	clob	clob	clob	clob

Table 1-336 Binary types

Type (vertical: type 1; horizontal: type 2)	raw	blob
raw	raw	raw
blob	blob	blob

Type Resolution for CASE and COALESCE in TD Compatibility Type

- If all inputs are of the same type and are not unknown, resolve them as the unknown type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If all inputs are of the unknown type, resolve them as the text type.
- If inputs are of the string type (including unknown which is resolved as text) and digit type, resolve them as the string type. If the inputs are not of the two types, an error will be reported.
- If the inputs are all of the same type category, choose the top preferred type in that category.
- Convert all inputs to the selected type. Fail if there is not an implicit conversion from a given input to the selected type.

Type Resolution for CASE in ORA-compatible Mode

decode(expr, search1, result1, search2, result2, ..., defresult): When **sql_beta_feature** is set to **a_style_coerce**, the final return value type of the expression is set to the data type of result1 or a higher-precision data type in the same type as result1, as that in ORA-compatible mode. For example, numeric and int are both numeric data types, but numeric has higher precision and priority than int. For CASE WHEN, the behavior is the same as the default behavior in ORA-compatible mode.

- If all inputs are of the same type and are not unknown, resolve them as the unknown type. Otherwise, proceed to the next step.
- Set the data type of result1 to the final return value type preferType, which belongs to preferCategory.
- Consider the data types of result2, result3, and defresult in sequence. If the type category is also preferCategory, which is the same as that of result1, check whether the precision (priority) is higher than that of preferType. If it is, update preferType to a data type with higher precision. If the precision is higher than that of preferCategory, check whether the category can be implicitly converted to preferType. If it cannot, an error is reported.
- Uses the data type recorded by preferType as the return value type of the expression. The expression result is implicitly converted to this data type.

Note 1:

There is a special case where the character type of a super-large number is converted to the numeric type, for example, **select decode(1, 2, 2, '53465465676465454657567678676')**, in which the large number exceeds the range of the bigint and double types. If result1 is of the numeric type and does not meet the condition that all inputs are of the same type, the type of the return value is set to numeric to be compatible with this special case.

Note 2:

Priority of the numeric types: numeric > float8 > float4 > int8 > int4 > int2 > int1

Priority of the character types: text > varchar (nvarchar2) > bpchar > char

Priority of the date type: timestamptz > timestamp > datea (valid only in ORA-compatible mode) > smalldatetime > date > abstime > timetz > time

Priority of date span types: interval > tinterval > reltime

Note 3:

In Oracle-compatible mode, when **set sql_beta_feature** is set to **'a_style_coerce'**, the supported implicit type conversions are shown in **Figure 1-14**. **** indicates that

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

conversion is not required, **yes** indicates that conversion is supported, and the blank value indicates that conversion is not supported.

Figure 1-14 Support for implicit conversions

	bool	int1	int2	int4	int8	float4	float8	numeric	money	char	bpchar	varchar2	nvarchar2	text/clob	raw	blob	date	time	timetz	timestamp	timestamptz	smalldatetime	interval	reltime	abstime
bool	\																								
int1		\	yes	yes	yes	yes	yes	yes		yes	yes	yes	yes	yes											
int2		yes	\	yes	yes	yes	yes	yes		yes	yes	yes	yes	yes											
int4		yes	yes	\	yes	yes	yes	yes		yes	yes	yes	yes	yes											
int8		yes	yes	yes	\	yes	yes	yes		yes	yes	yes	yes	yes											
float4		yes	yes	yes	yes	\	yes	yes		yes	yes	yes	yes	yes											
float8		yes	yes	yes	yes	yes	\	yes		yes	yes	yes	yes	yes											
numeric		yes	yes	yes	yes	yes	yes	\		yes	yes	yes	yes	yes											
money									\																
char		yes	yes	yes	yes	yes	yes	yes		\	yes	yes	yes	yes											
bpchar		yes	yes	yes	yes	yes	yes	yes		yes	\	yes	yes	yes											
varchar2		yes	yes	yes	yes	yes	yes	yes		yes	yes	\	yes	yes	yes										
nvarchar2		yes	yes	yes	yes	yes	yes	yes		yes	yes	yes	\	yes											
text/clob		yes	yes	yes	yes	yes	yes	yes		yes	yes	yes	yes	\											
raw												yes		yes	\	yes									
blob															yes	\									
date												yes	yes	yes			\			yes	yes	yes			yes
time														yes				\	yes						
timetz														yes				yes	\						
timestamp												yes	yes	yes			yes			\	yes	yes			yes
timestamptz														yes			yes			yes	\	yes			yes
smalldatetime												yes		yes			yes			yes	yes	\			yes
interval												yes	yes	yes									\	yes	
reltime														yes									yes	\	
abstime														yes			yes			yes	yes	yes			\

Examples

Example 1: Use type resolution with underspecified types in a union. The unknown-type literal **'b'** will be resolved to the text type.

```
gaussdb=# SELECT text 'a' AS "text" UNION SELECT 'b';
 text
------
 a
 b
(2 rows)
```

Example 2: Use type resolution in a simple union. The literal **1.2** is of numeric type, and the integer value **1** can be cast implicitly to numeric type, so that type is used.

```
gaussdb=# SELECT 1.2 AS "numeric" UNION SELECT 1;
 numeric
---------
       1
     1.2
(2 rows)
```

Example 3: Use type resolution in a transposed union. The type real cannot be implicitly cast to integer, but integer can be implicitly cast to real, the union result type is resolved as real.

```
gaussdb=# SELECT 1 AS "real" UNION SELECT CAST('2.2' AS REAL);
 real
------
    1
  2.2
(2 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example 4: In TD mode, if input parameters for **COALESCE** are of int and varchar types, resolve them as the varchar type. In ORA-compatible mode, an error is reported.

```
-- Create the ORA-compatible database ora_1.
gaussdb=# CREATE DATABASE ora_1 dbcompatibility = 'ORA';

-- Switch to the ora_1 database.
gaussdb=# \c ora_1

-- Create the t1 table.
ora_1=# CREATE TABLE t1(a int, b varchar(10));

-- Show the execution plan of a statement for querying the types int and varchar of input parameters for COALESCE.
a_1=# EXPLAIN SELECT coalesce(a, b) FROM t1;
ERROR:  COALESCE types integer and character varying cannot be matched
LINE 1: EXPLAIN SELECT coalesce(a, b) FROM t1;
                               ^
CONTEXT:  referenced column: coalesce

-- Delete the table.
ora_1=# DROP TABLE t1;

-- Switch to the testdb database.
ora_1=# \c testdb

-- In TD mode, create the td_1 database compatible with Teradata.
gaussdb=# CREATE DATABASE td_1 dbcompatibility = 'TD';

-- Switch to the td_1 database.
gaussdb=# \c td_1

-- Create the t2 table.
td_1=# CREATE TABLE t2(a int, b varchar(10));

-- Show the execution plan of a statement for querying the types int and varchar of input parameters for COALESCE.
td_1=# EXPLAIN VERBOSE select coalesce(a, b) from t2;
                                  QUERY PLAN
-------------------------------------------------------------------------------------
 Data Node Scan  (cost=0.00..0.00 rows=0 width=0)
   Output: (COALESCE((t2.a)::character varying, t2.b))
   Node/s: All datanodes
   Remote query: SELECT COALESCE(a::character varying, b) AS "coalesce" FROM public.t2
(4 rows)

-- Delete the table.
td_1=# DROP TABLE t2;

-- Switch to the testdb database.
td_1=# \c testdb

-- Delete Oracle- and TD-compatible databases.
gaussdb=# DROP DATABASE ora_1;
gaussdb=# DROP DATABASE td_1;
```

Example 5: In ORA-compatible mode, set the final return value type of the expression to the data type of result1 or a higher-precision data type whose category is the same as that of the data type of result1.

```
-- In ORA-compatible mode, create the ora_1 database compatible with ORA.
gaussdb=# CREATE DATABASE ora_1 dbcompatibility = 'ORA';

-- Switch to the ora_1 database.
gaussdb=# \c ora_1

-- Enable the decode compatibility parameters.
ora_1=# SET sql_beta_feature='a_style_coerce';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create the t1 table.
ora_1=# CREATE TABLE t1(c_int int, c_float8 float8, c_char char(10), c_text text, c_date date);

-- Insert data.
ora_1=# INSERT INTO t1 VALUES(1, 2, '3', '4', date '12-10-2010');

-- The data type of result1 is char and that of defresult is text. The precision of text is higher, and the type of the return value is changed to text from char.
ora_1=# SELECT decode(1, 2, c_char, c_text) AS result, pg_typeof(result) FROM t1;
 result | pg_typeof
--------+-----------
 4      | text
(1 row)

-- The data type of result1 is int, which is a numeric type. The type of the return value is set to numeric.
ora_1=# SELECT decode(1, 2, c_int, c_float8) AS result, pg_typeof(result) FROM t1;
 result | pg_typeof
--------+-----------
      2 | numeric
(1 row)

-- The implicit conversion from the data type of defresult to that of result1 does not exist. If it is performed, an error is reported.
ora_1=# SELECT decode(1, 2, c_int, c_date) FROM t1;
ERROR:  CASE types integer and timestamp without time zone cannot be matched
LINE 1: SELECT decode(1, 2, c_int, c_date) FROM t1;
                            ^
CONTEXT:  referenced column: c_date

-- Disable the decode compatibility parameters.
ora_1=# SET sql_beta_feature='none';

-- Delete the table.
ora_1=# DROP TABLE t1;
DROP TABLE

-- Switch to the testdb database.
ora_1=# \c testdb

-- Delete the ORA-compatible database.
gaussdb=# DROP DATABASE ora_1;
DROP DATABASE
```

1.10 System Operation

GaussDB runs SQL statements to perform different system operations, such as setting variables, displaying the execution plan, and collecting garbage data.

Setting Variables

For details about how to set various parameters for a session or transaction, see **SET**.

Displaying the Execution Plan

For details about how to display the execution plan that GaussDB makes for SQL statements, see **EXPLAIN**.

Specifying a Checkpoint in Transaction Logs

By default, WALs periodically specify checkpoints in a transaction log. CHECKPOINT performs an immediate checkpoint when the related command is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

issued, without waiting for a regular checkpoint scheduled by the system. For details, see **CHECKPOINT**.

Collecting Unnecessary Data

For details about how to collect garbage data and analyze a database as required, For details, see **VACUUM**.

Collecting Statistics

For details about how to collect statistics on tables in databases, For details, see **ANALYZE | ANALYSE**.

Setting the Constraint Check Mode for the Current Transaction

For details about how to set the constraint check mode for the current transaction, For details, see **SET CONSTRAINTS**.

1.11 Controlling Transactions

A transaction is a user-defined sequence of database operations, which form an integral unit of work.

Starting a Transaction

GaussDB starts a transaction using **START TRANSACTION** and **BEGIN**. For details, see **START TRANSACTION** and **BEGIN**.

Setting a Transaction

GaussDB sets a transaction using **SET TRANSACTION** or **SET LOCAL TRANSACTION**. For details, see **SET TRANSACTION**.

Committing a Transaction

GaussDB commits all operations of a transaction using **COMMIT** or **END**. For details, see **COMMIT | END**.

Rolling Back a Transaction

If a fault occurs during a transaction and the transaction cannot proceed, the system performs rollback to cancel all the completed database operations related to the transaction. For details, see **ROLLBACK**.

NOTE

If an execution request (not in a transaction block) received in the database contains multiple statements, the request is packed into a transaction. If one of the statements fails, the entire request will be rolled back.

1.12 Full-Text Search

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.12.1 Introduction

1.12.1.1 Full-Text Search

Text search operators have been used in databases for years. GaussDB provides ~, ~*, LIKE, and ILIKE operators for text data type searching and filtering. However, they lack many essential text attributes required by modern information systems. These problems can be solved by using full-text search indexes and dictionaries.

Text search lacks the following essential properties required by information systems:

- There is no semantic support, even for English.

  Regular expressions are not sufficient because they cannot easily handle derived words. For example, when you search for **satisfy** using the regular expressions, documents that contain **satisfies** cannot be found. It is possible to use OR to search for multiple derived forms, but this is tedious and error-prone because some words can have several thousand derivatives.
- They provide no ordering (ranking) of search results, which makes them ineffective when thousands of matching documents are found.
- They tend to be slow because there is no index support, so they must process all documents for every search.

Full-text search preprocesses documents to facilitate subsequent search. The preprocessing process includes:

- Parse documents into tokens.

  It is useful to identify various classes of tokens, for example, numbers, words, compound words, and email addresses, so that they can be processed differently. In principle, token classes depend on the specific application, but for most purposes it is adequate to use a predefined set of classes.
- Convert tokens into lexemes.

  A lexeme is a string, just like a token, but it has been normalized so that different forms of the same word are made alike. Normalization almost always includes folding upper-case letters to lower-case, and often involves removal of suffixes (such as "s" or "es" in English). This allows searches to find variant forms of the same word, without entering all the possible variants. Also, this step typically eliminates stop words, which are so common and usually useless for searching. (In short, tokens are raw fragments of the document text, while lexemes are words that are believed useful for indexing and searching.) GaussDB uses dictionaries to perform this step and provides various standard dictionaries.
- Store preprocessed documents.

  Each document can be represented as a sorted array of normalized lexemes. Along with the lexemes, it is often desirable to store positional information for proximity ranking. Therefore, a document that contains a more "dense" area of search words is assigned a higher rank than the one with scattered search words.

Dictionaries allow granular control over how tokens are normalized. With appropriate dictionaries, you can define stop words that should not be indexed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The tsvector data type is provided for storing preprocessed documents, along with the tsquery type for storing query conditions. For details, see **Text Search Types**. For details about the functions and operators available for these data types, see **Text Search Functions and Operators**. The match operator @@, which is the most important among those functions and operators, is described in **Basic Text Matching**.

1.12.1.2 What Is a Document

A document is the unit of searching in a full-text search system; for example, a magazine article or email message. The text search engine must be able to parse documents and store associations of lexemes (keywords) with their parent document. Later, these associations are used to search for documents that contain search words.

In GaussDB, a document is normally a textual column within a row of a database table, or possibly a combination (concatenation) of such columns, perhaps stored in several tables or obtained dynamically. In other words, a document can be constructed from different parts for indexing and it might not be stored anywhere as a whole. For example:

```
gaussdb=# CREATE TABLE document_table (id int,person text,time timestamp,location text,thing
text,document text);
CREATE TABLE
gaussdb=# INSERT INTO document_table VALUES('1','Jack','2024-01-01 10:00:00','company','working','Jack
worked at the company on 2024-01-01 10:00:00.');
INSERT 0 1
gaussdb=# SELECT person || '-' || thing || '-' || location || '-' || time AS identify_serials FROM document_table
WHERE id =1;
          identify_serials
----------------------------------------
 Jack-working-company-2024-01-01 10:00:00
(1 row)
gaussdb=# DROP TABLE document_table;
```

Another possibility is to store the documents as simple text files in the file system. In this case, the database can be used to store indexes for full-text search, and some unique identifiers can be used to search the documents in the file system. However, searching documents outside the database requires system administrator permissions or special function support. Therefore, this is less convenient than keeping all the data inside the database. Also, keeping everything inside the database allows easy access to document metadata to assist in indexing and display.

For text search purposes, each document must be compacted to the preprocessed tsvector format. Searching and relevance-based ranking are performed entirely on the tsvector representation of a document. The raw text is retrieved only when the document has been selected for display to a user. Therefore, tsvector is usually referred to as a document, but it is only a compact representation of the full document.

1.12.1.3 Basic Text Matching

Full-text search in GaussDB is based on the match operator @@, which returns **true** if a tsvector (document) matches a tsquery (query). It does not matter which data type is written first:

```
gaussdb=# SELECT 'a fat cat sat on a mat and ate a fat rat'::tsvector @@ 'cat & rat'::tsquery AS RESULT;
 result
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
----------
 t
(1 row)

gaussdb=# SELECT 'fat & cow'::tsquery @@ 'a fat cat sat on a mat and ate a fat rat'::tsvector AS RESULT;
 result
----------
 f
(1 row)
```

As the above example suggests, a tsquery is not only raw text, but more than a tsvector is. A tsquery contains search words, which must be already-normalized lexemes, and may combine multiple words using AND, OR, NOT, and PHRASE operators. For details, see **Text Search Types**. The to_tsquery, plainto_tsquery, and phraseto_tsquery functions are helpful for converting user-written text into a proper tsquery, for example, normalizing words appearing in the text. Similarly, to_tsvector is used to parse and normalize a document string. The following is an example of text search matching in practice:

```
gaussdb=# SELECT to_tsvector('fat cats ate fat rats') @@ to_tsquery('fat & rat') AS RESULT;
result
----------
 t
(1 row)
```

Note that this match would not succeed if it is written as follows:

```
gaussdb=# SELECT 'fat cats ate fat rats'::tsvector @@ to_tsquery('fat & rat')AS RESULT;
result
----------
 f
(1 row)
```

In the preceding match, no normalization of the word **rats** will occur. Therefore, **rats** does not match **rat**.

The @@ operator also supports textual input, allowing explicit conversion of a text string to tsvector or tsquery to be skipped in simple cases. The variants available are:

```
tsvector @@ tsquery
tsquery  @@ tsvector
text @@ tsquery
text @@ text
```

In the preceding information, the form **text @@ tsquery** is equivalent to **to_tsvector(text) @@ tsquery**. The form **text @@ text** is equivalent to **to_tsvector(text) @@ plainto_tsquery(text)**.

1.12.1.4 Configuration

Full-text search functionality includes the ability to do many more things, for example, skipping certain words (stop words), processing synonyms, and sophisticated parsing. This functionality is controlled by text search configurations. GaussDB supports predefined text search configurations in multiple languages.

During installation, you can select an appropriate text search configuration and set **default_text_search_config** accordingly in **gaussdb.conf**. If you are using the same text search configuration for the cluster, you can use the value in **gaussdb.conf**. To use different text search configurations in the cluster, run **ALTER DATABASE ... SET** in the corresponding database. Alternatively, you can set **default_text_search_config** in each session.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Each text search function has an optional **regconfig** parameter, so the configuration to be used can be explicitly specified. **default_text_search_config** is used only when this parameter is ignored.

1.12.2 Tables and Indexes

1.12.2.1 Searching a Table

It is possible to do a full-text search without an index.

- A simple query to print each row that contains the word **america** in its **body** column is as follows:

```
gaussdb=# DROP SCHEMA IF EXISTS tsearch CASCADE;
gaussdb=# CREATE SCHEMA tsearch;
gaussdb=# CREATE TABLE tsearch.pgweb(id int, body text, title text, last_mod_date date)
with(storage_type = ASTORE);
gaussdb=# INSERT INTO tsearch.pgweb VALUES(1, 'America is a rock band, formed in England in
1970 by multi-instrumentalists Dewey Bunnell, Dan Peek, and Gerry Beckley.', 'America', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(2, 'England is a country that is part of the United
Kingdom. It shares land borders with Scotland to the north and Wales to the west.', 'England',
'2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(3, 'Australia, officially the Commonwealth of
Australia, is a country comprising the mainland of the Australian continent, the island of Tasmania,
and numerous smaller islands.', 'Australia', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(4, 'Japan is an island country in East Asia.', 'Japan',
'2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(5, 'Germany, officially the Federal Republic of
Germany, is a sovereign state and federal parliamentary republic in central-western Europe.',
'Germany', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(6, 'France, is a sovereign state comprising territory in
western Europe and several overseas regions and territories.', 'France', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(7, 'Italy officially the Italian Republic, is a unitary
parliamentary republic in Europe.', 'Italy', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(8, 'India, officially the Republic of India, is a country
in South Asia.', 'India', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(9, 'Brazil, officially the Federative Republic of Brazil,
is the largest country in both South America and Latin America.', 'Brazil', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(10, 'Canada is a country in the northern half of
North America.', 'Canada', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(11, 'Mexico, officially the United Mexican States, is a
federal republic in the southern part of North America.', 'Mexico', '2010-1-1');

gaussdb=# SELECT id, body, title FROM tsearch.pgweb WHERE to_tsvector('english', body) @@
to_tsquery('english', 'america');
 id |                                                 body                                                 | title
----
+--------------------------------------------------------------------------------------------------------------------------------
----+---------
  1 | America is a rock band, formed in England in 1970 by multi-instrumentalists Dewey Bunnell, Dan
Peek, and Gerry Beckley. | America
  9 | Brazil, officially the Federative Republic of Brazil, is the largest country in both South America
and Latin America.   | Brazil
 10 | Canada is a country in the northern half of North
America.                                                | Canada
 11 | Mexico, officially the United Mexican States, is a federal republic in the southern part of North
America.             | Mexico
(4 rows)
```

  This will also find related words, such as **America**, since all these are processed to the same normalized lexeme.

  The query above specifies that the **english** tokenizer is used to parse and normalize the strings. You can also skip this configuration and perform the configuration through **default_text_search_config**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SHOW default_text_search_config;
 default_text_search_config
----------------------------
 pg_catalog.english
(1 row)

gaussdb=# SELECT id, body, title FROM tsearch.pgweb WHERE to_tsvector(body) @@
to_tsquery('america');
 id |                                                 body                                                 | title
----
+----------------------------------------------------------------------------------------------------------------------------------
----+---------
  1 | America is a rock band, formed in England in 1970 by multi-instrumentalists Dewey Bunnell, Dan
Peek, and Gerry Beckley. | America
  9 | Brazil, officially the Federative Republic of Brazil, is the largest country in both South America
and Latin America.   | Brazil
 10 | Canada is a country in the northern half of North
America.                                             | Canada
 11 | Mexico, officially the United Mexican States, is a federal republic in the southern part of North
America.              | Mexico
(4 rows)
```

- A more complex example to select the ten most recent documents that contain **north** and **america** in the **title** or **body** column is as follows:

  ```
  gaussdb=# SELECT title FROM tsearch.pgweb WHERE to_tsvector(title || ' ' || body) @@
  to_tsquery('north & america') ORDER BY last_mod_date DESC LIMIT 10;
   title
  --------
   Canada
   Mexico
  (2 rows)

  -- Drop the table.
  gaussdb=# DROP TABLE tsearch.pgweb;
  gaussdb=# DROP SCHEMA tsearch;
  ```

  For clarity we omitted the coalesce function calls which would be needed to find rows that contain **NULL** in one of the two columns.

  The preceding examples show queries without using indexes. Most applications will find this approach too slow. Therefore, practical use of text search usually requires creating an index, except perhaps for occasional ad-hoc searches.

1.12.2.2 Creating an Index

You can create a GIN to speed up text search.

```
gaussdb=# CREATE SCHEMA tsearch;
gaussdb=# CREATE TABLE tsearch.pgweb(id int, body text, title text, last_mod_date date) with(storage_type
= ASTORE);
gaussdb=# INSERT INTO tsearch.pgweb VALUES(1, 'America is a rock band, formed in England in 1970 by
multi-instrumentalists Dewey Bunnell, Dan Peek, and Gerry Beckley.', 'America', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(2, 'England is a country that is part of the United
Kingdom. It shares land borders with Scotland to the north and Wales to the west.', 'England', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(3, 'Australia, officially the Commonwealth of Australia, is a
country comprising the mainland of the Australian continent, the island of Tasmania, and numerous
smaller islands.', 'Australia', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(4, 'Japan is an island country in East Asia.', 'Japan',
'2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(5, 'Germany, officially the Federal Republic of Germany, is
a sovereign state and federal parliamentary republic in central-western Europe.', 'Germany', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(6, 'France, is a sovereign state comprising territory in
western Europe and several overseas regions and territories.', 'France', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(7, 'Italy officially the Italian Republic, is a unitary
parliamentary republic in Europe.', 'Italy', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(8, 'India, officially the Republic of India, is a country in
South Asia.', 'India', '2010-1-1');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# INSERT INTO tsearch.pgweb VALUES(9, 'Brazil, officially the Federative Republic of Brazil, is the largest country in both South America and Latin America.', 'Brazil', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(10, 'Canada is a country in the northern half of North America.', 'Canada', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(11, 'Mexico, officially the United Mexican States, is a federal republic in the southern part of North America.', 'Mexico', '2010-1-1');
gaussdb=# CREATE INDEX pgweb_idx_1 ON tsearch.pgweb USING gin(to_tsvector('english', body));
```

The to_tsvector() function comes in to two versions: the 1-argument version and the 2-argument version. When the 1-argument version is used, the system uses the default configuration specified by **default_text_search_config**.

Notice that the 2-argument version of to_tsvector is used for index creation. Only full-text search functions that specify text search configuration can be correctly used in expression indexes. This is because the index contents must be unaffected by **default_text_search_config**, whose value can be changed at any time. If they were affected, the index contents might be inconsistent because different entries could contain tsvectors that were created with different text search configurations, and there would be no way to guess which was which. It would be impossible to dump and restore such an index correctly.

Because the 2-argument version of to_tsvector was used in the index above, only a query reference that uses the 2-argument version of to_tsvector with the same configuration will use that index. That is, **WHERE to_tsvector('english', body) @@ 'a & b'** can use the index, but **WHERE to_tsvector(body) @@ 'a & b'** cannot. This ensures that an index will be used only with the same tokenizer used to create the index entries.

It is possible to set up more complex expression indexes wherein the tokenizer name is specified by another column. For example:

```
gaussdb=# ALTER TABLE tsearch.pgweb ADD COLUMN config_name regconfig;
gaussdb=# CREATE INDEX pgweb_idx_2 ON tsearch.pgweb USING gin(to_tsvector('ngram', body));
```

This allows for mixed configurations within the same index, while recording the specific configuration used for each index entry. For example, this may be useful when the document collection includes documents in various languages. Similarly, queries that use an index must match an expression index, for example, **WHERE to_tsvector(config_name, body) @@ 'a & b'**.

Indexes can even concatenate columns:

```
gaussdb=# CREATE INDEX pgweb_idx_3 ON tsearch.pgweb USING gin(to_tsvector('english', title || ' ' || body));
```

Another approach is to create a separate **tsvector** column to hold the output of to_tsvector. This example is a concatenation of **title** and **body**, using COALESCE to ensure that one column will still be indexed when the other is **NULL**:

```
gaussdb=# ALTER TABLE tsearch.pgweb ADD COLUMN textsearchable_index_col tsvector;
gaussdb=# UPDATE tsearch.pgweb SET textsearchable_index_col = to_tsvector('english', coalesce(title,'') || ' ' || coalesce(body,''));
```

Then, create a GIN to speed up the search:

```
gaussdb=# CREATE INDEX textsearch_idx_4 ON tsearch.pgweb USING gin(textsearchable_index_col);
```

Now you are ready to perform a fast full-text search:

```
gaussdb=# SELECT title
FROM tsearch.pgweb
WHERE textsearchable_index_col @@ to_tsquery('north & america')
ORDER BY last_mod_date DESC
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
LIMIT 10;

 title
--------
 Canada
 Mexico
(2 rows)

-- Drop the table.
gaussdb=# DROP TABLE tsearch.pgweb;
gaussdb=# DROP SCHEMA tsearch;
```

One advantage of the separate-column approach over an expression index is that it is unnecessary to explicitly specify the text search tokenizer in queries in order to use the index. As shown in the preceding example, the query can depend on **default_text_search_config**. Another advantage is that searches will be faster, since it will not be necessary to call the to_tsvector again to verify index matches. The expression-index approach is simpler to set up, however, it requires less disk space since the tsvector representation is not stored explicitly.

NOTE

- GINs are supported only under Astore.
- For details about the restrictions on other GINs, see **CREATE INDEX**.

tsvector supports the creation of UGINs in Ustore.

```
-- Create a table.
gaussdb=# CREATE TABLE ustore_tbl (id int, content text, ts tsvector) WITH (storage_type = USTORE);

-- Create indexes.
gaussdb=# CREATE INDEX textsearch_ugin_idx1 ON ustore_tbl USING ugin(ts);
gaussdb=# CREATE INDEX textsearch_ugin_idx2 ON ustore_tbl USING ugin(to_tsvector('english', content));

-- Drop the table.
gaussdb=# DROP TABLE ustore_tbl;
```

NOTE

- UGINs are supported only under Ustore. Other aspects of UGINs are basically the same as those of GINs.
- For details about the restrictions on other UGINs, see **CREATE INDEX**.

1.12.3 Controlling Text Search

To implement full-text search, there must be a function to create a tsvector from a document and a tsquery from a user query. Also, we need to return results in a useful order, so we need a function that compares documents with respect to their relevance to the query. It is also important to be able to display the results nicely. All these functions can be used in GaussDB.

1.12.3.1 Parsing Documents

GaussDB provides the to_tsvector function for converting a document to the tsvector data type.

```
to_tsvector([ config regconfig, ] document text) returns tsvector
```

The to_tsvector function parses a textual document into tokens, reduces the tokens to lexemes, and returns a tsvector. The tsvector contains each lexeme and the position of the lexeme in the document. The document is processed according to the default or specified text search tokenizer. Here is a simple example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT to_tsvector('english', 'a fat  cat sat on a mat - it ate a fat rats');
                  to_tsvector
-----------------------------------------------------
 'ate':9 'cat':3 'fat':2,11 'mat':7 'rat':12 'sat':4
```

In the preceding example, the tsvector result does not contain the words **a**, **on**, or **it**, the word **rats** became **rat**, and the hyphen (-) was ignored.

The to_tsvector function internally calls a parser which breaks the document text into tokens and assigns a type to each token. For each token, a list of dictionaries is consulted. where the list can vary depending on the token type. The first dictionary that recognizes the token will produce one or more normalized lexemes to represent the token. Example:

- **rats** became **rat** because one of the dictionaries recognized that the word **rats** is a plural form of **rat**.
- Some words are recognized as stop words (see **Stop Words**), which causes them to be ignored since they occur too frequently to be useful in searching. In our example, these are **a**, **on**, and **it**.
- If no dictionary in the list recognizes the token, then it is also ignored. In this example that happened to the punctuation sign (-) because there are no dictionaries assigned for its token type (space symbols), meaning space tokens will never be indexed.

The choices of parser, dictionaries, and which types of tokens to index are determined by the selected text search configuration. It is possible to have many different configurations in the same database and predefined configurations in multiple languages. In the preceding example, **english** is used by default.

The setweight function can be used to label the entries of a tsvector with a given weight, where a weight is one of the letters A, B, C, or D, which is typically used to mark entries coming from different parts of a document, such as the title and body. Then, the search results can be sorted based on the weights of the parts of the document.

Because to_tsvector(NULL) will return **NULL**, you are advised to use COALESCE whenever a column might be null. Here is the recommended method for creating a tsvector from a structured document:

```
gaussdb=# CREATE TABLE tsearch.tt (id int, title text, keyword text, abstract text, body text, ti tsvector);

gaussdb=# INSERT INTO tsearch.tt(id, title, keyword, abstract, body) VALUES (1, 'America', 'New York',
'America','America is a rock band, formed in England in 1970 by multi-instrumentalists Dewey Bunnell, Dan
Peek, and Gerry Beckley.');

gaussdb=# UPDATE tsearch.tt SET ti =
    setweight(to_tsvector(coalesce(title,'')), 'A')    ||
    setweight(to_tsvector(coalesce(keyword,'')), 'B')  ||
    setweight(to_tsvector(coalesce(abstract,'')), 'C') ||
    setweight(to_tsvector(coalesce(body,'')), 'D');
gaussdb=# DROP TABLE tsearch.tt;
```

In this example, **setweight** is used to label the source of each lexeme in the finished tsvector, and then the labeled tsvector values are concatenated using the tsvector concatenation operator ||. For details about these operations, see **Manipulating tsvector**.

1.12.3.2 Parsing Queries

GaussDB provides functions to_tsquery, plainto_tsquery, and phraseto_tsquery for converting a query to the tsquery data type. The to_tsquery function offers access

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

to more features than plainto_tsquery and phraseto_tsquery, but is less forgiving about its input.

```
to_tsquery([ config regconfig, ] querytext text) returns tsquery
```

The to_tsquery function creates a tsquery value from querytext, which must consist of single tokens separated by the operators & (AND), | (OR), ! (NOT), and <-> or <n> (PHRASE). These operators can be grouped using parentheses. The input to to_tsquery must follow the general rules for tsquery input, as described in **Text Search Types**. The difference is that while basic tsquery input takes the tokens at face value, to_tsquery normalizes each token to a lexeme using the specified or default configuration, and discards any tokens that are stop words according to the configuration. For example:

```
gaussdb=# SELECT to_tsquery('english', 'The & Fat & Rats');
  to_tsquery
---------------
 'fat' & 'rat'
(1 row)
```

As in basic tsquery input, **weight(s)** can be attached to each lexeme to restrict it to match only tsvector lexemes of those **weight(s)**. For example:

```
gaussdb=# SELECT to_tsquery('english', 'Fat | Rats:AB');
    to_tsquery
------------------
 'fat' | 'rat':AB
(1 row)
```

Also, the asterisk (*) can be attached to a lexeme to specify prefix matching.

```
gaussdb=# SELECT to_tsquery('supern:*A & star:A*B');
        to_tsquery
--------------------------
 'supern':*A & 'star':*AB
(1 row)
```

Such a lexeme will match any word having the specified string and weight in a tsquery.

```
plainto_tsquery([ config regconfig, ] querytext text) returns tsquery
```

The plainto_tsquery function converts unformatted text querytext to tsquery. The text is parsed and normalized much as for to_tsvector, then the & (AND) Boolean operator is inserted between surviving words.

For example:

```
gaussdb=# SELECT plainto_tsquery('english', 'The Fat Rats');
 plainto_tsquery
-----------------
 'fat' & 'rat'
(1 row)
```

Note that plainto_tsquery cannot recognize operators, weight labels, or prefix-match labels in its input.

```
gaussdb=# SELECT plainto_tsquery('english', 'The Fat & Rats:C');
  plainto_tsquery
---------------------
 'fat' & 'rat' & 'c'
(1 row)
```

All the input punctuation was discarded as being space symbols.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Similar to plainto_tsquery, phraseto_tsquery converts unformatted text querytext to tsquery, with the connection operator <-> (PHRASE).

1.12.3.3 Ranking Search Results

Ranking attempts to measure how relevant documents are to a particular query, so that when there are many matches the most relevant ones can be shown first. GaussDB provides two predefined ranking functions. The functions take into account lexical, proximity, and structural information; that is, how often the search words appear in the document, how close together the words are in the document, and how important is the part of the document where the words appear. However, the concept of relevancy is vague and application-specific. Different applications might require additional information for ranking, for example, document modification time. The built-in ranking functions are only examples. You can write your own ranking functions and/or combine their results with additional factors to fit your specific needs.

The two ranking functions currently available are:

```
ts_rank([ weights float4[], ] vector tsvector, query tsquery [, normalization integer ]) returns float4
```

The ts_rank function ranks vectors based on the frequency of matching lexemes in the vector.

```
ts_rank_cd([ weights float4[], ] vector tsvector, query tsquery [, normalization integer ]) returns float4
```

The ts_rank_cd function calculates a cover density ranking for a given document vector and query. Cover density is similar to ts_rank ranking, except that it considers the proximity of matching lexemes to each other.

This function requires positional information in its input. When the matching token lacks position information, the result is **0**.

For both these functions, the optional **weights** parameter offers the ability to weigh word instances more or less heavily depending on how they are labeled. The weight arrays specify how heavily to weigh each category of words, in the order:

```
{D-weight, C-weight, B-weight, A-weight}
```

If no weights are provided, then the default is used: **{0.1, 0.2, 0.4, 1.0}**.

Typically weights are used to mark words from special areas of the document, like the title or an initial abstract, so they can be treated with more or less importance than words in the document body.

Since a longer document has a greater chance of containing a search word, it is reasonable to take into account document size. For example, a hundred-word document with five instances of a search word is probably more relevant than a thousand-word document with five instances. Both ranking functions take an integer normalization option that specifies whether and how a document's length should impact its rank. The integer option controls several behaviors, so it is a bit mask: you can specify one or more behaviors using a vertical bar (|) (for example, 2|4).

- **0** (default): ignores the document length.
- **1**: divides the rank by (1 + logarithm of the document length).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **2**: divides the rank by the document length.
- **4**: divides the rank by the mean harmonic distance between extents. This is implemented only by ts_rank_cd.
- **8** divides the rank by the number of unique words in document.
- **16** divides the rank by (1 + logarithm of the number of unique words in document).
- **32** divides the rank by (itself + 1).

If more than one flag bit is specified, the transformations are applied in the order listed.

It is important to note that the ranking functions do not use any global information, so it is impossible to produce a fair normalization to 1% or 100% as sometimes desired. Normalization option 32 (rank/(rank + 1)) can be applied to scale all ranks into the range zero to one, but this is just a cosmetic change; it will not affect the ordering of the search results.

The following is an example that selects only the top 10 matches:

```
gaussdb=# CREATE SCHEMA tsearch;
gaussdb=# CREATE TABLE tsearch.pgweb(id int, body text, title text, last_mod_date date) with(storage_type = ASTORE);
gaussdb=# INSERT INTO tsearch.pgweb VALUES(1, 'America is a rock band, formed in England in 1970 by multi-instrumentalists Dewey Bunnell, Dan Peek, and Gerry Beckley.', 'America', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(2, 'England is a country that is part of the United Kingdom. It shares land borders with Scotland to the north and Wales to the west.', 'England', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(3, 'Australia, officially the Commonwealth of Australia, is a country comprising the mainland of the Australian continent, the island of Tasmania, and numerous smaller islands.', 'Australia', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(4, 'Japan is an island country in East Asia.', 'Japan', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(5, 'Germany, officially the Federal Republic of Germany, is a sovereign state and federal parliamentary republic in central-western Europe.', 'Germany', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(6, 'France, is a sovereign state comprising territory in western Europe and several overseas regions and territories.', 'France', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(7, 'Italy officially the Italian Republic, is a unitary parliamentary republic in Europe.', 'Italy', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(8, 'India, officially the Republic of India, is a country in South Asia.', 'India', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(9, 'Brazil, officially the Federative Republic of Brazil, is the largest country in both South America and Latin America.', 'Brazil', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(10, 'Canada is a country in the northern half of North America.', 'Canada', '2010-1-1');
gaussdb=# INSERT INTO tsearch.pgweb VALUES(11, 'Mexico, officially the United Mexican States, is a federal republic in the southern part of North America.', 'Mexico', '2010-1-1');

gaussdb=# SELECT id, title, ts_rank_cd(to_tsvector(body), query) AS rank
FROM tsearch.pgweb, to_tsquery('america') query
WHERE query @@ to_tsvector(body)
ORDER BY rank DESC
LIMIT 10;
 id |  title  | rank
----+---------+------
 11 | Brazil  |   .2
  2 | America |   .1
 12 | Canada  |   .1
 13 | Mexico  |   .1
(4 rows)
```

The following is the same example using normalized ranking:

```
gaussdb=# SELECT id, title, ts_rank_cd(to_tsvector(body), query, 32 /* rank/(rank+1) */ ) AS rank
FROM tsearch.pgweb, to_tsquery('america') query
WHERE  query @@ to_tsvector(body)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ORDER BY rank DESC
LIMIT 10;
 id |  title  |   rank
----+---------+----------
 11 | Brazil  |  .166667
  2 | America | .0909091
 12 | Canada  | .0909091
 13 | Mexico  | .0909091
(4 rows)

-- Drop the table.
gaussdb=# DROP TABLE tsearch.pgweb;
gaussdb=# DROP SCHEMA tsearch;
```

Ranking can be expensive since it requires consulting the tsvector of each matching document, which can be subject to I/O bound and therefore slow.

1.12.3.4 Highlighting Search Results

To present search results, it is ideal to show a part of each document and how it is related to the query. Typically, the search engine highlights the search words within the document snippets displayed. GaussDB provides the ts_headline function to implement this functionality.

```
ts_headline([ config regconfig, ] document text, query tsquery [, options text ]) returns text
```

ts_headline accepts a document and a query, and returns a snippet from the document where terms from the query are highlighted. The configuration used to parse the document can be specified by **config**. If **config** is omitted, the configuration specified by **default_text_search_config** is used.

options indicates optional configurations, consisting of one or more option=value pairs, which must be separated by commas. The available options are:

- **StartSel**, **StopSel**: The strings with which to delimit search words appearing in the document, to distinguish them from other excerpted words. You must double-quote these strings if they contain spaces or commas. The maximum length is 32767 bytes.
- **MaxWords**, **MinWords**: These numbers determine the longest and shortest excerpted words to output.
- **ShortWord**: Words of this length or less will be dropped at the start and end of an excerpt. The default value **3** eliminates common English articles.
- **HighlightAll**: Boolean flag. If **true** is returned, the whole document will be used as an excerpt, ignoring the preceding three parameters.
- **MaxFragments**: Maximum number of text excerpts or fragments to display. The default value **0** selects a non-fragmented excerpt generation method. A value greater than zero selects fragmented excerpt generation. This method finds text fragments with as many search words as possible and stretches those fragments around the search words. As a result, search words are close to the middle of each fragment and have words on each side. Each fragment contains a maximum of words specified by **MaxWords** and words of a length specified by **ShortWord** or less are dropped at the start and end of each fragment. If not all search words are found in the document, then a single fragment specified by **MinWords** will be displayed at the beginning of the document.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **FragmentDelimiter**: When more than one fragment is displayed, the fragments will be separated by this string. The maximum length is 32767 bytes.

Any unspecified options receive these defaults:

```
StartSel=<b>, StopSel=</b>,
MaxWords=35, MinWords=15, ShortWord=3, HighlightAll=FALSE,
MaxFragments=0, FragmentDelimiter=" ... "
```

For example:

```
gaussdb=# SELECT ts_headline('english',
'The most common type of search
is to find all documents containing given query terms
and return them in order of their similarity to the
query.',
to_tsquery('english', 'query & similarity'));
                      ts_headline
------------------------------------------------------------
 containing given <b>query</b> terms
 and return them in order of their <b>similarity</b> to the
 <b>query</b>.
(1 row)

gaussdb=# SELECT ts_headline('english',
'The most common type of search
is to find all documents containing given query terms
and return them in order of their similarity to the
query.',
to_tsquery('english', 'query & similarity'),
'StartSel = <, StopSel = >');
                      ts_headline
------------------------------------------------------------
 containing given <query> terms
 and return them in order of their <similarity> to the
 <query>.
(1 row)
```

The ts_headline function uses the original document, not a tsvector excerpt, so it may be slow and should be used with care.

1.12.4 Additional Features

1.12.4.1 Manipulating tsvector

GaussDB provides functions and operators that can be used to manipulate documents that are already in tsvector type.

- tsvector || tsvector

  The tsvector concatenation operator returns a new tsvector which combines the lexemes and positional information of the two tsvectors given as arguments. Positions and weight labels are retained during the concatenation. Positions appearing in the right-hand tsvector are offset by the largest position mentioned in the left-hand tsvector, so that the result is nearly equivalent to the result of performing to_tsvector on the concatenation of the two original document strings. (The equivalence is not exact because any stop words removed from the end of the left-hand argument will not affect the result, whereas they would have affected the positions of the lexemes in the right-hand argument if textual concatenation were used.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

One advantage of using concatenation in the tsvector form, rather than concatenating text before applying to_tsvector, is that you can use different tokenizers to parse different sections of the document. Also, because the setweight function marks all lexemes of the given tsvector in the same way, it is necessary to parse the text and do setweight before concatenating if you want to label different parts of the document with different weights.

- setweight(vector tsvector, weight "char") returns tsvector

  The setweight function returns a copy of the input tsvector in which every position has been labeled with the given weight, either A, B, C, or D. (D is the default for new tsvectors and as such is not displayed on output.) These labels are retained when tsvectors are concatenated, allowing words from different parts of a document to be weighted differently by ranking functions.

  NOTICE

  Note that weight labels apply to positions, not lexemes. If the input tsvector has been stripped of positions, then setweight does not take effect.

- length(vector tsvector) returns integer

  Returns the number of lexemes stored in the vector.

- strip(vector tsvector) returns tsvector

  Removes position and weight information from all lexemes in the input tsvector, leaving only the distinct lexemes, and returns a new tsvector.

1.12.4.2 Manipulating Queries

GaussDB provides functions and operators that can be used to manipulate queries that are already in tsquery type.

- tsquery && tsquery

  Returns the AND-combination of the two given queries.

- tsquery || tsquery

  Returns the OR-combination of the two given tsqueries.

- !! tsquery

  Returns the negation (NOT) of the given tsquery.

- tsquery <-> tsquery

  Returns the phrase result of the two given tsqueries.

- numnode(query tsquery) returns integer

  Returns the number of nodes (lexemes plus operators) in a tsquery. This function is useful to determine if the query is meaningful (return result > 0), or contains only stop words (returns 0). For example:

```
gaussdb=# SELECT numnode(plainto_tsquery('the any'));
NOTICE:  text-search query contains only stop words or doesn't contain lexemes, ignored
CONTEXT:  referenced column: numnode
 numnode
---------
       0

gaussdb=# SELECT numnode('foo & bar'::tsquery);
 numnode
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
---------
       3
```

- querytree(query tsquery) returns text

  Returns the portion of a tsquery that can be searched with an index. This function is useful for detecting unindexable queries, for example those containing only stop words or only negated terms. For example:

```
gaussdb=# SELECT querytree(to_tsquery('!defined'));
 querytree
-----------
 T
(1 row)
```

1.12.4.3 Rewriting Queries

Functions in the ts_rewrite family search a given tsquery for occurrences of a target subquery, and replace each occurrence with a substitute subquery. In essence, this operation is a tsquery-specific version by substring replacement. A target and substitute query combination can be considered as a query rewriting rule. A collection of such rewriting rules can be a powerful search aid. For example, you can expand the search scope using synonyms (such as, new york, big apple, nyc, and gotham) or narrow down the search scope.

- ts_rewrite (query tsquery, target tsquery, substitute tsquery) returns tsquery

  This form of ts_rewrite simply applies a single rewriting rule: the target query is replaced by the substitute query wherever it appears. For example:

```
gaussdb=# SELECT ts_rewrite('a & b'::tsquery, 'a'::tsquery, 'c'::tsquery);
 ts_rewrite
------------
 'b' & 'c'
```

- ts_rewrite (query tsquery, select text) returns tsquery

  This form of ts_rewrite accepts a starting query and an SQL **SELECT** command, which is given as a text string. The **SELECT** command must yield two columns of the tsquery type. For each row of the query result, occurrences of the first column value (the target) are replaced by the second column value (the substitute) within the current query.

  NOTE

  Note that when multiple rewriting rules are applied in this way, the order of rewriting can be important; therefore in practice, perform ORDER BY on the original query based on some columns.

  Consider a real-life astronomical example. Use table-driven rewriting rules to expand the search scope of a supernovae.

```
gaussdb=# CREATE TABLE tsearch.aliases (id int, t tsquery, s tsquery);

gaussdb=# INSERT INTO tsearch.aliases VALUES(1, to_tsquery('supernovae'), to_tsquery('supernovae|
sn'));

gaussdb=# SELECT ts_rewrite(to_tsquery('supernovae & crab'), 'SELECT t, s FROM tsearch.aliases');
          ts_rewrite
---------------------------------
 'crab' & ( 'supernova' | 'sn' )
```

  You can change the rewriting rules just by updating the table.

```
gaussdb=# UPDATE tsearch.aliases
SET s = to_tsquery('supernovae|sn & !nebulae')
WHERE t = to_tsquery('supernovae');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT ts_rewrite(to_tsquery('supernovae & crab'), 'SELECT t, s FROM tsearch.aliases');
               ts_rewrite
--------------------------------------------
 'crab' & ( 'supernova' | 'sn' & !'nebula' )
```

Rewriting can be slow when there are many rewriting rules, since it checks every rule for a possible match. To filter out obvious non-candidate rules, we can use operators of the tsquery type. In the example below, only those rules which might match the original query are selected:

```
gaussdb=# SELECT ts_rewrite('a & b'::tsquery, 'SELECT t,s FROM tsearch.aliases WHERE ''a &
b''::tsquery @> t');
 ts_rewrite
------------
 'b' & 'a'
(1 row)
gaussdb=# DROP TABLE tsearch.aliases;
```

1.12.4.4 Collecting Document Statistics

The ts_stat function is useful for checking your configuration and for finding stop-word candidates.

```
ts_stat(sqlquery text, [ weights text, ]
       OUT word text, OUT ndoc integer,
       OUT nentry integer) returns setof record
```

sqlquery is a text value containing an SQL query which must return a single tsvector column. ts_stat executes the query and returns statistics about each distinct lexeme (word) contained in the tsvector data. The columns returned are:

- **word text**: a lexeme.
- **ndoc integer**: number of a lexeme in a tsvector (document).
- **nentry integer**: total number of occurrences of a lexeme.

If **weights** is specified, only occurrences having one of those weights are counted. For example, to find the ten most frequent words in a document collection:

```
gaussdb=# SET tsearch_options = 'debug_function';
gaussdb=# CREATE SCHEMA tpcds;
gaussdb=# CREATE TABLE tpcds.store_returns (
    sr_reason_sk     integer,
    sr_reason_id     character(16),
    sr_reason_desc   character(100),
    sr_customer_sk   integer);
gaussdb=# INSERT INTO tpcds.store_returns VALUES(1, 'test_001', 'test_desc_001', 5), (12, 'test_002',
'test_desc_002', 8), (23, 'test_003', 'test_desc_003', 9), (14, 'test_004', 'test_desc_004', 10), (45, 'test_005',
'test_desc_005', 12), (55, 'test_006', 'test_desc_006', 1), (43, 'test_007', 'test_desc_007', 12);
gaussdb=# SELECT * FROM ts_stat('SELECT to_tsvector(''english'', sr_reason_sk) FROM tpcds.store_returns
WHERE sr_customer_sk < 10') ORDER BY nentry DESC, ndoc DESC, word LIMIT 10;
 word | ndoc | nentry
------+------+--------
 1    |    1 |      1
 12   |    1 |      1
 23   |    1 |      1
 55   |    1 |      1
(4 rows)
```

The same, but counting only word occurrences with weight A or B:

```
gaussdb=# SELECT * FROM ts_stat('SELECT to_tsvector(''english'', sr_reason_sk) FROM tpcds.store_returns
WHERE sr_customer_sk < 10', 'a') ORDER BY nentry DESC, ndoc DESC, word LIMIT 10;
 word | ndoc | nentry
------+------+--------
(0 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop the table.
gaussdb=# DROP TABLE tpcds.store_returns;
gaussdb=# DROP SCHEMA tpcds;
gaussdb=# SET tsearch_options = '';
```

1.12.5 Parsers

A text search parser breaks raw document text into tokens and recognizes each token's type, where the set of types is defined by the parser itself. Note that a parser does not modify the text. It simply identifies plausible word boundaries. Because of this limited scope, there is less need for application-specific custom parsers than there is for custom dictionaries.

Currently, GaussDB provides the following built-in parsers: pg_catalog.default for English tokenization, and pg_catalog.ngram and pg_catalog.pound for full-text search in texts containing Chinese, or both Chinese and English.

The built-in parser pg_catalog.default can recognize 23 token types, as shown in **Table 1-337**.

Table 1-337 Default parser's token types

Token Type	Description	Example
asciiword	Word, all ASCII letters	elephant
word	Word, all letters	mañana
numword	Word, letters and digits	beta1
asciihword	Hyphenated word, all ASCII	up-to-date
hword	Hyphenated word, all letters	lógico-matemática
numhword	Hyphenated word, letters and digits	test-beta1
hword_asciipart	Hyphenated word part, all ASCII	test in the context test-beta1
hword_part	Hyphenated word part, all letters	lógico or matemática in the context lógico-matemática
hword_numpart	Hyphenated word part, letters and digits	beta1 in the context test-beta1
email	Email address	foo@example.com
protocol	Protocol head	http://
url	URL	example.com/stuff/index.html
host	Host	example.com
url_path	URL path	/stuff/index.html, in the context of a URL

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Token Type	Description	Example
file	File or path name	/usr/local/foo.txt, if not within a URL
sfloat	Scientific notation	-1.23E+56
float	Decimal notation	-1.234
int	Signed integer	-1234
uint	Unsigned integer	1234
version	Version number	8.3.0
tag	XML tag	<a href="dictionaries.html">
entity	XML entity	&
blank	Space symbols	(any whitespace or punctuation not otherwise recognized)

Note: The parser's notion of a "letter" is determined by the database's locale setting, specifically **lc_ctype**. Words containing only the basic ASCII letters are reported as a separate token type, since it is sometimes useful to distinguish them. In most European languages, token types word and asciiword should be treated alike.

The email type does not support all valid email characters as defined by RFC 5322. Specifically, the only non-alphanumeric characters supported for email usernames are period, dash, and underscore.

It is possible for the parser to recognize overlapping tokens in the same piece of text. For example, a hyphenated word will be reported both as the entire word and as each component.

```
gaussdb=# SELECT alias, description, token FROM ts_debug('english','foo-bar-beta1');
      alias      |              description               |     token
-----------------+----------------------------------------+----------------
 numhword        | Hyphenated word, letters and digits    | foo-bar-beta1
 hword_asciipart | Hyphenated word part, all ASCII        | foo
 blank           | Space symbols                          | -
 hword_asciipart | Hyphenated word part, all ASCII        | bar
 blank           | Space symbols                          | -
 hword_numpart   | Hyphenated word part, letters and digits | beta1
```

This behavior is desirable since it allows searches to work for both the whole compound word and for components. Here is another example:

```
gaussdb=# SELECT alias, description, token FROM ts_debug('english','http://example.com/stuff/index.html');
  alias   |  description  |            token
----------+---------------+------------------------------
 protocol | Protocol head | http://
 url      | URL           | example.com/stuff/index.html
 host     | Host          | example.com
 url_path | URL path      | /stuff/index.html
```

N-gram is a mechanical word segmentation method, and applies to non-semantic Chinese word segmentation scenarios. The N-gram segmentation method ensures

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

the completeness of tokens. However, to cover all the possibilities, it adds unnecessary words to the index, resulting in a large number of index items. N-gram supports Chinese encoding, including GBK and UTF-8. **Table 1-338** lists six built-in token types.

Table 1-338 Token types

Alias	Description
zh_words	chinese words
en_word	english word
numeric	numeric data
alnum	alnum string
grapsymbol	graphic symbol
multisymbol	multiple symbol

Pound segments words in a fixed format. It is used to segment to-be-parsed nonsense Chinese and English words that are separated by fixed delimiters. It supports Chinese encoding (including GBK and UTF-8) and English encoding (including ASCII). Pound has six built-in token types (as listed in **Table 1-339**) and supports five delimiters (as listed in **Table 1-340**). The default delimiter is the pound key (#). For Pound, the maximum length of a token is 256 characters.

Table 1-339 Token types

Alias	Description
zh_words	chinese words
en_word	english word
numeric	numeric data
alnum	alnum string
grapsymbol	graphic symbol
multisymbol	multiple symbol

Table 1-340 Delimiter types

Delimiter	Description
@	Special character
#	Special character
$	Special character

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Delimiter	Description
%	Special character
/	Special character

1.12.6 Dictionaries

1.12.6.1 Overview

A dictionary is used to define stop words, that is, words to be ignored in full-text search.

A dictionary can also be used to normalize words so that different derived forms of the same word will match. A normalized word is called a lexeme.

In addition to improving retrieval quality, normalization and removal of stop words can reduce the size of the tsvector representation of a document, thereby improving performance. Normalization and removal of stop words do not always have linguistic meaning. Users can define normalization and removal rules in dictionary definition files based on application environments.

A dictionary is a program that receives a token as input and returns:

- An array of lexemes if the input token is known to the dictionary (note that one token can produce more than one lexeme).
- A lexeme. (The lexeme has the TSL_FILTER flag set. The TSL_FILTER flag is automatically set in a filtering dictionary and is not perceived by users.) A new token will replace the original one and is passed to the subsequent dictionary. (The current dictionary may be referred to as a filtering dictionary.)
- An empty array if the input token is known to the dictionary but is a stop word.
- **NULL** if the dictionary does not recognize the token.

GaussDB provides predefined dictionaries for many languages and also provides the predefined dictionary templates Simple and Snowball. These templates can be used to create new dictionaries with custom parameters.

When using full-text search, you are advised to:

- In the text search configuration, configure a parser together with a set of dictionaries to process the parser's output tokens. For each token type that the parser can return, a separate list of dictionaries is specified by the configuration. When a token of that type is found by the parser, each dictionary in the list is consulted in turn, until a dictionary recognizes it as a known word. If it is identified as a stop word, or no dictionary recognizes the token, it will be discarded and not indexed or searched for. Generally, the first dictionary that returns a non-**NULL** output determines the result, and any remaining dictionaries are not consulted. However, a filtering dictionary can replace the input token with a modified one, which is then passed to subsequent dictionaries.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The general rule for configuring a list of dictionaries is to place first the most narrow, most specific dictionary, then the more general dictionaries, finishing with a very general dictionary, like the Snowball stemmer dictionary or the Simple dictionary, which recognizes everything.

1.12.6.2 Stop Words

Stop words are words that are very common, appear in almost every document, and have no discrimination value. Therefore, they can be ignored in the context of full-text search. Each type of dictionaries treats stop words in different ways. For example, Ispell dictionaries first normalize words and then check the list of stop words, while Snowball dictionaries first check the list of stop words.

Every English text contains words like "a" and "the", so it is useless to store them in an index. However, stop words affect the positions in tsvector, which in turn affect ranking.

```
gaussdb=# SELECT to_tsvector('english','in the list of stop words');
        to_tsvector
----------------------------
 'list':3 'stop':5 'word':6
```

Words in positions 1, 2, and 4 are stop words and are not displayed. Ranks calculated for documents with and without stop words are quite different:

```
gaussdb=# SELECT ts_rank_cd (to_tsvector('english','in the list of stop words'), to_tsquery('list & stop'));
 ts_rank_cd
------------
        .05

gaussdb=# SELECT ts_rank_cd (to_tsvector('english','list stop words'), to_tsquery('list & stop'));
 ts_rank_cd
------------
         .1
```

1.12.6.3 Simple Dictionary

A Simple dictionary operates by converting the input token to lower case and checking it against a list of stop words. If the token is found in the list, an empty array will be returned, causing the token to be discarded. If it is not found, the lower-cased form of the word is returned as a normalized lexeme. In addition, you can set **Accept** (default: **true**) to **false** for Simple dictionaries to report non-stop-words as unrecognized, allowing them to be passed on to the next dictionary in the list.

Precautions

- Most types of dictionaries rely on dictionary configuration files. The name of a configuration file can only be lowercase letters, digits, and underscores (_).
- A dictionary cannot be created in the pg_temp schema.
- Dictionary configuration files must be stored in UTF-8 encoding. They will be translated to the actual database encoding, if that is different, when they are read into the server.
- Generally, a session will read a dictionary configuration file only once, when it is first used within the session. To modify a configuration file, run the **ALTER TEXT SEARCH DICTIONARY** statement to update and reload the file.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Procedure

Step 1 Create a Simple dictionary.

```
gaussdb=# CREATE TEXT SEARCH DICTIONARY public.simple_dict (
    TEMPLATE = pg_catalog.simple,
    STOPWORDS = english
);
```

english.stop is the full name of the stop word file. For details about the syntax and parameters for creating a Simple dictionary, see **CREATE TEXT SEARCH DICTIONARY**.

Step 2 Use the Simple dictionary.

```
gaussdb=# SELECT ts_lexize('public.simple_dict','YeS');
 ts_lexize
-----------
 {yes}
(1 row)

gaussdb=# SELECT ts_lexize('public.simple_dict','The');
 ts_lexize
-----------
 {}
(1 row)
```

Step 3 Set **Accept** to **false** so that the Simple dictionary returns **NULL** instead of a lower-cased non-stop word.

```
gaussdb=# ALTER TEXT SEARCH DICTIONARY public.simple_dict ( Accept = false );
ALTER TEXT SEARCH DICTIONARY
gaussdb=# SELECT ts_lexize('public.simple_dict','YeS');
 ts_lexize
-----------

(1 row)

gaussdb=# SELECT ts_lexize('public.simple_dict','The');
 ts_lexize
-----------
 {}
(1 row)

gaussdb=# DROP TEXT SEARCH DICTIONARY public.simple_dict;
```

----End

1.12.6.4 Snowball Dictionary

A Snowball dictionary is based on a project by Martin Porter and is used for stem analysis, providing stemming algorithms for many languages. GaussDB provides predefined Snowball dictionaries for many languages. You can query the PG_TS_DICT system catalog to view the predefined Snowball dictionaries and supported stemming algorithms.

A Snowball dictionary recognizes everything, no matter whether it is able to simplify the word. Therefore, it should be placed at the end of the dictionary list. It is not recommended that you place it before any other dictionary because a token will never pass it through to the next dictionary.

For details about the syntax of Snowball dictionaries, see **CREATE TEXT SEARCH DICTIONARY**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.12.7 Configuration Examples

Text search configuration specifies the following components required for converting a document into a tsvector:

- A parser, used for breaking a text into tokens.
- A list of dictionaries, used for converting each token into a lexeme.

Each time when the to_tsvector or to_tsquery function is called, a text search configuration is required to specify a processing procedure. The GUC parameter **default_text_search_config** specifies the default text search configuration, which will be used if the text search function does not explicitly specify a text search configuration.

GaussDB provides some predefined text search configurations. You can also create user-defined text search configurations. In addition, to facilitate the management of text search objects, multiple gsql meta-commands are provided to display information about text search objects. (For details, see **Meta-Command Reference**.)

Procedure

Step 1 Set the default text search configuration used by and valid only in the current session. The current full-text search configuration can be set to **ngram**, **pound**, **simple**, or **english**.

```
gaussdb=# set default_text_search_config = 'ngram';
SET
```

----End

1.12.8 Testing and Debugging Text Search

NOTICE

This function is for internal use only. You are advised not to use it.

The behavior of a custom text search tokenizer can easily become confusing. The functions described in this section are useful for testing text search objects. You can test a complete tokenizer, or test parsers and dictionaries separately.

1.12.8.1 Testing a Tokenizer

The ts_debug function allows easy testing of a text search tokenizer.

```
ts_debug([ config regconfig, ] document text,
        OUT alias text,
        OUT description text,
        OUT token text,
        OUT dictionaries regdictionary[],
        OUT dictionary regdictionary,
        OUT lexemes text[])
        returns setof record
```

ts_debug displays information about every token of a document as produced by the parser and processed by the configured dictionaries. It uses the tokenizer specified by **config**, or **default_text_search_config** if that argument is omitted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

ts_debug returns one row for each token identified in the text by the parser. The columns returned are:

- **alias**: short name of the token, which is of the text type.
- **description**: token description, which is of the text type.
- **token**: token text, which is of the text type.
- **dictionaries**: dictionaries selected for tokens, which are of the regdictionary type.
- **dictionary**: dictionary that recognizes tokens, which is of the regdictionary type. If this parameter is left empty, tokens will not be recognized.
- **lexemes**: lexeme produced by the dictionary that recognizes tokens, which is of the text array type. If this parameter is left empty, no lexeme will be produced. An empty array ({}) means that the token will be recognized as a stop word.

Here is a simple example:

```
gaussdb=# SELECT * FROM ts_debug('english','a fat  cat sat on a mat - it ate a fat rats');
   alias   |   description   | token |  dictionaries  |  dictionary  | lexemes
-----------+-----------------+-------+----------------+--------------+---------
 asciiword | Word, all ASCII | a     | {english_stem} | english_stem | {}
 blank     | Space symbols   |       | {}             |              |
 asciiword | Word, all ASCII | fat   | {english_stem} | english_stem | {fat}
 blank     | Space symbols   |       | {}             |              |
 asciiword | Word, all ASCII | cat   | {english_stem} | english_stem | {cat}
 blank     | Space symbols   |       | {}             |              |
 asciiword | Word, all ASCII | sat   | {english_stem} | english_stem | {sat}
 blank     | Space symbols   |       | {}             |              |
 asciiword | Word, all ASCII | on    | {english_stem} | english_stem | {}
 blank     | Space symbols   |       | {}             |              |
 asciiword | Word, all ASCII | a     | {english_stem} | english_stem | {}
 blank     | Space symbols   |       | {}             |              |
 asciiword | Word, all ASCII | mat   | {english_stem} | english_stem | {mat}
 blank     | Space symbols   |       | {}             |              |
 blank     | Space symbols   | -     | {}             |              |
 asciiword | Word, all ASCII | it    | {english_stem} | english_stem | {}
 blank     | Space symbols   |       | {}             |              |
 asciiword | Word, all ASCII | ate   | {english_stem} | english_stem | {ate}
 blank     | Space symbols   |       | {}             |              |
 asciiword | Word, all ASCII | a     | {english_stem} | english_stem | {}
 blank     | Space symbols   |       | {}             |              |
 asciiword | Word, all ASCII | fat   | {english_stem} | english_stem | {fat}
 blank     | Space symbols   |       | {}             |              |
 asciiword | Word, all ASCII | rats  | {english_stem} | english_stem | {rat}
(24 rows)
```

1.12.8.2 Testing a Parser

The ts_parse function allows direct testing of a text search parser.

```
ts_parse(parser_name text, document text,
        OUT tokid integer, OUT token text) returns setof record
```

ts_parse parses the given document and returns a series of records, one for each token produced by parsing. Each record includes a **tokid** showing the assigned token type and a **token** which is the text of the token. For example:

```
gaussdb=# SELECT * FROM ts_parse('default', '123 - a number');
 tokid | token
-------+--------
    22 | 123
    12 |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
   12 | -
    1 | a
   12 |
    1 | number
(6 rows)
```

The ts_token_type function returns the token type and description of the specified parser.

```
ts_token_type(parser_name text, OUT tokid integer,
              OUT alias text, OUT description text) returns setof record
```

ts_token_type returns a table which describes each type of token the specified parser can recognize. For each token type, the table gives the integer **tokid** that the parser uses to label a token of that type, the **alias** that names the token type in tokenizer commands, and a short description. For example:

```
gaussdb=# SELECT * FROM ts_token_type('default');
 tokid |      alias      |                description
-------+-----------------+-------------------------------------------
     1 | asciiword       | Word, all ASCII
     2 | word            | Word, all letters
     3 | numword         | Word, letters and digits
     4 | email           | Email address
     5 | url             | URL
     6 | host            | Host
     7 | sfloat          | Scientific notation
     8 | version         | Version number
     9 | hword_numpart   | Hyphenated word part, letters and digits
    10 | hword_part      | Hyphenated word part, all letters
    11 | hword_asciipart | Hyphenated word part, all ASCII
    12 | blank           | Space symbols
    13 | tag             | XML tag
    14 | protocol        | Protocol head
    15 | numhword        | Hyphenated word, letters and digits
    16 | asciihword      | Hyphenated word, all ASCII
    17 | hword           | Hyphenated word, all letters
    18 | url_path        | URL path
    19 | file            | File or path name
    20 | float           | Decimal notation
    21 | int             | Signed integer
    22 | uint            | Unsigned integer
    23 | entity          | XML entity
(23 rows)
```

1.12.8.3 Testing a Dictionary

The ts_lexize function facilitates dictionary testing.

ts_lexize(dict regdictionary, token text) returns text[]. ts_lexize returns an array of lexemes if the input token is known to the dictionary, or an empty array if the token is known to the dictionary but it is a stop word, or **NULL** if it is an unknown word.

For example:

```
gaussdb=# SELECT ts_lexize('english_stem', 'stars');
 ts_lexize
-----------
 {star}

gaussdb=# SELECT ts_lexize('english_stem', 'a');
 ts_lexize
-----------
 {}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

The ts_lexize function expects a single token, not text.

1.12.9 Restrictions

GaussDB's full-text search feature has the following restrictions:

- The length of each tsvector lexeme must be less than 2047 bytes.
- The length of a tsvector (lexemes+positions) must be less than 1 MB.
- Position values in tsvector must be greater than 0 and less than or equal to 16383. Values greater than 16383 are forcibly set to **16383**.
- Less than or equal to 255 positions per tsvector lexeme. Excessive positions, if any, will be discarded.
- The total number of keywords and operators in a tsquery must be less than 32768.
- The accuracy of full-text search is related to the parser and dictionary list in the text search configuration. In Chinese scenarios, you are advised to use ngram or pound as the text search configuration item. In other scenarios, you are advised to use english or simple as the text search configuration item.
- ILM policies can be added or modified for data objects only in the tsvector or tsquery function of **Row Expression Function Whitelist**.
- The Tokenweight dictionary function is used internally by the BM25 index. You are advised not to use it in the full-text search feature because it may contain unknown risks. Use other dictionary functions instead.
- The distance specified by the phrase operator (PHRASE) must be greater than or equal to 0 and less than or equal to 16384. Otherwise, an error is reported.

1.13 SQL Syntax

1.13.1 SQL Syntax

Table 1-341 SQL syntax

Format	Description
[ ]	The part enclosed in brackets ([]) is optional.
...	Preceding elements can appear repeatedly.
[x | y | ...]	One item is selected from two or more options or no item is selected.
{ x | y | ... }	One item is selected from two or more options.
[x | y | ...] [...]	Multiple parameters or no parameter can be selected. If multiple parameters are selected, separate them with spaces.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Format	Description
[x | y | ...] [,...]	Multiple parameters or no parameter can be selected. If multiple parameters are selected, separate them with commas (,).
{ x | y | ... } [...]	At least one parameter can be selected. If multiple parameters are selected, separate them with spaces.
{ x | y | ... } [,...]	At least one parameter can be selected. If multiple parameters are selected, separate them with commas (,).

1.13.2 DCL Syntax Overview

Data control language (DCL) is used to define and control users' access permissions on database objects (such as tables, views, and stored procedures) and database instances.

Granting Permissions

GaussDB provides a statement for granting permissions to data objects and roles. For details, see **GRANT**.

Revoking Permissions

GaussDB provides a statement for revoking permissions. For details, see **REVOKE**.

Setting Default Permissions

GaussDB allows you to set permissions for objects that will be created in the future. For details, see **ALTER DEFAULT PRIVILEGES**.

Changing Owners

GaussDB provides a statement for changing the owner of a database object. For details, see **REASSIGN OWNED**.

Defining a Role

Roles are used to manage permissions in a database. For database security purposes, management and operation permissions should be assigned to different roles. **Table 1-342** lists the involved SQL statements.

Table 1-342 SQL statements for defining a role

Description	SQL Statement
Creating a role	**CREATE ROLE**
Altering role attributes	**ALTER ROLE**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	SQL Statement
Dropping a role	DROP ROLE

Defining a User

In database management, a user is an identity credential for accessing the database system. By granting permissions to users, you can precisely control their access to database objects and the database instance itself, thereby implementing refined management of database access. **Table 1-343** lists the involved SQL statements.

Table 1-343 SQL statements for defining a user

Description	SQL Statement
Creating a user	CREATE USER
Altering user attributes	ALTER USER
Dropping a user	DROP USER

Modifying, Displaying, and Restoring a GUC Parameter

Table 1-344 SQL statements for modifying, displaying, and restoring a GUC parameter

Description	SQL Statement
Modifying a GUC parameter	SET
Altering a session	ALTER SESSION
Showing the current value of a GUC parameter	SHOW
Restoring a GUC parameter to the default value	RESET

1.13.3 DDL Syntax Overview

Data definition language (DDL) is a standard SQL language used to define and alter database objects. It mainly includes statements such as CREATE, ALTER, and DROP, which are used to create, alter, and drop database objects, such as tables, indexes, views, constraints, and data type definitions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> GaussDB does not support DDL operations when the CN is incomplete. For example, if a CN in cluster is faulty, creating a database or a table will fail.

Dropping Database Objects Owned by a Database Role

GaussDB provides a statement for dropping database objects owned by a database role. For details, see **DROP OWNED**.

Defining a CMK

A client master key (CMK) is mainly used to encrypt column encryption keys (CEKs) in the encrypted database feature. Defining a CMK includes creating and dropping a CMK. **Table 1-345** lists the involved SQL statements.

Table 1-345 SQL statements for defining a CMK

Description	SQL Statement
Creating a CMK	**CREATE CLIENT MASTER KEY**
Dropping a CMK	**DROP CLIENT MASTER KEY**

Defining a Column Encryption Key (CEK)

CEKs are mainly used to encrypt data in the encrypted database feature. Defining a CEK includes creating a CEK, rotating the CMK for encrypting the CEK, and dropping a CEK. **Table 1-346** lists the involved SQL statements.

Table 1-346 SQL statements for defining a CEK

Description	SQL Statement
Creating a CEK	**CREATE COLUMN ENCRYPTION KEY**
Changing the client master key specified by a CEK	**ALTER COLUMN ENCRYPTION KEY**
Dropping a CEK	**DROP COLUMN ENCRYPTION KEY**

Defining a Database

A database is the warehouse for organizing, storing, and managing data. Defining a database includes: creating a database, altering the database attributes, and dropping the database. **Table 1-347** lists the involved SQL statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-347 SQL statements for defining a database

Description	SQL Statement
Creating a database	**CREATE DATABASE**
Altering database attributes	**ALTER DATABASE**
Dropping a Database	**DROP DATABASE**

Defining a Schema

A schema is the set of a group of database objects and is used to control the access to the database objects. **Table 1-348** lists the involved SQL statements.

Table 1-348 SQL statements for defining a schema

Description	SQL Statement
Creating a schema	**CREATE SCHEMA**
Altering schema attributes	**ALTER SCHEMA**
Dropping a schema	**DROP SCHEMA**

Defining a Tablespace

A tablespace is used to manage data objects and corresponds to a catalog on a disk. **Table 1-349** lists the involved SQL statements.

Table 1-349 SQL statements for defining a tablespace

Description	SQL Statement
Creating a tablespace	**CREATE TABLESPACE**
Altering tablespace attributes	**ALTER TABLESPACE**
Dropping a tablespace	**DROP TABLESPACE**

Defining a Table

A table is a special data structure in a database and is used to store data objects and relationship between data objects. **Table 1-350** lists the involved SQL statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-350 SQL statements for defining a table

Description	SQL Statement
Creating a table	**CREATE TABLE**
Altering table attributes	**ALTER TABLE**
Modifying one or more tables to be flashed back	**RENAME TABLE**
Dropping a table	**DROP TABLE**
Creating a table from the results of a query	**CREATE TABLE AS**

Defining a Partitioned Table

Partitioned tables are a type of logical table designed to improve query performance, though data is physically stored in ordinary tables. **Table 1-351** lists the involved SQL statements.

Table 1-351 SQL statements for defining a partitioned table

Description	SQL Statement
Creating a partitioned table	**CREATE TABLE PARTITION**
Modifying a partitioned table (including adding/deleting partitions and modifying partition attributes)	**ALTER TABLE PARTITION**
Dropping a partitioned table	**DROP TABLE**
Creating a partitioned table based on the query result	**CREATE TABLE PARTITION | SUBPARTITION AS**

Defining an Index

An index indicates the sequence of values in one or more columns in a database table. It is a data structure that improves the speed of data access to specific information in a database table. **Table 1-352** lists the involved SQL statements.

Table 1-352 SQL statements for defining an index

Description	SQL Statement
Creating an index	**CREATE INDEX**
Altering index attributes	**ALTER INDEX**
Dropping an index	**DROP INDEX**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	SQL Statement
Rebuilding an index	REINDEX
Creating a global secondary index in a specified table	CREATE GLOBAL INDEX

Defining a Stored Procedure

A stored procedure is a set of SQL statements for achieving specific functions and is stored in the database after compilation. Users can specify a name and provide parameters (if necessary) to execute the stored procedure. **Table 1-353** lists the involved SQL statements.

Table 1-353 SQL statements for defining a stored procedure

Description	SQL Statement
Creating a stored procedure	CREATE PROCEDURE
Dropping a stored procedure	DROP PROCEDURE
Altering the attributes of a user-defined stored procedure or recompiling a stored procedure	ALTER PROCEDURE

Defining a Function

In GaussDB, a function is similar to a stored procedure, which is a set of SQL statements. The function and stored procedure are used the same. **Table 1-354** lists the involved SQL statements.

Table 1-354 SQL statements for defining a function

Description	SQL Statement
Creating a function	CREATE FUNCTION
Modifying attributes of a function or recompiling the function	ALTER FUNCTION
Dropping a function	DROP FUNCTION

Defining a Package

A package consists of the package specification and package body. It is used to manage stored procedures and functions by class, which is similar to classes in languages such as Java and C++. **Table 1-355** lists the involved SQL statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-355 SQL statements for defining a package

Description	SQL Statement
Creating a package	CREATE PACKAGE
Dropping a package	DROP PACKAGE
Altering a package attribute	ALTER PACKAGE

NOTE

In the current mode, the PACKAGE syntax must be executed in an ORA-compatible database.

Defining a Resource Pool

A resource pool is a system catalog used by the workload management module to specify attributes related to resource management, such as Cgroups. **Table 1-356** lists the involved SQL statements.

Table 1-356 SQL statements for defining a resource pool

Description	SQL Statement
Creating a resource pool	CREATE RESOURCE POOL
Altering a resource pool attribute	ALTER RESOURCE POOL
Dropping a resource pool	DROP RESOURCE POOL

Defining an Aggregate Function

Table 1-357 SQL statements for defining an aggregate function

Description	SQL Statement
Creating an aggregate function	CREATE AGGREGATE
Modifying an aggregate function	ALTER AGGREGATE
Dropping an aggregate function	DROP AGGREGATE

Defining a Plug-in Extension

This feature is for internal use. Only advanced packages or extensions in the database can be operated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-358 SQL statements for defining a plug-in extension

Description	SQL Statement
Creating (installing) an extension	**CREATE EXTENSION**
Dropping an extension from the database	**DROP EXTENSION**

Defining a Data Type

Table 1-359 SQL statements for defining a data type

Description	SQL Statement
Creating a data type	**CREATE TYPE**
Modifying a data type	**ALTER TYPE**
Dropping a data type	**DROP TYPE**

Defining a Database Link

A database link is an object that can be used to operate a remote database. **Table 1-360** lists the related SQL statements.

Table 1-360 DATABASE LINK-related SQL statements

Description	SQL Statement
Creating a database link	**CREATE DATABASE LINK**
Modifying a database link	**ALTER DATABASE LINK**
Dropping a database link	**DROP DATABASE LINK**

Defining an Audit Policy

Table 1-361 SQL statements for defining an audit policy

Description	SQL Statement
Creating a unified audit policy	**CREATE AUDIT POLICY**
Altering a unified audit policy	**ALTER AUDIT POLICY**
Dropping an audit policy	**DROP AUDIT POLICY**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting the nodeis_active Column of a Specified Node

ALTER COORDINATOR describes how to change the value of the **nodeis_active** column of a specified node in the pgxc_node system catalog.

Defining a Directory

Table 1-362 SQL statements for defining a directory

Description	SQL Statement
Creating a directory	**CREATE DIRECTORY**
Modifying attributes of a directory	**ALTER DIRECTORY**
Dropping a specified directory	**DROP DIRECTORY**

Defining a Foreign Table

Table 1-363 SQL statements for defining a foreign table

Description	SQL Statement
Creating a foreign table	**CREATE FOREIGN TABLE**
Altering attributes of a foreign table	**ALTER FOREIGN TABLE**
Dropping a foreign table	**DROP FOREIGN TABLE**

Defining a Foreign Data Wrapper

Table 1-364 SQL statements related to the foreign data wrapper

Description	SQL Statement
Creating a foreign data wrapper	**CREATE FOREIGN DATA WRAPPER**
Altering attributes of a foreign data wrapper	**ALTER FOREIGN DATA WRAPPER**
Dropping a foreign data wrapper	**DROP FOREIGN DATA WRAPPER**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

SQL Statements Related to the gs_global_config System Catalog

Table 1-365 SQL statements related to the gs_global_config system catalog

Description	SQL Statement
Adding and changing parameter values in a system catalog	ALTER GLOBAL CONFIGURATION
Dropping parameter values from a system catalog	DROP GLOBAL CONFIGURATION
Inserting one or more weak passwords into a system catalog	CREATE WEAK PASSWORD DICTIONARY
Dropping all weak passwords in a system catalog	DROP WEAK PASSWORD DICTIONARY

Defining a User Group

Table 1-366 SQL statements for defining a user group

Description	SQL Statement
Creating a user group	CREATE GROUP
Altering attributes of a user group	ALTER GROUP
Dropping a user group	DROP GROUP

Defining a Procedural Language

Table 1-367 SQL statements for defining a procedural language

Description	SQL Statement
Creating a procedural language	CREATE LANGUAGE
Altering attributes of a procedural language	ALTER LANGUAGE
Dropping a procedural language	DROP LANGUAGE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Defining a Masking Policy

Table 1-368 SQL statements for defining a masking policy

Description	SQL Statement
Creating a masking policy	**CREATE MASKING POLICY**
Altering a masking policy	**ALTER MASKING POLICY**
Dropping a masking policy	**DROP MASKING POLICY**

Defining a Materialized View

Table 1-369 SQL statements for defining a materialized view

Description	SQL Statement
Creating a complete-refresh materialized view	**CREATE MATERIALIZED VIEW**
Creating a fast-refresh materialized view	**CREATE INCREMENTAL MATERIALIZED VIEW**
Altering multiple auxiliary attributes of an existing materialized view	**ALTER MATERIALIZED VIEW**
Forcibly deleting an existing materialized view from the database	**DROP MATERIALIZED VIEW**
Refreshing a materialized view in complete-refresh mode	**REFRESH MATERIALIZED VIEW**
Refreshing a materialized view in fast-refresh mode	**REFRESH INCREMENTAL MATERIALIZED VIEW**

Defining a Node

Table 1-370 SQL statements for defining a node

Description	SQL Statement
Creating a node	**CREATE NODE**
Altering attributes of an existing node	**ALTER NODE**
Dropping a node	**DROP NODE**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Defining a resource label

Table 1-371 SQL statements for defining a resource label

Description	SQL Statement
Creating a resource label	**CREATE RESOURCE LABEL**
Altering a resource label	**ALTER RESOURCE LABEL**
Dropping a resource label	**DROP RESOURCE LABEL**

Defining a row-level security policy

Table 1-372 SQL statements for defining a row-level security policy

Description	SQL Statement
Creating a row-level security policy for a table	**CREATE ROW LEVEL SECURITY POLICY**
Modifying an existing row-level security policy	**ALTER ROW LEVEL SECURITY POLICY**
Dropping a row-level security policy from a table	**DROP ROW LEVEL SECURITY POLICY**

Defining a Sequence

Table 1-373 SQL statements for defining a sequence

Description	SQL Statement
Adding a sequence to the current database	**CREATE SEQUENCE**
Altering parameters of a specified sequence	**ALTER SEQUENCE**
Dropping a sequence from the current database	**DROP SEQUENCE**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Defining a Foreign Server

Table 1-374 SQL statements for defining a foreign server

Description	SQL Statement
Creating a foreign server	**CREATE SERVER**
Adding, altering, and dropping parameters of a specified server	**ALTER SERVER**
Dropping a specified server	**DROP SERVER**

Defining a Synonym Object

Table 1-375 SQL statements for defining a synonym object

Description	SQL Statement
Creating a synonym object	**CREATE SYNONYM**
Modifying the owner of a synonym object	**ALTER SYNONYM**
Dropping a specified synonym object	**DROP SYNONYM**

Defining a Trigger

Table 1-376 SQL statements for defining a trigger

Description	SQL Statement
Creating a trigger	**CREATE TRIGGER**
Altering trigger attributes	**ALTER TRIGGER**
Dropping a trigger	**DROP TRIGGER**

Defining a View

Table 1-377 SQL statements for defining a view

Description	SQL Statement
Creating a view	**CREATE VIEW**
Modifying the auxiliary attributes of a view	**ALTER VIEW**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	SQL Statement
Forcibly dropping an existing view from the database	**DROP VIEW**

Collecting Statistics

For details about how to collect statistics related to the contents of ordinary tables in the database, see **ANALYZE | ANALYSE**.

Creating a Synchronization Point

For details about how to create a synchronization point between nodes in a new cluster, see **CREATE BARRIER**.

Creating an Encoding Conversion Task

For details about how to create a new conversion between two character set encodings, see **CREATE CONVERSION**.

Defining a Model

Table 1-378 SQL statements for defining a model

Description	SQL Statement
Training a machine learning model and saving the model	**CREATE MODEL**
Dropping a model that has been trained and saved	**DROP MODEL**

Defining a Security Label

Table 1-379 SQL statements for defining a security label

Description	SQL Statement
Creating a security label	**CREATE SECURITY LABEL**
Applying, updating, or canceling a security label	**SECURITY LABEL ON**
Dropping a security label	**DROP SECURITY LABEL**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Importing a Database/Table

Table 1-380 SQL statements for data import

Description	SQL Statement
Preparation phase for importing a database	**IMPDP DATABASE CREATE**
Execution phase of importing a database	**IMPDP RECOVER**
Preparation phase for importing a table	**IMPDP TABLE PREPARE**
Execution phase of importing a table	**IMPDP TABLE**

Exporting a File

Table 1-381 SQL statements for exporting files

Description	SQL Statement
Exporting all physical files of a database	**EXPDP DATABASE**
Exporting all files related to a table	**EXPDP TABLE**

Full-Text Search

Table 1-382 SQL statements for full-text search

Description	SQL Statement
Creating a text search configuration	**CREATE TEXT SEARCH CONFIGURATION**
Creating a full-text search dictionary	**CREATE TEXT SEARCH DICTIONARY**
Altering attributes of a text search configuration	**ALTER TEXT SEARCH CONFIGURATION**
Altering attributes of a full-text search dictionary	**ALTER TEXT SEARCH DICTIONARY**
Dropping an existing text search configuration	**DROP TEXT SEARCH CONFIGURATION**
Dropping a full-text search dictionary	**DROP TEXT SEARCH DICTIONARY**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Clearing a Recycle Bin

GaussDB provides a statement for clearing a recycle bin. For details, see **PURGE**.

Clustering a Table

GaussDB provides a statement for clustering a table based on a specified index. For details, see **CLUSTER**.

Defining an Object Comment

GaussDB provides a statement for creating and modifying comments of a specified object. For details, see **COMMENT**.

Creating a Table Based on Query Results

GaussDB supports statements for creating a table based on query results and inserting queried data into the new table. For details, see **SELECT INTO**.

Restoring a Table to an Earlier State

GaussDB allows you to restore a table to an earlier state in the event of a manual operation or application error. For details, see **TIMECAPSULE TABLE**.

Truncating Table Data

GaussDB provides a statement for quickly truncating all rows from a table. For details, see **TRUNCATE**.

Recycling Storage Space

GaussDB supports statements for recycling storage space occupied by deleted rows in a table or B-tree index. For details, see **VACUUM**.

1.13.4 DML Syntax Overview

Data manipulation language (DML) is used to perform operations on data in database tables, such as inserting, updating, querying, or deleting data.

Inserting Data

Table 1-383 SQL statements for inserting data

Function	SQL Statement
Inserting one or more records to a database table	**INSERT**
Inserting one or more records to multiple tables	**INSERT ALL**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Updating Data

For details about how to modify one or more records in a database table, see **UPDATE**.

Changing or Inserting Data

GaussDB provides a statement for matching data in the target table with that in the source table based on join conditions. If the join conditions are met, the target table is updated. If the join conditions are not met, data is inserted into the target table. For details, see **MERGE INTO**.

Querying Data

The query statement SELECT is used to retrieve required data in a database. For details, see **SELECT**.

Deleting Data

GaussDB provides a statement for deleting data that meets specified conditions from a table. For details, see **DELETE**.

Copying Data

GaussDB provides a statement for copying data between tables and files. For details, see **COPY**.

Locks

Table 1-384 Lock-related SQL statements

Function	SQL Statement
Multiple lock modes (used to control concurrent access to table data)	**LOCK**
Bucket-based lock	**LOCK BUCKETS**

Calling a Function

GaussDB provides multiple statements for calling functions. These statements have the same syntax structure. For details, see **CALL** and **CREATE FUNCTION**.

Prepared Statements

Table 1-385 SQL statements related to prepared statements

Description	SQL Statement
Creating a prepared statement	**PREPARE**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	SQL Statement
Executing a prepared statement	**EXECUTE**
Deallocating a prepared statement	**DEALLOCATE**

Killing a Session

A session is a connection established between a user and a database. GaussDB provides a statement for killing a session. For details, see **ALTER SYSTEM KILL SESSION**.

Executing an Anonymous Code Block

GaussDB provides a statement for executing an anonymous code block. For details, see **DO**.

Executing an SQL Statement on a Specified Node

GaussDB provides a statement for executing an SQL statement on a specified node. For details, see **EXECUTE DIRECT**.

Retrieving Data Using a Cursor

GaussDB provides a statement for retrieving data using a created cursor. For details, see **FETCH**.

Inserting or Replacing Data

GaussDB provides a statement for inserting or replacing data in a table. For details, see **REPLACE**.

Calculating a Value based on an Expression

GaussDB provides a statement for calculating a value of a row or a group of rows based on a given value expression. For details, see**VALUES**.

Importing Data

GaussDB provides a statement for importing data from a file to a specified table in the database. For details, see **LOAD DATA**.

Defining a Cursor

To process SQL statements, a stored procedure thread allocates memory to store context. Cursors are handles or pointers pointing to context regions. With cursors, stored procedures can control alterations in context areas. **Table 1-386** lists the involved SQL statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-386 SQL statements for defining a cursor

Description	SQL Statement
Creating a cursor	CURSOR
	DECLARE
Moving a cursor	MOVE
Closing a cursor	CLOSE

Showing an Execution Plan of an SQL Statement

GaussDB provides a statement for showing an execution plan of an SQL statement. For details, see **EXPLAIN**.

1.13.5 Other Syntax List

In addition to DCL, DDL, and DML syntax, GaussDB also provides syntax for other functions.

Shutting Down The Current Node

GaussDB allows users to run the **SHUTDOWN** command to shut down the current database node. For details, see **SHUTDOWN**.

SQL Statements Related to Bucket Scale-out

GaussDB supports statements used by the scale-out tool to notify the kernel of the buckets that have been migrated. For details, see **MARK BUCKETS**.

Clearing Database Connections

GaussDB supports statements for clearing database connections. For details, see **CLEAN CONNECTION**.

Storing Execution Plans

GaussDB provides a statement for storing query execution plan information in the **PLAN_TABLE** table. For details, see **EXPLAIN PLAN**.

Prediction

GaussDB provides a statement for using a trained model to perform prediction tasks. For details, see **PREDICT BY**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Managing a Transaction

Table 1-387 SQL statements related to transactions

Description	SQL Statement
Rolling back the current transaction and canceling the changes in the transaction	ABORT
	ROLLBACK
Starting a transaction	BEGIN
	SET TRANSACTION
	START TRANSACTION
Setting transaction log checkpoints	CHECKPOINT
Committing the current transaction	COMMIT | END
Committing a prepared two-phase transaction	COMMIT PREPARED
Preparing the current transaction for two-phase commit	PREPARE TRANSACTION
Deleting a savepoint previously defined for the current transaction	RELEASE SAVEPOINT
Canceling a transaction ready for two-phase commit	ROLLBACK PREPARED
Rolling back to a savepoint	ROLLBACK TO SAVEPOINT
Creating a savepoint in the current transaction	SAVEPOINT
Setting constraint check timing for the current transaction	SET CONSTRAINTS

Setting a User Identifier

Table 1-388 SQL statements for setting a user identifier

Description	SQL Statement
Setting the current user identifier of the current session	SET ROLE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	SQL Statement
Setting the session user identifier and the current user identifier of the current session to a specified user	SET SESSION AUTHORIZATION

Upgrading a Database

Table 1-389 SQL statements for upgrading a database

Description	SQL Statement
Generating an upgrade rollback script for system objects	GENERATED UPDATE SYSTEM OBJECT
Updates the version information of system objects to the system catalog GS_SYSTEM_OBJECTS_VERSION to determine the changes of system objects during the upgrade.	REFRESH SYSTEM OBJECT

1.13.6 Online DDL and Related Operations

DDL operations on table objects (definition or modification of database table objects) are classified into the following types based on the impact on data:

Instant Change DDL

Definition and feature:

- Only the metadata of the table is modified. No data operation is required.
- The execution is fast and usually completed in milliseconds.
- An exclusive lock is held for a short period of time during the execution, which has little impact on concurrent services.

Impact of concurrency:

- Normal scenario: If there is no long transaction on the target table, the impact on services is small.
- Abnormal scenario: If a long transaction exists, DDL execution may be blocked and subsequent services may be affected.

Table 1-390 lists the supported instant change DDL operations. For details about the syntax, see ALTER TABLE.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-390 Instant change DDL operations supported by the current version

Category	Operation	Required Time	Remarks
Column operations	The supported operations are as follows: • Add columns, that is, set default values of columns to an expression. • Drop columns. • Rename columns. • Set and drop default values of columns. • Extend the varchar size (only when the length increases). • Modify the numeric precision **numeric(*p*,*s*)**, only when *p* increases and *s* remains unchanged. • Modify the definition of an ENUM column.	When only metadata is changed, the time required is irrelevant to the data volume.	• Instant DDL operations do not require the ONLINE keyword. If the ONLINE keyword is added to an instant DDL operation, it will be ignored. • During the execution of instant DDL operations, an exclusive lock is held for a short period of time. If there is no concurrent long transaction, DDL operations are completed in milliseconds, which has little impact on concurrent transactions. Before a DDL operation is executed, if there are concurrent long transactions on the target table, the DDL operation is blocked while waiting for an exclusive lock. This further blocks subsequent concurrent transactions
Table operations	The supported operations are as follows: • Rename tables. • Specify character sets. • Drop tables.	When only metadata is changed, the time required is irrelevant to the data volume.	
Tablespace operations	The supported operations are as follows: • Rename tablespaces. • Change owners of tablespaces. • Delete tablespaces.	When only metadata is changed, the time required is irrelevant to the data volume.	
Constraint operations	The supported operations are as follows: • Set a column constraint to **NULL**.	When only metadata is changed, the time required is irrelevant to the data volume.	
Partitio	The supported	When only metadata	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Category	Operation	Required Time	Remarks
n operations	operations are as follows: • Add, drop, and truncate partitions, enable or disable row movement, and invalidate partitioned indexes.	is changed, the time required is irrelevant to the data volume.	after the DDL operation.

Data Operation DDL

Definition and feature:

- The actual data needs to be reorganized or checked.
- The execution time is proportional to the data volume.
- In traditional mode, tables cannot provide read or write services during execution, which may cause long-term service interruption.

Online DDL

Definition and feature:

- With the ONLINE keyword, data operation DDL statements are executed online by reorganizing (checking) basic data and appending (checking) incremental data.
- ROWID tables are not supported.

Impact of concurrency:

- During the execution, users can continue to read and write the database for most of the time.
- An exclusive lock is held only for a short period of time at the beginning and end.
- The impact on normal services is significantly reduced.

Table 1-391 lists the supported online DDL operations. For details about online DDL parameter settings and behavior limitations, see **ALTER TABLE**, and **VACUUM**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-391 Online DDL operations supported by the current version

Category	Operation	Required Time	Remarks
Column operations	The supported operations are as follows: • Change column lengths. • Change column precisions. • Change column types.	When data needs to be reorganized, the time required depends on the data volume.	• To perform online DDL operations, you need to use the ONLINE keyword. • Online DDL operations still hold an exclusive lock for a short period of time during initialization and final metadata switching, which may be affected by concurrent long transactions on the target table. Before an online DDL operation is executed or during metadata switching, if there are concurrent long transactions on the target table, the DDL operation may be blocked while waiting for an exclusive lock. This further blocks subsequent concurrent transactions. • Online DDL operations do not support rowid tables. For details about the syntax, see **ALTER TABLE** and **VACUUM**.
Table operations	The supported operations are as follows: Move tablespaces online at the table level.	When data needs to be moved, the time required depends on the data volume.	
Constraint operations	The supported operations are as follows: Add NOT NULL, CHECK, UNIQUE, and PRIMARY KEY constraints.	When data needs to be queried, the time required depends on the data volume.	
Partition operations	The supported operations are as follows: Split, merge, and exchange partitions and move partition tablespaces.	When data needs to be moved, the time required depends on the data volume.	
Recycling storage space	VACUUM FULL	When data needs to be reorganized, the time required depends on the data volume.	
Index operations	Build, rebuild, or drop an index using CONCURRENTLY.	When data needs to be queried, the time required depends on the data volume.	For details about index operations, see **CREATE INDEX**, **REINDEX** and **DROP INDEX**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.7 A

1.13.7.1 ABORT

Description

Rolls back the current transaction and cancels the changes in the transaction.

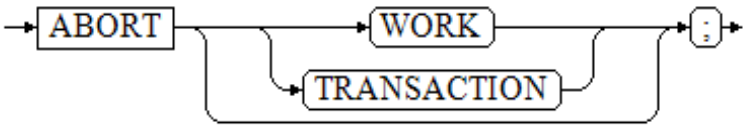
It is equivalent to **ROLLBACK**, and is present only for historical reasons. Now **ROLLBACK** is recommended.

Precautions

ABORT has no impact outside a transaction, but will return a NOTICE message.

Syntax

```
ABORT [ WORK | TRANSACTION ] ;
```



Parameters

WORK | TRANSACTION

Optional keyword, which is used only to improve readability.

Examples

```
-- Create the customer_demographics_t1 table.
gaussdb=# CREATE TABLE customer_demographics_t1
(
    CD_DEMO_SK              INTEGER              NOT NULL,
    CD_GENDER               CHAR(1)                      ,
    CD_MARITAL_STATUS       CHAR(1)                      ,
    CD_EDUCATION_STATUS     CHAR(20)                     ,
    CD_PURCHASE_ESTIMATE    INTEGER                      ,
    CD_CREDIT_RATING        CHAR(10)                     ,
    CD_DEP_COUNT            INTEGER                      ,
    CD_DEP_EMPLOYED_COUNT   INTEGER                      ,
    CD_DEP_COLLEGE_COUNT    INTEGER
)
DISTRIBUTE BY HASH (CD_DEMO_SK);

-- Insert data.
gaussdb=# INSERT INTO customer_demographics_t1 VALUES(1920801,'M', 'U', 'DOCTOR DEGREE', 200,
'GOOD', 1, 0,0);

-- Start a transaction.
gaussdb=# START TRANSACTION;

-- Update the column value. Change the value of cd_education_status to Unknown.
gaussdb=# UPDATE customer_demographics_t1 SET cd_education_status= 'Unknown';

-- Abort the transaction. All updates are rolled back.
gaussdb=# ABORT;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Query data and find that the value of cd_education_status is not changed to Unknown.
gaussdb=# SELECT * FROM customer_demographics_t1 WHERE cd_demo_sk = 1920801;
cd_demo_sk | cd_gender | cd_marital_status | cd_education_status  | cd_purchase_estimate | cd_credit_rating
| cd_dep_count | cd_dep_employed_count | cd_dep_college_count
------------+-----------+-------------------+----------------------+----------------------+-----------------
+--------------+-----------------------+----------------------
   1920801 | M         | U                 | DOCTOR DEGREE        |                200 | GOOD             |           1
|                0 |                     0
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE customer_demographics_t1;
```

Helpful Links

SET TRANSACTION, **COMMIT | END**, and **ROLLBACK**

1.13.7.2 ALTER AGGREGATE

Description

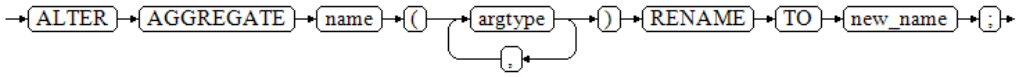
Modifies the definition of an aggregate function, including the name, owner, and schema.

Precautions

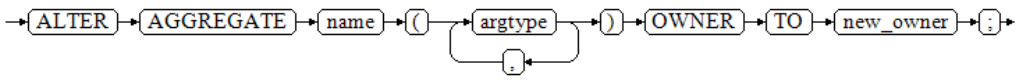
To use the ALTER AGGREGATE statement, you must be the owner of the aggregate function. To modify the schema of the aggregate function, you must have the CREATE permission on the new schema. To modify the owner of the aggregate function, you must be a direct or indirect member of the new owner and the member must have the CREATE permission on the schema of the aggregate function. The preceding rules prevent owners from performing illegal operations by deleting and rebuilding aggregate functions. However, users with the SYSADMIN permission can still alter the ownership of any aggregate function in any way.

Syntax

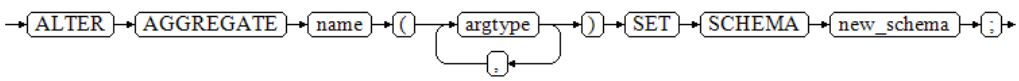
```
ALTER AGGREGATE name ( argtype [ , ... ] ) RENAME TO new_name;
```




```
ALTER AGGREGATE name ( argtype [ , ... ] ) OWNER TO new_owner;
```




```
ALTER AGGREGATE name ( argtype [ , ... ] ) SET SCHEMA new_schema;
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **name**

  Name (optionally schema-qualified) of an existing aggregate function.

- **argtype**

  Input data type of the aggregate function. To reference a zero-parameter aggregate function, you can pass an asterisk (*) instead of the input data type list.

- **new_name**

  New name of an aggregate function.

- **new_owner**

  New owner of the aggregate function.

- **new_schema**

  New schema of the aggregate function.

Examples

- Change the name of the aggregate function.

```
-- Create a user-defined function.
gaussdb=# CREATE OR REPLACE FUNCTION int_add(int,int)
 RETURNS int AS $BODY$
DECLARE
BEGIN
 RETURN $1 + $2;
END;
$BODY$ language plpgsql;

-- Create an aggregate function.
gaussdb=# CREATE AGGREGATE myavg (int)
(
    sfunc = int_add,
    stype = int,
    initcond = '0'
);

-- Rename the aggregate function myavg that accepts int-type parameters to my_average.
gaussdb=# ALTER AGGREGATE myavg(int) RENAME TO my_average;
```

- Change the owner of the aggregate function.

```
-- Create a user joe.
gaussdb=# CREATE USER joe PASSWORD '********';

-- Change the owner of the aggregate function my_average that accepts int-type parameters to joe.
gaussdb=# ALTER AGGREGATE my_average(int) OWNER TO joe;
```

- Change the schema of the aggregate function.

```
-- Create a schema myschema.
gaussdb=# CREATE SCHEMA myschema;

-- Move the aggregate function my_average that accepts int-type parameters to myschema.
gaussdb=# ALTER AGGREGATE my_average(int) SET SCHEMA myschema;

-- Delete the schema, user, and related function.
gaussdb=# DROP SCHEMA myschema CASCADE;
gaussdb=# DROP USER joe;
gaussdb=# DROP FUNCTION int_add(int,int);
```

Helpful Links

CREATE AGGREGATE and **DROP AGGREGATE**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Compatibility

The SQL standard does not contain the ALTER AGGREGATE statement.

1.13.7.3 ALTER ASYNC ENCRYPTION KEY ROTATION

Description

Rotates the key for asynchronous TDE.

Precautions

- This statement can be executed only when **enable_tde** and **enable_tde_async_encryption** are enabled.
- Only users with the SYSADMIN permission can run this statement.
- M-compatible databases do not support this statement.
- Unified audit logging is not supported.

Syntax

```
ALTER ASYNC ENCRYPTION KEY ROTATION;
```



Examples

```
-- Enable the asynchronous TDE.
gaussdb=# show enable_tde_async_encryption;
 enable_tde_async_encryption
-----------------------------
 on
(1 row)

-- Create two TDE tables t1 and t2. Run the \d command. The output indicates that the two tables use the same data key (that is, the asynchronous encryption key).
gaussdb=# CREATE TABLE t1 (c1 INT, c2 TEXT) WITH (enable_tde = on);
CREATE TABLE
gaussdb=# CREATE TABLE t2 (c1 INT, c2 TEXT) WITH (enable_tde = on);
CREATE TABLE
gaussdb=# \d

-- Rotate the asynchronous encryption key.
gaussdb=# ALTER ASYNC ENCRYPTION KEY ROTATION;
ALTER TDE ROTATE ENCRYPTION KEY

-- Create a TDE table t3. Run the \d command. The output indicates that the data key of table t3 is different from that of tables t1 and t2, that is, the asynchronous encryption key is rotated.
gaussdb=# CREATE TABLE t3 (c1 INT, c2 TEXT) WITH (enable_tde = on);
CREATE TABLE
gaussdb=# \d

-- Drop encrypted tables.
gaussdb=# DROP TABLE t1,t2,t3;
```

1.13.7.4 ALTER AUDIT POLICY

Description

Modifies a unified audit policy.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

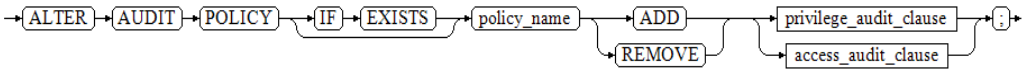
Precautions

- Only users with the POLADMIN or SYSADMIN permission, or the initial user can maintain audit policies.
- The unified audit policy takes effect only after **enable_security_policy** is enabled.

Syntax

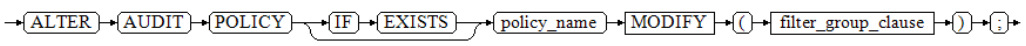
Add or delete an operation type in the audit policy.

```
ALTER AUDIT POLICY [ IF EXISTS ] policy_name { ADD | REMOVE } { privilege_audit_clause |
access_audit_clause };
```



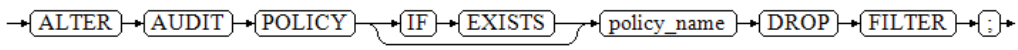
Change the filter criteria in the audit policy.

```
ALTER AUDIT POLICY [ IF EXISTS ] policy_name MODIFY ( filter_group_clause );
```



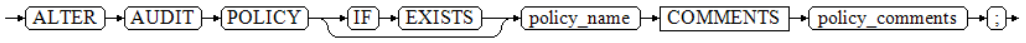
Delete the filter criteria from the audit policy.

```
ALTER AUDIT POLICY [ IF EXISTS ] policy_name DROP FILTER;
```



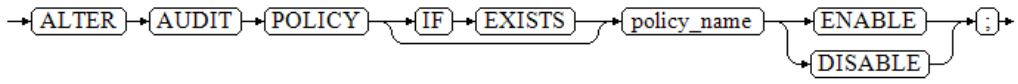
Change the description of the audit policy.

```
ALTER AUDIT POLICY [ IF EXISTS ] policy_name COMMENTS policy_comments;
```



Enable or disable the audit policy.

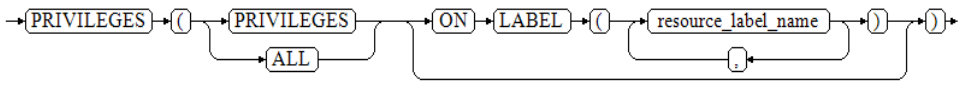
```
ALTER AUDIT POLICY [ IF EXISTS ] policy_name { ENABLE | DISABLE };
```



- privilege_audit_clause

  PRIVILEGES operation type and target resource label in the audit policy.

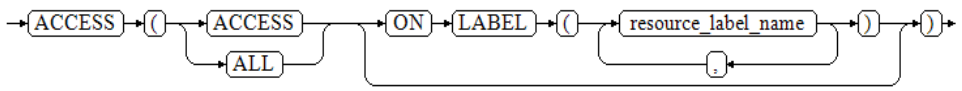
  ```
  PRIVILEGES ({ PRIVILEGES | ALL } [ ON LABEL ( resource_label_name [, ... ] ) ])
  ```

- access_audit_clause

  ACCESS operation type and target resource label in the audit policy.

  ```
  ACCESS ({ ACCESS | ALL } [ ON LABEL ( resource_label_name [, ... ] ) ])
  ```

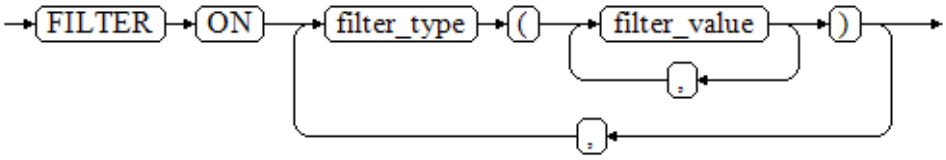
  

- filter_group_clause

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Filter criteria in the audit policy.

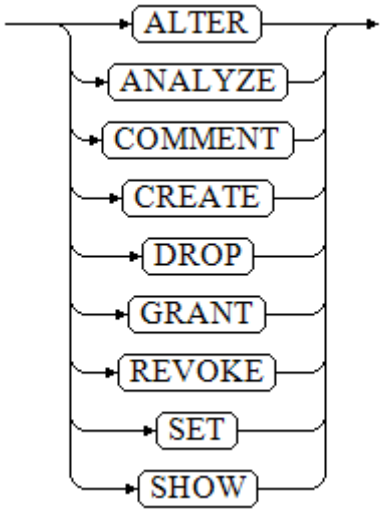
```
FILTER ON { filter_type ( filter_value [, ... ] ) } [, ... ]
```



- PRIVILEGES:

  Perform audit when the following operations are performed on the database.

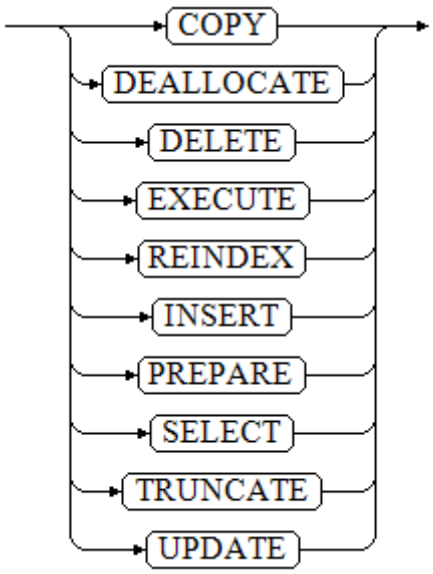
  ```
  { ( ALTER | ANALYZE | COMMENT | CREATE | DROP | GRANT | REVOKE | SET | SHOW ) }
  ```

- ACCESS:

  Perform audit when the following operations are performed on the database.

  ```
  { ( COPY | DEALLOCATE | DELETE | EXECUTE | REINDEX | INSERT | PREPARE | SELECT | TRUNCATE |
  UPDATE ) }
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **policy_name**

  Specifies the name of an audit policy, which must be unique.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **resource_label_name**

  Specifies a resource label name.

- **PRIVILEGES**

  Specifies the database operations to be audited. Currently, the following operations can be audited: ALTER, ANALYZE, COMMENT, CREATE, DROP, GRANT, REVOKE, SET, and SHOW.

- **ACCESS**

  Specifies the database operations to be audited. Currently, the following operations can be audited: COPY, DEALLOCATE, DELETE, EXECUTE, REINDEX, INSERT, PREPARE, SELECT, TRUNCATE, and UPDATE.

- **ALL**

  Audits all database operations supported by **PRIVILEGES** or **ACCESS**. When the format is { PRIVILEGES | ALL }, **ALL** indicates all operations specified by **PRIVILEGES**. When the format is { ACCESS | ALL }, **ALL** indicates all operations specified by **ACCESS**.

- **filter_type**

  Specifies the types of information to be filtered by the policy: **IP**, **ROLES**, and **APP**.

- **filter_value**

  Specifies the detailed information to be filtered.

- **policy_comments**

  Records comments on the audit policy.

- **ENABLE | DISABLE**

  Enables or disables the unified audit policy.

Examples

- Add or delete an operation type in the audit policy.

```
-- Create audit policy adt1 for executing CREATE on the database.
gaussdb=# CREATE AUDIT POLICY adt1 PRIVILEGES CREATE;

-- Add DROP to the adt1 audit policy.
gaussdb=# ALTER AUDIT POLICY adt1 ADD PRIVILEGES (DROP);

-- Delete DROP from the adt1 audit policy.
gaussdb=# ALTER AUDIT POLICY adt1 REMOVE PRIVILEGES (DROP);
```

- Change the comments of the audit policy.

```
-- Change the comment of the adt1 audit policy to adt1_comments.
gaussdb=#  ALTER AUDIT POLICY adt1 COMMENTS 'adt1_comments';
```

- Change the filter information of the audit policy.

```
-- Create a user bob_audit.
gaussdb=#  CREATE USER bob_audit PASSWORD '********';

-- Change the filtering user of the adt1 audit policy to bob_audit.
gaussdb=#  ALTER AUDIT POLICY adt1 MODIFY (FILTER ON (ROLES(bob_audit)));
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Delete user bob_audit.
gaussdb=# DROP USER bob_audit;
```

- Delete the filter criteria from the audit policy.

```
-- Delete the filter criteria in the adt1 audit policy.
gaussdb=#  ALTER AUDIT POLICY adt1  DROP FILTER;
```

- Disable the audit policy.

```
-- Disable the adt1 audit policy.
gaussdb=#  ALTER AUDIT POLICY adt1  DISABLE;

-- Delete the adt1 audit policy.
gaussdb=# DROP AUDIT POLICY adt1;
```

Helpful Links

CREATE AUDIT POLICY and DROP AUDIT POLICY

1.13.7.5 ALTER COLUMN ENCRYPTION KEY

Description

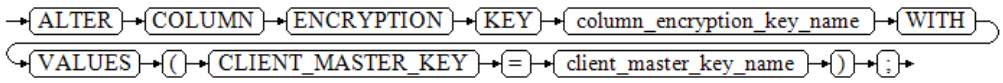
Encrypts the CMKs of CEKs in round robin (RR) mode and encrypts the plaintext of CEKs.

Precautions

- This syntax is specific to a fully-encrypted database. When connecting to the database server, enable the fully-encrypted database before using this syntax.
- This syntax takes effect on CMKs only. Encrypting the plaintext of CEKs does not change the ciphertext of the encrypted columns.

Syntax

```
ALTER COLUMN ENCRYPTION KEY column_encryption_key_name WITH VALUES ( CLIENT_MASTER_KEY = client_master_key_name );
```



Parameters

- **column_encryption_key_name**

  Specifies the key name. In the same namespace, the value of this parameter must be unique.

  Value range: character strings complying with Identifier Naming Conventions.

- **client_master_key_name**

  Specifies the new CMK used to re-encrypt the CEK.

  Value range: name of an existing CMK created using the CREATE CLIENT MASTER KEY syntax. The encrypted CMKs are different from those specified before RR encryption.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

SM algorithm constraints: SM2, SM3, and SM4 are Chinese national cryptography standards. To avoid legal risks, these algorithms must be used together. If the CMK used before CEK rotation is an SM algorithm, the CMK specified for CEK rotation must also be an SM algorithm.

Helpful Links

CREATE COLUMN ENCRYPTION KEY and **DROP COLUMN ENCRYPTION KEY**

1.13.7.6 ALTER COORDINATOR

Description

Changes the CN status. You can run the statement on any normal CN in the cluster to change the value of the **nodeis_active** field of a specified node in the pgxc_node system catalog. You can also specify the nodes where the system catalog is modified.

Precautions

- ALTER COORDINATOR is a statement for modifying system catalogs. Only administrators and users in internal maintenance mode (for example, CM administrator) can execute this statement.
- This statement is dedicated for CN removal and must be used together with other operations. You are advised not to run it by yourself.
- After this statement is executed, **SELECT reload_active_coordinator()** needs to be called to update the connection pool information of the node on which the system catalog is modified.

Syntax

```
ALTER COORDINATOR nodename SET status
    WITH (other_nodename [, ... ]) (nodetype);
```

Parameters

- **nodename**

  Specifies the CN name corresponding to a row of records in the pgxc_node system catalog. After the node name is specified, the value of **nodeis_active** in the record is changed.

  Value range: character strings. Only CN names are allowed, and the names must have corresponding records in the pgxc_node system catalog.

- **status**

  Specifies the CN status, that is the updated value of **nodeis_active** in the pgxc_node system catalog.

  Value range: **FALSE** and **TRUE**

- **other_nodename**

  Specifies the range of nodes on which the SQL statement is executed. When **ALTER COORDINATOR** is executed, the SQL statement is automatically

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

delivered to all nodes in the range. The current execution node must be included.

Value range: a string. Only CNs are supported. Ensure that the node name has a corresponding record in the pgxc_node system catalog and the node is normal. Otherwise, the SQL statement fails to be executed.

- **nodetype**

  Type of the node to be changed, which is the value of **node_type** in the pgxc_node system catalog.

  Value range: **"C"** (CN)

Examples

- Remove a faulty CN.

```
-- The cluster has three CNs: cn_5001, cn_5002, and cn_5003, which are working properly. If cn_5001 is faulty and the removal time meets the requirement, cn_5001 needs to be removed from the cluster.
gaussdb=# ALTER COORDINATOR cn_5001 SET False WITH (cn_5002,cn_5003);

-- Refresh the pgxc_node system catalog. On the cn_5002 and cn_5003 nodes, run the following SQL statement to update the value of nodeis_active corresponding to the cn_5001 record in the pgxc_node system catalog to false:
gaussdb=# SELECT reload_active_coordinator();

-- Check the CN status. The value of nodeis_active changes to f (false).
gaussdb=#  SELECT nodeis_active,node_name FROM pgxc_node where node_name='cn_5001';
 nodeis_active | node_name
---------------+-----------
 f             | cn_5001
(1 row)
```

- Restore the CN whose fault has been rectified.

```
-- After the cn_5001 fault is rectified, add cn_5001 back to the cluster.
gaussdb=#  ALTER COORDINATOR cn_5001 SET True WITH (cn_5002,cn_5003);

-- Refresh the pgxc_node system catalog. On the cn_5002 and cn_5003 nodes, run the following SQL statement to update the value of nodeis_active corresponding to the cn_5001 record in the pgxc_node system catalog to true:
gaussdb=# SELECT reload_active_coordinator();

-- Check the CN status. The value of nodeis_active changes to t (true).
gaussdb=# SELECT nodeis_active,node_name FROM pgxc_node where node_name='cn_5001';
 nodeis_active | node_name
---------------+-----------
 t             | cn_5001
(1 row)
```

1.13.7.7 ALTER DATABASE

Description

Alters database attributes, including the database name, owner, connection limit, and object isolation.

Precautions

- Only the database owner or a user with the ALTER permission can run the **ALTER DATABASE** command. A system administrator has this permission by default. Here are the permission limits for the attributes you want to alter:
  - To change the database name, you must have the CREATEDB permission.

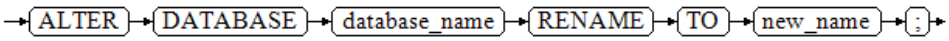
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - To change a database owner, you must be a database owner or system administrator and a member of the new owner role, with the CREATEDB permission.
  - To change the default tablespace of the database, the user must have the CREATE permission on the tablespace. After this operation, tables and indexes in a default tablespace are physically migrated to a new tablespace. Note that tables and indexes outside the default tablespace are not affected.
- Do not directly rename the current database. If you need to rename the database, connect to another database and rename the original database.
- Do not rename other schemas as **templatem** or rename the **templatem** or **templatea** schema as another name.

Syntax

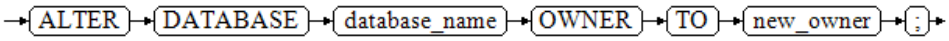
- Rename the database.

```
ALTER DATABASE database_name
   RENAME TO new_name;
```



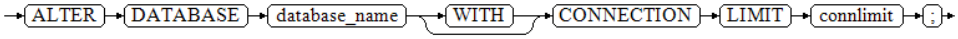
- Modify the owner of a database.

```
ALTER DATABASE database_name
   OWNER TO new_owner;
```



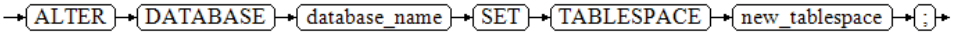
- Change the maximum number of connections to the database.

```
ALTER DATABASE database_name
   [ WITH ] CONNECTION LIMIT connlimit;
```



- Change the default tablespace of the database.

```
ALTER DATABASE database_name
   SET TABLESPACE new_tablespace;
```

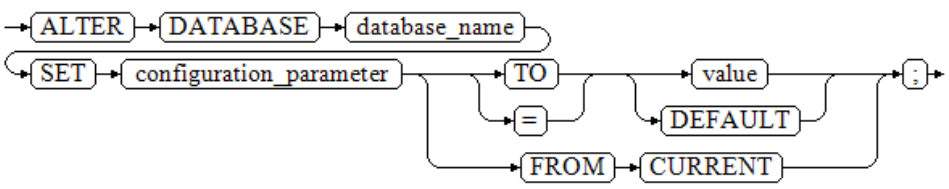


NOTICE

If some tables or objects in the database have been created in new_tablespace, the default tablespace of the database cannot be changed to **new_tablespace**. An error will be reported during the execution.

- Change the session parameter value of the database. The current version does not support setting database-level parameters.

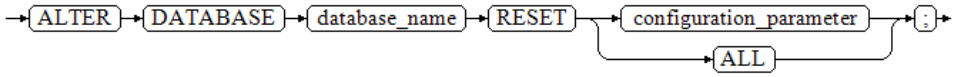
```
ALTER DATABASE database_name
   SET configuration_parameter { { TO | = } { value | DEFAULT } | FROM CURRENT };
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

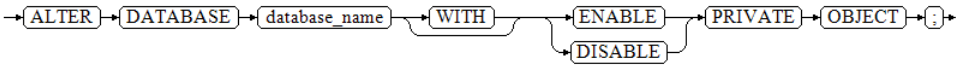
- Reset the database configuration parameter. The current version does not support setting database-level parameters.

```
ALTER DATABASE database_name RESET
  { configuration_parameter | ALL };
```



- Change the object isolation attribute of the database. To change the object isolation attribute of a database, the database must be connected. Otherwise, the change will fail.

```
ALTER DATABASE database_name [ WITH ] { ENABLE | DISABLE } PRIVATE OBJECT;
```

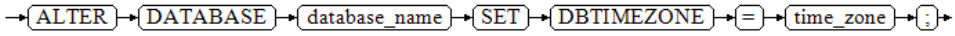


NOTE

For a newly created database, the object isolation attribute is disabled by default. When this attribute is enabled, the database automatically adds row-level security policies to the PG_CLASS, PG_ATTRIBUTE, PG_PROC, PG_NAMESPACE, PGXC_SLICE, PG_DATABASE, and PG_PARTITION system catalogs. Common users can view the objects (such as tables, functions, views, and columns) that they have the permission to access in the system catalogs. Administrators are not affected by this attribute. After this attribute is enabled, administrators can still view all database objects.

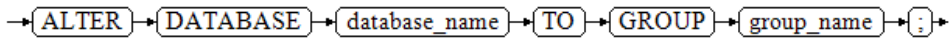
- Change the database time zone.

```
ALTER DATABASE database_name SET DBTIMEZONE = time_zone;
```



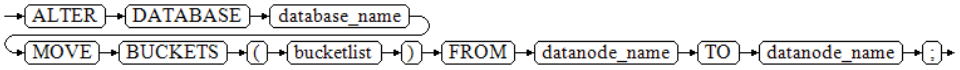
- Bind a node group to the hash bucket table to determine the bucket mapping.

```
ALTER DATABASE database_name
  TO GROUP group_name ;
```



- Migrate a specified bucket from one DN to another.

```
ALTER DATABASE database_name
  MOVE BUCKETS(bucketlist) FROM datanode_name TO datanode_name;
```



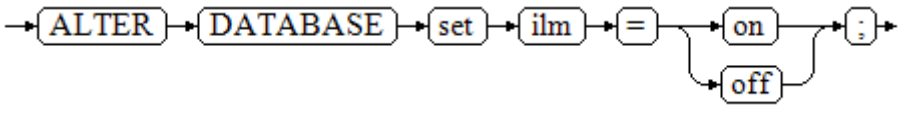
NOTE

This statement can be called only during hash bucket scale-out. It is a hash bucket database-level redistribution command. Example:

```
ALTER DATABASE testdb MOVE BUCKETS (0, 1, 2, 3) FROM datanode1 TO datanode3;
```

- Enable or disable the ILM feature of the database.

```
ALTER DATABASE set ilm = { on | off };
```



Parameters

- **database_name**

  Specifies the name of the database whose attributes are to be changed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: character strings complying with **Identifier Naming Conventions**.

- **new_name**

  Specifies the new name of a database.

  Value range: a string. It must comply with **Identifier Naming Conventions** and contain a maximum of 63 characters.

  - If the value contains more than 63 characters, the database truncates it and retains the first 63 characters as the database name.
  - If the database name contains uppercase letters, they are automatically converted to lowercase letters.
  - If you want to create a database name that contains uppercase letters, enclose the name with double quotation marks ("").

- **new_owner**

  Specifies the new owner of a database.

  Value range: a string. It must be a valid username.

- **connlimit**

  Specifies the maximum number of concurrent connections that can be made to this database (excluding administrators' connections).

  Value range: an integer in the range [–1,2^{31} – 1]. The default value is **–1**, indicating that the maximum number of concurrent database connections is not limited. You are advised to set the value to an integer ranging from 1 to 50.

- **new_tablespace**

  Specifies the new default tablespace of a database. The tablespace exists in the database.

  Value range: a string. It must be a valid tablespace name. The default tablespace is **pg_default**.

  NOTE

  If you change the default tablespace of a database, the tables and indexes in the old tablespace are moved into the new tablespace. This operation does not affect the tables or indexes in other non-default tablespaces.

- **time_zone**

  To set the time zone of the database, you must have the permission to operate the database.

  Value range: a string. Currently, the following time zones are supported:

  - Time zones supported by the system and their abbreviations.
  - [–15:59,+15:00]

  NOTE

  This parameter can be specified only when the connected database is in ORA-compatible mode and **a_format_version** is set to **'10c'** and **a_format_dev_version** is set to **'s2'**.

- **group_name**

  Specifies the name of the node group bound to a hash bucket table. Common users can only bind the table to the installation node group but cannot alter it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **bucketlist**

  Specifies the bucketlist involved in redistribution

  Value range: a string. The value range of each bucket is [0,1023].

- **datanode_name**

  Specifies the name of the node involved in bucket migration.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **configuration_parameter**

  Specifies the database session parameter whose attributes are to be altered.

  NOTICE

  The current version does not support setting database-level parameters.

  - **value**

    Sets a specified database session parameter to a given value.

    Value range: a string.

    - **DEFAULT**: The default system setting is used in the new session.
    - **OFF**: The setting is disabled.
    - **RESET**: The default system setting is used in the new session.
    - User-specified value: The value must meet the restriction of the modified parameter.

  - **FROM CURRENT**

    Uses the value of **configuration_parameter** of the current session.

- **RESET configuration_parameter**

  Resets the specified database session parameter.

  NOTICE

  The current version does not support resetting database-level parameters.

- **RESET ALL**

  Resets all database session parameters.

  NOTICE

  The current version does not support resetting database-level parameters.

Examples

- Rename the database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create the database testdb.
gaussdb=# CREATE DATABASE testdb;

-- Rename testdb to test_db1.
gaussdb=# ALTER DATABASE testdb RENAME TO test_db1;
```

- Change the maximum number of database connections.

```
-- Change the maximum number of connections to test_db1 to 100.
gaussdb=# ALTER DATABASE test_db1 WITH CONNECTION LIMIT 100;

-- View information about test_db1.
gaussdb=# SELECT datname,datconnlimit FROM pg_database WHERE datname = 'test_db1';
 datname  | datconnlimit
----------+--------------
 test_db1 |          100
(1 row)
```

- Change the database owner.

```
-- Create user scott.
gaussdb=# CREATE USER scott PASSWORD '********';

-- Change the owner of test_db1 to scott.
gaussdb=# ALTER DATABASE test_db1 OWNER TO scott;

-- View information about test_db1.
gaussdb=# SELECT t1.datname, t2.usename
          FROM pg_database t1, gs_user t2
          WHERE t1.datname='test_db1' AND t1.datdba=t2.usesysid;
 datname  | usename
----------+---------
 test_db1 | scott
(1 row)
```

- Change the default tablespace of the database.

```
-- Create a tablespace.
gaussdb=# CREATE TABLESPACE tbs_data1 RELATIVE LOCATION 'tablespace1/tbs_data1';

-- Change the default tablespace of test_db1.
gaussdb=# ALTER DATABASE test_db1 SET TABLESPACE tbs_data1;

-- View information about test_db1.
gaussdb=# SELECT t1.datname AS database, t2.spcname AS tablespace
          FROM pg_database t1, pg_tablespace t2
          WHERE t1.dattablespace = t2.oid AND
               t1.datname = 'test_db1';
 database | tablespace
----------+------------
 test_db1 | tbs_data1
(1 row)
```

- Change the database time zone.

  Prerequisites: The currently connected database is an ORA-compatible database, **a_format_version** is set to **'10c'**, and **a_format_dev_version** is set to **'s2'**.

```
-- Create an ORA-compatible database. The PRC time zone is used by default.
gaussdb=# CREATE DATABASE testdb2 DBCOMPATIBILITY = 'ORA';

-- Check the time zone of the testdb2 database.
gaussdb=# SELECT datname,datcompatibility,dattimezone FROM pg_database WHERE datname =
'testdb2';
 datname | datcompatibility | dattimezone
---------+------------------+-------------
 testdb2 | ORA              | PRC
(1 row)

-- Switch to the testdb2 database and change the database time zone to '+08:00'.
gaussdb=# \c testdb2
testdb2=# SET a_format_version='10c';
testdb2=# SET a_format_dev_version='s2';
testdb2=# ALTER DATABASE testdb2 SET DBTIMEZONE = '+08:00';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Check the database time zone.
testdb2=# SELECT datname,datcompatibility,dattimezone FROM pg_database WHERE datname =
'testdb2';
 datname | datcompatibility | dattimezone
---------+------------------+-------------
 testdb2 | ORA              | +08:00
(1 row)
```

- Change the object isolation attribute of the database.

```
-- Create user jack.
gaussdb=# CREATE USER jack PASSWORD '********';

-- Create the test_tbl1 table in test_db1.
gaussdb=# \c test_db1
test_db1=# CREATE TABLE test_tbl1(c1 int,c2 int);

-- Switch to user jack and view pg_tables.
test_db1=# SET ROLE jack PASSWORD '********';
test_db1=> SELECT tablename FROM pg_tables WHERE tablename = 'test_tbl1';
 tablename
-----------
 test_tbl1
(1 row)

-- Change the object isolation attribute.
test_db1=> SET ROLE scott PASSWORD '********';
test_db1=> ALTER DATABASE test_db1 ENABLE PRIVATE OBJECT;

-- Switch to user jack and view pg_tables.
test_db1=> SET ROLE jack PASSWORD '********';

-- Due to the isolation attribute, 0 data records can be queried.
test_db1=> SELECT tablename FROM pg_tables WHERE tablename = 'test_tbl1';
 tablename
-----------
(0 rows)

-- Switch to the default user and perform the deletion.
test_db1=> RESET ROLE;
test_db1=# DROP TABLE public.test_tbl1;

-- Switch to the default database. Change the database name based on actual situation.
test_db1=# \c postgres
gaussdb=# DROP DATABASE test_db1;
gaussdb=# DROP DATABASE testdb2;
gaussdb=# DROP TABLESPACE tbs_data1;
gaussdb=# DROP USER jack;
gaussdb=# DROP USER scott;
```

Helpful Links

CREATE DATABASE and **DROP DATABASE**

1.13.7.8 ALTER DATABASE LINK

Description

Modifies database link objects.

Precautions

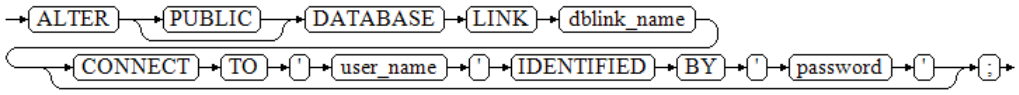
- For details about database links, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If the database link is connected to GaussDB, only the username and password of the database link object can be altered.

Syntax

```
ALTER [PUBLIC] DATABASE LINK dblink_name
   [CONNECT TO 'user_name' IDENTIFIED BY 'password'];
```



Parameters

- **dblink_name**

  Name of a connection.

- **user_name**

  Username for connecting to a remote database.

- **password**

  Password for connecting to a remote database.

- **PUBLIC**

  Connection type. If **PUBLIC** is not specified, the default value **PRIVATE** is used.

Examples

```
-- Create an ORA-compatible data record.
gaussdb=#  CREATE DATABASE ora_test_db DBCOMPATIBILITY 'ORA';
CREATE DATABASE

-- Switch to another database.
gaussdb=# \c ora_test_db

-- Create a user with the SYSADMIN permission.
ora_test_db=# CREATE USER local_user WITH SYSADMIN PASSWORD '********';
CREATE ROLE
ora_test_db=# SET ROLE local_user PASSWORD '********';
SET

-- Create a public database link.
ora_test_db=> CREATE PUBLIC DATABASE LINK public_dblink CONNECT TO 'local_user' IDENTIFIED BY
'********' USING (host '***.***.***.***',port '*****',dbname 'remote_db');
CREATE DATABASE LINK

-- Create a common user.
ora_test_db=> CREATE USER remote_user PASSWORD '********';
CREATE ROLE

-- Change database link object information.
ora_test_db=> ALTER PUBLIC DATABASE LINK public_dblink CONNECT TO 'remote_user' IDENTIFIED BY
'********';
ALTER DATABASE LINK

-- Drop the public database link.
ora_test_db=> DROP PUBLIC DATABASE LINK public_dblink;
DROP DATABASE LINK

-- Drop users.
ora_test_db=> RESET ROLE;
RESET
ora_test_db=# DROP USER local_user;
DROP ROLE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ora_test_db=# DROP USER remote_user;
DROP ROLE

-- Switch back to the initial database and drop the test database. Replace postgres with the actual database name.
ora_test_db=# \c postgres
gaussdb=# DROP DATABASE ora_test_db;
DROP DATABASE
```

Helpful Links

CREATE DATABASE LINK and **DROP DATABASE LINK**

1.13.7.9 ALTER DEFAULT PRIVILEGES

Description

Alters the default permissions of a user on a specific object in the database. It does not affect the permissions assigned to existing objects.

Precautions

Currently, you can change only the permissions for tables (including views), sequences, functions, types, CMKs of encrypted databases, and CEKs.

Syntax

```
ALTER DEFAULT PRIVILEGES
   [ FOR { ROLE | USER } target_role [, ...] ]
   [ IN SCHEMA schema_name [, ...] ]
   abbreviated_grant_or_revoke;
```

- abbreviated_grant_or_revoke grants or revokes permissions on some objects.

  ```
  {grant_on_tables_clause
    | grant_on_sequences_clause
    | grant_on_functions_clause
    | grant_on_types_clause
    | grant_on_client_master_keys_clause
    | grant_on_column_encryption_keys_clause
    | revoke_on_tables_clause
    | revoke_on_sequences_clause
    | revoke_on_functions_clause
    | revoke_on_types_clause
    | revoke_on_client_master_keys_clause
    | revoke_on_column_encryption_keys_clause
  }
  ```

- grant_on_tables_clause grants permissions on tables.

  ```
  GRANT { { SELECT | INSERT | UPDATE | DELETE | TRUNCATE | REFERENCES | ALTER | DROP |
  COMMENT | INDEX | VACUUM }
     [, ...] | ALL [ PRIVILEGES ] }
     ON TABLES
     TO { [ GROUP ] role_name | PUBLIC } [, ...]
     [ WITH GRANT OPTION ]
  ```

- grant_on_sequences_clause grants permissions on sequences.

  ```
  GRANT { { SELECT | UPDATE | USAGE | ALTER | DROP | COMMENT }
     [, ...] | ALL [ PRIVILEGES ] }
     ON SEQUENCES
     TO { [ GROUP ] role_name | PUBLIC } [, ...]
     [ WITH GRANT OPTION ]
  ```

- grant_on_functions_clause grants permissions on functions.

  ```
  GRANT { { EXECUTE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
     ON FUNCTIONS
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    TO { [ GROUP ] role_name | PUBLIC } [, ...]
    [ WITH GRANT OPTION ]
```

- grant_on_types_clause grants permissions on types.

```
GRANT { { USAGE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
    ON TYPES
    TO { [ GROUP ] role_name | PUBLIC } [, ...]
    [ WITH GRANT OPTION ]
```

- grant_on_client_master_keys_clause grants permissions on CMKs.

```
GRANT { { USAGE  | DROP } [, ...] | ALL [ PRIVILEGES ] }
    ON CLIENT_MASTER_KEYS
    TO { [ GROUP ] role_name | PUBLIC } [, ...]
    [ WITH GRANT OPTION ]
```

- grant_on_column_encryption_keys_clause grants permissions on CEKs.

```
GRANT { { USAGE | DROP } [, ...] | ALL [ PRIVILEGES ] }
    ON COLUMN_ENCRYPTION_KEYS
    TO { [ GROUP ] role_name | PUBLIC } [, ...]
    [ WITH GRANT OPTION ]
```

- revoke_on_tables_clause revokes permissions on tables.

```
REVOKE [ GRANT OPTION FOR ]
    { { SELECT | INSERT | UPDATE | DELETE | TRUNCATE | REFERENCES | ALTER | DROP | COMMENT |
INDEX | VACUUM }
    [, ...] | ALL [ PRIVILEGES ] }
    ON TABLES
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT | CASCADE CONSTRAINTS ]
```

- revoke_on_sequences_clause revokes permissions on sequences.

```
REVOKE [ GRANT OPTION FOR ]
    { { SELECT | UPDATE | USAGE | ALTER | DROP | COMMENT }
    [, ...] | ALL [ PRIVILEGES ] }
    ON SEQUENCES
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT | CASCADE CONSTRAINTS ]
```

- revoke_on_functions_clause revokes permissions on functions.

```
REVOKE [ GRANT OPTION FOR ]
    { { EXECUTE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
    ON FUNCTIONS
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT | CASCADE CONSTRAINTS ]
```

- revoke_on_types_clause revokes permissions on types.

```
REVOKE [ GRANT OPTION FOR ]
    { { USAGE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
    ON TYPES
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT | CASCADE CONSTRAINTS ]
```

- revoke_on_client_master_keys_clause revokes permissions on CMKs.

```
REVOKE [ GRANT OPTION FOR ]
    { { USAGE | DROP } [, ...] | ALL [ PRIVILEGES ] }
    ON CLIENT_MASTER_KEYS
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT | CASCADE CONSTRAINTS ]
```

- revoke_on_column_encryption_keys_clause revokes permissions on CEKs.

```
REVOKE [ GRANT OPTION FOR ]
    { { USAGE | DROP } [, ...] | ALL [ PRIVILEGES ] }
    ON COLUMN_ENCRYPTION_KEYS
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT | CASCADE CONSTRAINTS ]
```

Parameters

- **target_role**

  Specifies an existing role name. If **FOR { ROLE | USER}** is not specified, the current role or user is used by default.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: existing role names.

- **schema_name**

  Specifies an existing schema name.

  Value range: existing schema names.

  > NOTE
  >
  > The target role (**target_role**) must have the CREATE permission on the schema (**schema_name**).

- **role_name**

  Specifies the name of the role whose permissions need to be granted or revoked.

  Value range: existing role names.

NOTICE

To drop a role for which the default permissions have been granted, reverse the changes in its default permissions or use DROP OWNED BY to get rid of the default permission entry for the role.

Examples

```
-- Create a schema.
gaussdb=# CREATE SCHEMA tpcds;

-- Grant the SELECT permission on all the tables (and views) in tpcds to every user.
gaussdb=# ALTER DEFAULT PRIVILEGES IN SCHEMA tpcds GRANT SELECT ON TABLES TO PUBLIC;

-- Create a common user jack.
gaussdb=# CREATE USER jack PASSWORD '******';

-- Grant the INSERT permission on all the tables in tpcds to the user jack.
gaussdb=# ALTER DEFAULT PRIVILEGES IN SCHEMA tpcds GRANT INSERT ON TABLES TO jack;

-- Grant the INSERT permission on all tables created by jack in tpcds to user jake (assuming that the user already exists in the database).
gaussdb=#  GRANT USAGE,CREATE ON SCHEMA tpcds TO jack;
gaussdb=# ALTER DEFAULT PRIVILEGES FOR ROLE jack IN SCHEMA tpcds GRANT INSERT ON TABLES TO jake;

-- Revoke the preceding permissions.
gaussdb=# ALTER DEFAULT PRIVILEGES IN SCHEMA tpcds REVOKE SELECT ON TABLES FROM PUBLIC;
gaussdb=# ALTER DEFAULT PRIVILEGES IN SCHEMA tpcds REVOKE INSERT ON TABLES FROM jack;

-- Drop user jack.
gaussdb=# DROP USER jack CASCADE;

-- Drop a schema.
gaussdb=# DROP SCHEMA tpcds;
```

Helpful Links

GRANT and **REVOKE**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.7.10 ALTER DIRECTORY

Description

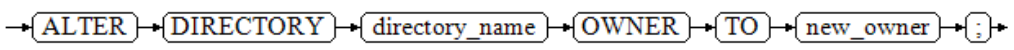
Alters the attributes of a directory.

Precautions

- Currently, only the owner of a directory can be altered.
- When **enable_access_server_directory** is set to **off**, only the initial user is allowed to alter the owner of a directory. When **enable_access_server_directory** is set to **on**, users with the SYSADMIN permission and the directory owner can alter the directory owner, and the they must be a member of the new owner.

Syntax

```
ALTER DIRECTORY directory_name
    OWNER TO new_owner;
```



Parameters

- **directory_name**

  Specifies the directory whose attributes are to be altered.

  Value range: an existing directory name.
- **new_owner**

  Specifies the new owner of the directory.

Examples

```
-- Create a directory.
gaussdb=# CREATE OR REPLACE DIRECTORY dir as '/tmp/';

-- Create a user.
gaussdb=# CREATE USER jim PASSWORD '********';

-- Change the owner of the directory.
gaussdb=# ALTER DIRECTORY dir OWNER TO jim;

-- Delete the directory object.
gaussdb=# DROP DIRECTORY dir;

-- Delete the user.
gaussdb=# DROP USER jim;
```

Helpful Links

CREATE DIRECTORY and **DROP DIRECTORY**

1.13.7.11 ALTER FOREIGN DATA WRAPPER

Description

Alters the attributes of a foreign data wrapper.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

A foreign data wrapper can be altered only during the internal initialization phase or within an upgrade script. In other scenarios, users cannot directly use this syntax.

Syntax

Set the attributes of a foreign data wrapper.

```
ALTER FOREIGN DATA WRAPPER name
[ HANDLER handler_function | NO HANDLER ]
[ VALIDATOR validator_function | NO VALIDATOR ]
[ OPTIONS ( { [ ADD | SET | DROP ] option ['value'] } [, ... ]) ];
```

Parameters

- **name**

  Specifies a name of an existing foreign data wrapper.

- **HANDLER handler_function**

  Specifies a new handler function for a foreign data wrapper.

- **NO HANDLER**

  Specifies that a foreign data wrapper no longer has the handler function.

  NOTICE

  Foreign tables that use foreign data wrapper but do not have handlers cannot be accessed.

- **VALIDATOR validator_function**

  Specifies a new validator function for a foreign data wrapper.

  NOTICE

  Depending on the new validator, an existing option for the foreign data wrapper or dependent server, user mapping, or foreign table may be invalid. Before using foreign data wrapper, ensure that these options are correct. However, any options specified in the **ALTER FOREIGN DATA WRAPPER** command will be checked using the new validator function.

- **NO VALIDATOR**

  Specifies that a foreign data wrapper no longer has the validator function.

- **OPTIONS ([ADD | SET | DROP] option ['value'] [, ...])**

  Alters the options of a foreign data wrapper. The option name must be unique. The actions are ADD, SET, and DROP. If no action is specified, ADD is used. When foreign data is used to wrap validator functions, names and values are also validated.

Helpful Links

CREATE FOREIGN DATA WRAPPER and **DROP FOREIGN DATA WRAPPER**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.7.12 ALTER FOREIGN TABLE

Description

Alters foreign table attributes.

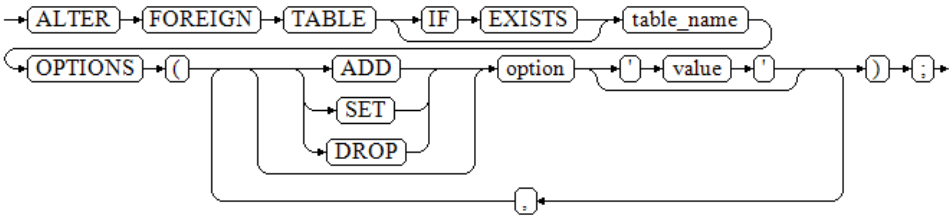
Precautions

When multi-layer quotation marks are used for sensitive columns (such as **password** and **secret_access_key**) in **OPTIONS**, the semantics is different from that in scenarios where quotation marks are not used. Therefore, these columns are not identified as sensitive columns for anonymization.

Syntax

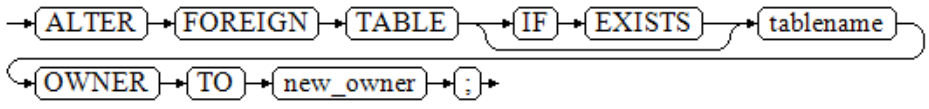
- Set the attributes of a foreign table.

```
ALTER FOREIGN TABLE [ IF EXISTS ]  table_name
    OPTIONS ( { [ ADD | SET | DROP ] option ['value']}[, ... ]);
```



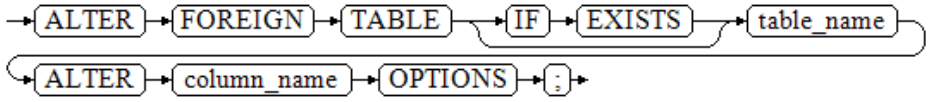
- Set a new owner.

```
ALTER FOREIGN TABLE [ IF EXISTS ] tablename
    OWNER TO new_owner;
```



- Set foreign table column options.

```
ALTER FOREIGN TABLE [ IF EXISTS ] table_name
    ALTER column_name OPTIONS;
```



Parameters

- **table_name**

  Specifies the name of an existing foreign table to be modified.

  Value range: an existing foreign table name.

- **OPTIONS ([ADD | SET | DROP] option ['value'] [, ...])**

  Specifies the option of a foreign table or foreign table column to be altered. **ADD**, **SET**, and **DROP** are operations to be performed. If not specified, **SET** is used by default.

  – **option**: specifies the name of the option to be altered.

    Value range: See **OPTIONS ({ option_name ' value ' } [, ...])** in **CREATE FOREIGN TABLE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - **value**: specifies the new value of **option**.
- **new_owner**

  Specifies the new owner of a foreign table.
- **column_name**

  Specifies the name of an existing column.

  Value range: character strings complying with **Identifier Naming Conventions**.

Examples

```
-- Create a foreign table.
gaussdb=# CREATE SCHEMA tpcds;
gaussdb=# CREATE FOREIGN TABLE tpcds.customer_ft
(
    c_customer_sk             integer               ,
    c_customer_id             char(16)              ,
    c_current_cdemo_sk        integer               ,
    c_current_hdemo_sk        integer               ,
    c_current_addr_sk         integer               ,
    c_first_shipto_date_sk    integer               ,
    c_first_sales_date_sk     integer               ,
    c_salutation              char(10)              ,
    c_first_name              char(20)              ,
    c_last_name               char(30)              ,
    c_preferred_cust_flag     char(1)               ,
    c_birth_day               integer               ,
    c_birth_month             integer               ,
    c_birth_year              integer               ,
    c_birth_country           varchar(20)           ,
    c_login                   char(13)              ,
    c_email_address           char(50)              ,
    c_last_review_date        char(10)
)
    SERVER gsmpp_server
    OPTIONS
(
    location 'gsfs://10.185.179.143:5000/customer1*.dat',
    FORMAT 'TEXT' ,
    DELIMITER '|',
    encoding 'utf8',
    mode 'Normal')
READ ONLY;

-- Change foreign table attributes and delete the mode option.
gaussdb=# ALTER FOREIGN TABLE tpcds.customer_ft options(drop mode);

-- Delete the foreign table.
gaussdb=# DROP FOREIGN TABLE tpcds.customer_ft;
gaussdb=# DROP SCHEMA tpcds CASCADE;
```

Helpful Links

CREATE FOREIGN TABLE and **DROP FOREIGN TABLE**

1.13.7.13 ALTER FUNCTION

Description

Modifies the attributes of a user-defined function or recompiles a function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

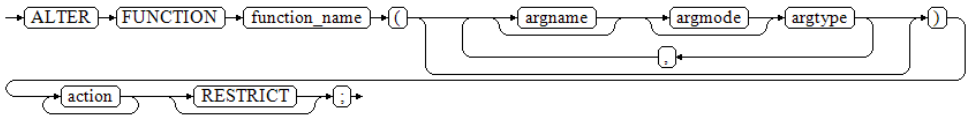
Precautions

- Only the function owner or a user with the ALTER permission can run the **ALTER FUNCTION** command. System administrators have this permission by default. Here are the permission limits for the attributes you want to alter:
  - If a function involves operations on temporary tables, ALTER FUNCTION cannot be used.
  - To change the owner or schema of a function, you must be a function owner or system administrator and a member of the new owner role.
  - Only system administrators and the initial user can change the schema of a function to public.
  - The modification of function attributes is affected by the GUC parameter **using_danger_feature**.
- When separation of duties is enabled, no role is allowed to modify the owner of a function with the definer permission.
- When separation of duties is disabled, only the initial user and system administrators can change the owner of a function that has the definer permission.
- Only the initial user can change the owner of a function to themselves.

Syntax

- Change the additional parameters of the user-defined function.

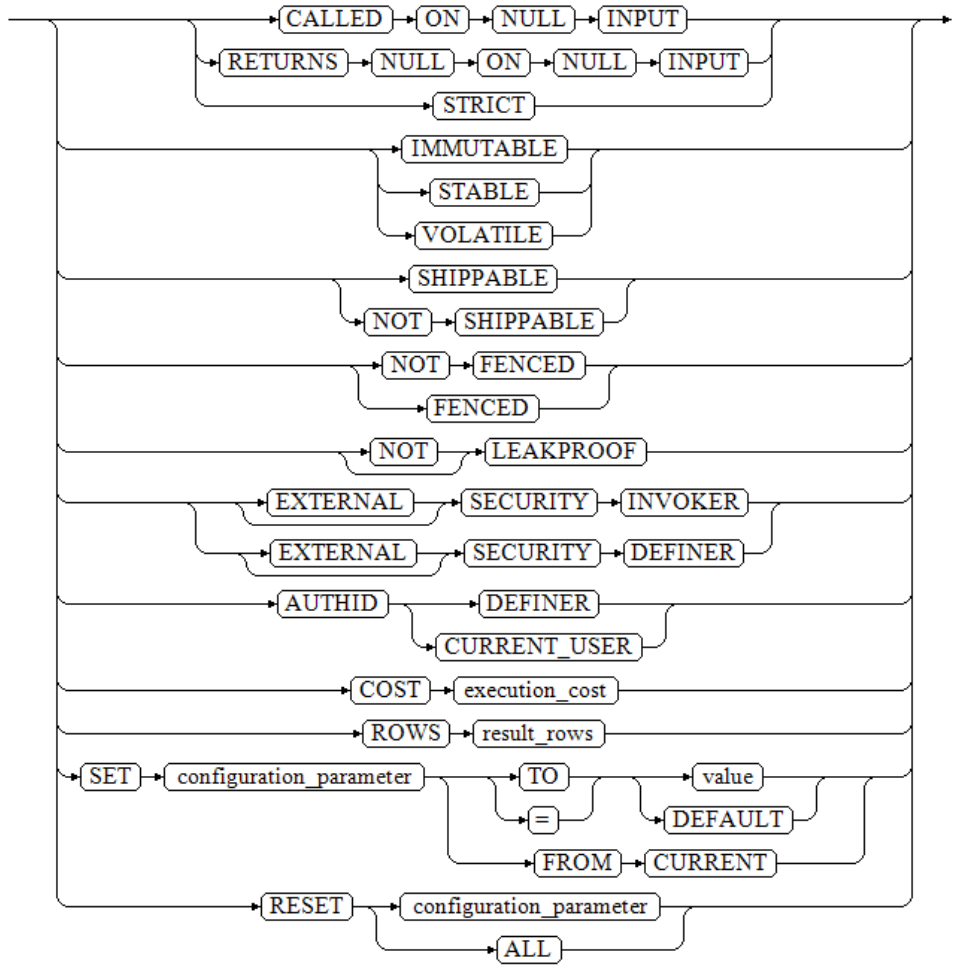
```
ALTER FUNCTION function_name ( [ { [ argname ] [ argmode ] argtype} [, ...] ] )
    action [ ... ] [ RESTRICT ];
```



The syntax of the ACTION clause is as follows:

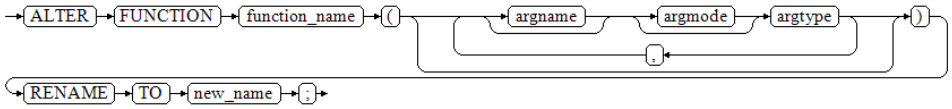
```
{{CALLED ON NULL INPUT | RETURNS NULL ON NULL INPUT | STRICT}
 | {IMMUTABLE | STABLE | VOLATILE}
 | {SHIPPABLE | NOT SHIPPABLE}
 | {NOT FENCED | FENCED}
 | [ NOT ] LEAKPROOF
 | { [ EXTERNAL ] SECURITY INVOKER | [ EXTERNAL ] SECURITY DEFINER }
 | AUTHID { DEFINER | CURRENT_USER }
 | COST execution_cost
 | ROWS result_rows
 | SET configuration_parameter { { TO | = } { value | DEFAULT }| FROM CURRENT}
 | RESET {configuration_parameter | ALL}}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



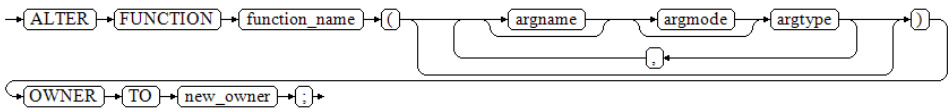
- Rename the user-defined function.

```
ALTER FUNCTION function_name ( [ { [ argname ] [ argmode ] argtype} [, ...] ] )
    RENAME TO new_name;
```



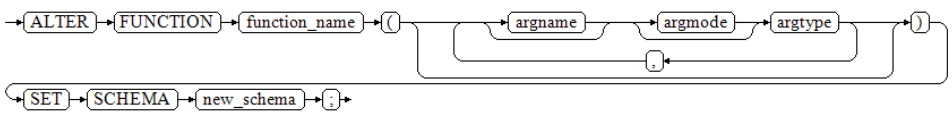
- Alter the owner of the user-defined function.

```
ALTER FUNCTION function_name ( [ { [ argname ] [ argmode ] argtype} [, ...] ] )
    OWNER TO new_owner;
```



- Alter the schema of the user-defined function.

```
ALTER FUNCTION function_name ( [ { [ argname ] [ argmode ] argtype} [, ...] ] )
    SET SCHEMA new_schema;
```



Parameters

- **function_name**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the name of the function to be modified.

Value range: an existing function name

- **argmode**

  Specifies whether a parameter is an input or output parameter.

  Value range:

  - **IN**: declares input parameters.
  - **OUT**: declares output parameters.
  - **INOUT**: declares INOUT parameters.
  - **VARIADIC**: declares parameters of the array type.

- **argname**

  Parameter name.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **argtype**

  Specifies the type of the function parameter.

  Value range: a valid type. For details, see **Data Types**.

- **CALLED ON NULL INPUT**

  Specifies that the function can be called normally when some parameters of the function are NULL. Omitting this parameter is the same as specifying it.

- **RETURNS NULL ON NULL INPUT | STRICT**

  STRICT specifies that the function returns NULL when a parameter of the function is NULL. If this parameter is specified, the function is not executed when there are null parameters; instead a null result is returned automatically.

  RETURNS NULL ON NULL INPUT and **STRICT** have the same functions.

- **IMMUTABLE**

  Specifies that the function always returns the same result when given the same parameter values.

- **STABLE**

  Specifies that the function cannot modify the database, and that within a single table scan it will consistently return the same result for the same parameter value, but its result varies by SQL statements.

- **VOLATILE**

  Specifies that the function value can change in a single table scan and no optimization is performed.

- **SHIPPABLE | NOT SHIPPABLE**

  Specifies whether the function can be pushed down to DNs for execution.

  - Functions of the IMMUTABLE type can always be pushed down to DNs.
  - Functions of the STABLE or VOLATILE type can be pushed down to DNs only if their attribute is **SHIPPABLE**.

- **LEAKPROOF**

  Specifies that the function has no side effect and the parameter contains only the return value. **LEAKPROOF** can be set only by system administrators.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **EXTERNAL**

  (Optional) The purpose is to be compatible with SQL. This keyword applies to all functions, not only external functions.

- **SECURITY INVOKER | AUTHID CURRENT_USER**

  Specifies that the function will be executed with the permissions of the function calling user. Omitting this parameter is the same as specifying it.

  SECURITY INVOKER and **AUTHID CURRENT_USER** have the same functions.

- **SECURITY DEFINER | AUTHID DEFINER**

  Specifies that the function will be executed with the permissions of the function definer.

  AUTHID DEFINER and **SECURITY DEFINER** have the same functions.

- **COST execution_cost**

  Estimates the execution cost of a function.

  The unit of **execution_cost** is **cpu_operator_cost**.

  Value range: a positive integer

- **ROWS result_rows**

  Estimates the number of rows returned by the function. This is allowed when the function is declared to return a set.

  Value range: a positive number. The default value is **1000**.

- **configuration_parameter**
  - **value**

    Sets a specified database session parameter to a specific value.

    Value range: a string.

    - **DEFAULT**: The default system setting is used in the new session.
    - **OFF**: The setting is disabled.
    - **RESET**: The default system setting is used in the new session.
    - *User-specified value*: The value must comply with the parameter modification restrictions.

  - **FROM CURRENT**

    Uses the value of **configuration_parameter** of the current session.

- **new_name**

  Specifies the new name of a function. To change the schema of a function, you must have the CREATE permission on the new schema.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **new_owner**

  Specifies the new owner of a function. To change the owner of a function, the new owner must have the CREATE permission on the schema to which the function belongs. Note that only the initial user can set the owner of a function to the initial user.

  Value range: an existing user role

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **new_schema**

  Specifies the new schema of a function.

  Value range: an existing schema name.

Examples

The following is an example of altering a function:

```
-- Create a function.
gaussdb=# CREATE OR REPLACE FUNCTION test_func(a int) RETURN int
IS
    proc_var int;
BEGIN
    proc_var := a;
    return 1;
END;
/

-- Change the name of the function test_func(a int) to test_func_tk(a int).
gaussdb=# ALTER FUNCTION test_func(a int) RENAME TO test_func_tk;

-- Create user jim.
gaussdb=# CREATE USER jim PASSWORD '********';

-- Create a schema named test.
gaussdb=# CREATE SCHEMA test;

-- Change the owner of the function to jim.
gaussdb=# ALTER FUNCTION test_func_tk(a int) OWNER TO jim;

-- Change the function schema to test.
gaussdb=# ALTER FUNCTION test_func_tk(a int) SET SCHEMA test;

-- Drop the function.
gaussdb=# DROP FUNCTION test.test_func_tk(a int);

-- Drop user jim.
gaussdb=# DROP USER jim;

-- Drop the schema.
gaussdb=# DROP SCHEMA test;
```

Helpful Links

CREATE FUNCTION and **DROP FUNCTION**

1.13.7.14 ALTER GLOBAL CONFIGURATION

Description

Adds or alters **key-value** of the gs_global_config system catalog. You can alter an existing parameter or add a new one.

Precautions

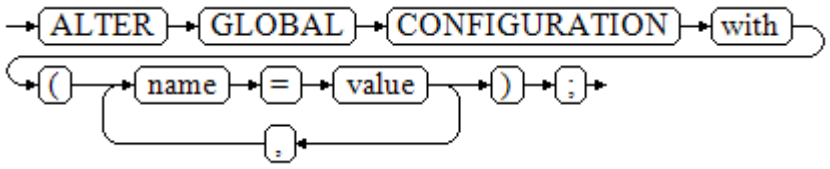
- Only the initial database user can run this command.
- The parameter name cannot be **weak_password** or **undostoragetype**.
- If a parameter name belongs to a parameter reserved for scale-out redistribution, set **enable_cluster_resize** to **on** or use the scale-out tool to modify the name. The involved parameters are **lockwait_timeout**,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

lockwait_interval, **trylock_threshold**, **enable_cancel**, **last_catchup_threshold**, **catchup_times**, **write_error_mode**, **catchup_query_dop**, **parallel_catchup_threshold**, **parallel_reindex_jobs**, **redis_retry_times**, **redis_retry_interval**, **redis_bucket_expansion**, **redis_bucket_batchsize**, **redis_bucket_log_level**, **redis_bucket_lockhold_timeout**, **redis_bucket_timeout_penalty**, **redis_bucket_max_lockwait_timeout_retry**, **redis_bucket_max_lockhold_timeout_retry**, **redis_bucket_strategy_history_capacity**, **redis_bucket_strategy_recent_threshold**, **redis_bucket_workload_rule**, **redis_bucket_lsn_rule**, **redis_bucket_tx_bucket_min**, **redis_bucket_tx_bucket_max**, **redis_max_expected_tps**, **redis_bucket_sender_concurrency**, **redis_bucket_receiver_concurrency**, **redis_bucket_parse_redo_num**, **redis_bucket_workers_per_paser**, **redis_bucket_keep_bucketxlogs**, **redis_bucket_reindex_cbi_offline**, and **redis_bucket_reindex_parallel_workers**.

Syntax

```
ALTER GLOBAL CONFIGURATION with ( { name = value } [, ...] );
```



Parameters

- **name**

  Parameter name, which is of the text type. The value cannot be **weak_password** or **undostoragetype**.

  NOTE

  If a parameter name belongs to a parameter reserved for scale-out redistribution, set **enable_cluster_resize** to **on** or use the scaling tool to modify the name.

- **value**

  Parameter value, which is of the text type.

Examples

```
-- Insert.
gaussdb=# ALTER GLOBAL CONFIGURATION with(redis_is_ok = true);

-- Query.
gaussdb=# SELECT * FROM gs_global_config;
      name       | value
-----------------+-------
 buckets_len     | 16384
 undostoragetype | page
 redis_is_ok     | true
(3 rows)

-- Modify.
gaussdb=# ALTER GLOBAL CONFIGURATION with(redis_is_ok = false);

-- Query.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT * FROM gs_global_config;
      name       | value
-----------------+-------
 buckets_len     | 16384
 undostoragetype | page
 redis_is_ok     | false
(3 rows)

-- Drop.
gaussdb=# DROP GLOBAL CONFIGURATION redis_is_ok;

-- Query.
gaussdb=# SELECT * FROM gs_global_config;
      name       | value
-----------------+-------
 buckets_len     | 16384
 undostoragetype | page
(2 rows)
```

Helpful Links

DROP GLOBAL CONFIGURATION

1.13.7.15 ALTER GROUP

Description

Modifies the attributes of a user group.

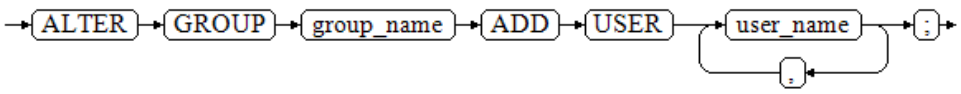
Precautions

- ALTER GROUP is not a standard SQL statement and is not recommended.
- Two clauses (ADD USER and DROP USER) are used to add users to or delete users from a user group. Any user can be a user or a user group. These two clauses are equivalent to granting or revoking the permissions of a user or role to other users or roles. Therefore, you are advised to replace them with GRANT or REVOKE.
- The RENAME TO clause changes the user group name, which is equivalent to renaming roles using ALTER ROLE.

Syntax

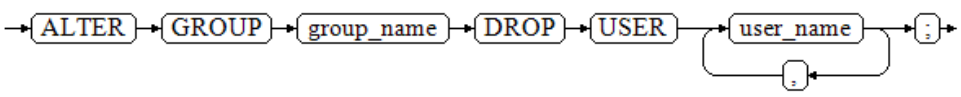
- Add users to a user group.

```
ALTER GROUP group_name
    ADD USER user_name [, ... ];
```



- Drop users from a user group.

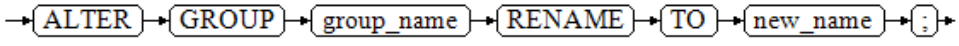
```
ALTER GROUP group_name
    DROP USER user_name [, ... ];
```



- Rename a user group.

```
ALTER GROUP group_name
    RENAME TO new_name;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- **user_name**

  Specifies an existing role name.

  Value range: an existing role name. If a role name contains uppercase letters, enclose the name with double quotation marks ("").

- **group_name**

  Specifies an existing user group name.

  Value range: an existing user group name. If a user group name contains uppercase letters, enclose the name with double quotation marks ("").

- **new_name**

  Specifies a new role name.

  Value range: a string that complies with **Identifier Naming Conventions** and can contain a maximum of 63 characters. If the role name exceeds 63 characters, the database will truncate it to the first 63 characters. If a role name contains uppercase letters, the database automatically converts them to lowercase letters. To create a role name that contains uppercase letters, enclose the name with double quotation marks ("").

Examples

- Rename a user group.

```
-- Create a user test.
gaussdb=# CREATE ROLE test PASSWORD '********';

-- Change the username, which is equivalent to ALTER ROLE RENAME.
gaussdb=# ALTER GROUP test RENAME TO tu_a1;
```

- Add users to or drops users from the user group.

```
-- Create users tu_a2 and tu_a3.
gaussdb=# CREATE ROLE tu_a2 PASSWORD '********';
gaussdb=# CREATE ROLE tu_a3 PASSWORD '********';

-- Add user tu_a2 to the user group tu_a1.
gaussdb=# ALTER GROUP tu_a1 ADD USER tu_a2;

-- The preceding SQL statement is equivalent to the GRANT statement.
gaussdb=# GRANT tu_a1 TO tu_a3;

-- Query.
gaussdb=# SELECT groname, grolist FROM pg_group WHERE groname = 'tu_a1';
 groname |    grolist
---------+---------------
 tu_a1   | {25590,25593}
(1 row)
gaussdb=# SELECT rolname, oid FROM gs_roles WHERE oid IN (25590,25593);
 rolname |  oid
---------+-------
 tu_a2   | 25590
 tu_a3   | 25593
(2 rows)

-- Drop.
gaussdb=# DROP ROLE tu_a1,tu_a2,tu_a3;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

CREATE GROUP, **DROP GROUP**, and **ALTER ROLE**

1.13.7.16 ALTER INDEX

Description

Modifies the definition of an existing index.

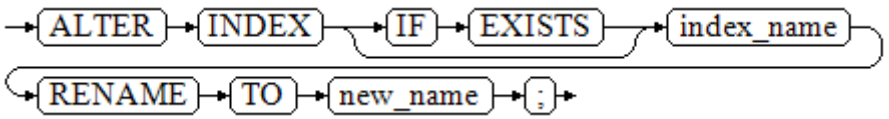
Precautions

- Only the index owner, a user who has the INDEX permission on the table where the index resides, or a user who has the ALTER ANY INDEX permission can run this command. When separation of duties is disabled, system administrators have this permission by default.
- Do not keep a large number of invisible indexes on the same base table. Otherwise, the performance of DML operations such as INSERT, UPDATE, and DELETE may be affected.
- For a **rowid** system column index, you can only change the tablespace to which the index belongs and rebuild the index or index partition.

Syntax

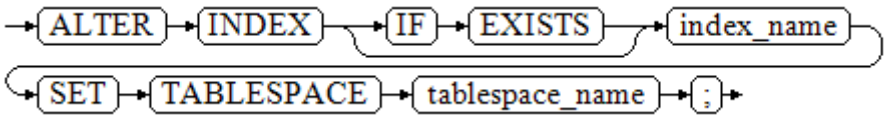
- Rename a table index.

```
ALTER INDEX [ IF EXISTS ] index_name
    RENAME TO new_name;
```



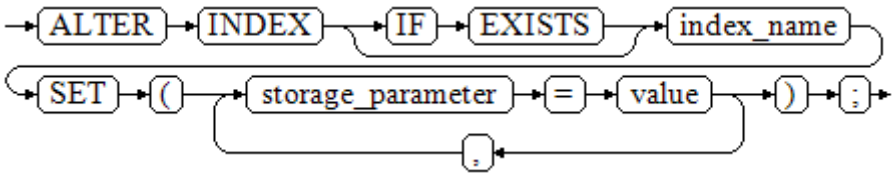
- Change the tablespace to which a table index belongs.

```
ALTER INDEX [ IF EXISTS ] index_name
    SET TABLESPACE tablespace_name;
```



- Change the storage parameter of a table index.

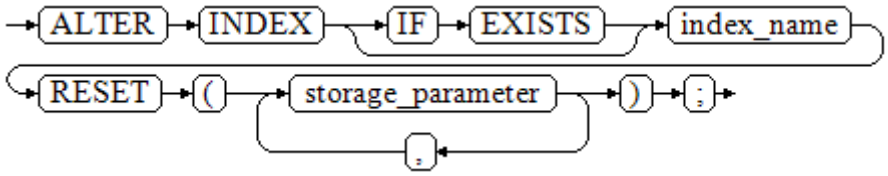
```
ALTER INDEX [ IF EXISTS ] index_name
    SET ( {storage_parameter = value} [, ... ] );
```



- Reset the storage parameter of a table index.

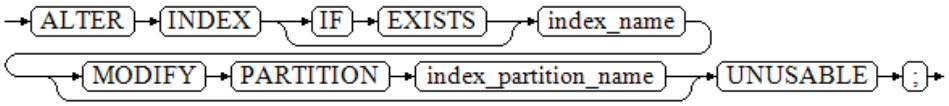
```
ALTER INDEX [ IF EXISTS ] index_name
    RESET ( storage_parameter [, ... ] );
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



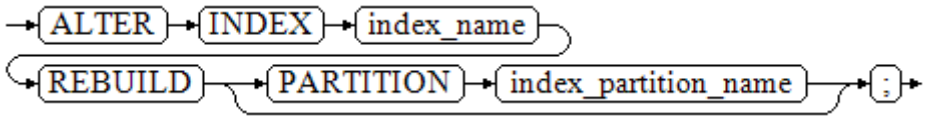
- Set a table index or an index partition to be unavailable.

```
ALTER INDEX [ IF EXISTS ] index_name
    [ MODIFY PARTITION index_partition_name ] UNUSABLE;
```



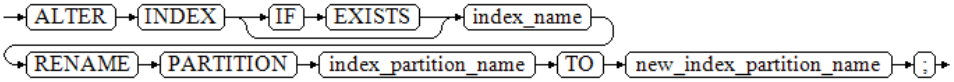
- Rebuild a table index or index partition.

```
ALTER INDEX index_name
    REBUILD [ PARTITION index_partition_name ];
```



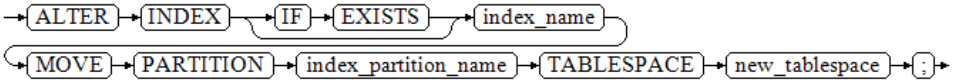
- Rename an index partition.

```
ALTER INDEX [ IF EXISTS ] index_name
    RENAME PARTITION index_partition_name TO new_index_partition_name;
```



- Change the tablespace to which an index partition belongs.

```
ALTER INDEX [ IF EXISTS ] index_name
    MOVE PARTITION index_partition_name TABLESPACE new_tablespace;
```



- Set the distributed global secondary index status.

```
ALTER INDEX [ IF EXISTS ] index_name GSIVALID;
```



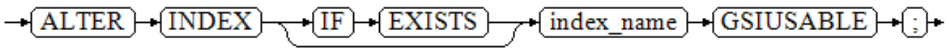

```
ALTER INDEX [ IF EXISTS ] index_name GSIREADY;
```



> **NOTE**
>
> This syntax is called internally by the CREATE GLOBAL INDEX CONCURRENTLY/ REINDEX INDEX CONCURRENTLY function to modify the distributed global secondary index status. You are advised not to use this syntax. If it is used, an error message is displayed.

- Set the distributed global secondary index to be available.

```
ALTER INDEX [ IF EXISTS ] index_name GSIUSABLE;
```



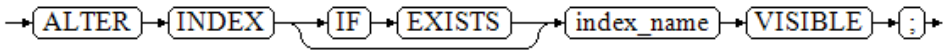
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

This syntax is internally called by the VACUUM FULL function to change the status of a global secondary index. You are advised not to use this syntax. Otherwise, the index data may be inconsistent with the table data. To enable a GSI in the UNUSABLE state, you are advised to use the REINDEX INDEX syntax to rebuild the GSI.

- Set the index to be visible.

```
ALTER INDEX [ IF EXISTS ] index_name VISIBLE;
```



NOTE

- This syntax can be set only in the ORA-compatible database (that is, **sql_compatibility** is set to **'ORA'**).
- When **disable_keyword_options** is set to **"visible"**, the VISIBLE keyword cannot be used.
- This syntax is not supported in the upgrade uncommitted phase.

- Set the index to be invisible.

```
ALTER INDEX [ IF EXISTS ] index_name INVISIBLE;
```

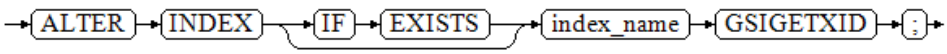


NOTE

- This syntax can be set only in the ORA-compatible database (that is, **sql_compatibility** is set to **'ORA'**).
- When **disable_keyword_options** is set to **"invisible"**, the INVISIBLE keyword cannot be used.
- This syntax is not supported in the upgrade uncommitted phase.
- In the read on standby scenario, after the index is set to invisible, the execution plan of the query statement may change, and the query performance of the standby node may be affected.

- Record the next unallocated xid and csn in reloption of the temporary user table on each node for pruning when temporary tables are merged.

```
ALTER INDEX [ IF EXISTS ] index_name GSIGETXID;
```

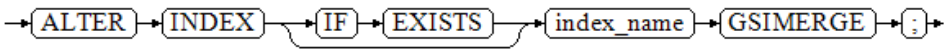


NOTE

This syntax is called internally by the CREATE GLOBAL INDEX CONCURRENTLY/ REINDEX INDEX CONCURRENTLY function. You are advised not to use this syntax. If it is used, an error message is displayed.

- Merge distributed global secondary indexes and temporary user tables.

```
ALTER INDEX [ IF EXISTS ] index_name GSIMERGE;
```



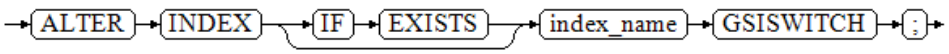
NOTE

This syntax is called internally by the CREATE GLOBAL INDEX CONCURRENTLY/ REINDEX INDEX CONCURRENTLY function. You are advised not to use this syntax. If it is used, an error message is displayed.

- Switch between the old and new distributed global secondary indexes during online rebuilding of distributed global secondary indexes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ALTER INDEX [ IF EXISTS ] index_name GSISWITCH;
```



NOTE

This syntax is called internally by the CREATE GLOBAL INDEX CONCURRENTLY/ REINDEX INDEX CONCURRENTLY function. You are advised not to use this syntax. If it is used, an error message is displayed.

Parameters

- **index_name**

  Specifies the index name to be modified.

- **IF EXISTS**

  Issues a notice rather than an error if the specified index does not exist.

- **RENAME TO new_name**

  Changes only the name of the index without affecting stored data.

  – **new_name**

    Specifies a new index name.

    Value range: character strings complying with **Identifier Naming Conventions**.

- **SET TABLESPACE tablespace_name**

  Changes the index tablespace to the specified tablespace and moves index-related data files to the new tablespace.

  – **tablespace_name**

    Specifies the tablespace name.

    Value range: an existing tablespace name.

- **SET ({storage_parameter = value} [, ...])**

  Changes one or more index-method-specific storage parameters of an index. Note that the index content will not be modified immediately by this statement. You may need to use REINDEX to rebuild the index based on different parameters to achieve the expected effect.

  – **storage_parameter**

    Specifies the name of an index-method-specific parameter. **ACTIVE_PAGES** indicates the number of index pages, which may be less than the actual number of physical file pages and can be used for optimization. Currently, this parameter is valid only for the local index of the Ustore partitioned table and will be updated by VACUUM and ANALYZE (including AUTOVACUUM). You are advised not to manually set this parameter because it is invalid in distributed mode. When ALTER is executed on GSI, the **INTERNAL_MASK**, **APPEND_MODE_INTERNAL**, **WAIT_CLEAN_GPI**, **PARTITION_DDL_FLAG**, **COLLATE**, **CROSSBUCKET**, and **DEDUPLICATION** parameters are not supported. The behavior is the same as that of UB-tree. The **FILLFACTOR**, and **INDEXSPLIT** parameters are supported. **ACTIVE_PAGES** and **WAIT_CLEAN_CBI** are not applicable to GSI.

  – **value**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the new value for an index-method-specific storage parameter. This might be a number or a word depending on the parameter.

- **RESET ({ storage_parameter } [, ...])**

  Resets one or more index-method-specific storage parameters of an index to the default value. Similar to the SET statement, REINDEX may be used to completely update the index.

- **[MODIFY PARTITION index_partition_name] UNUSABLE**

  Sets the indexes on a table or index partition to be unavailable.

  NOTE

  If a unique index is specified, the subsequent insert and update operations on the table are controlled by the GUC parameter **enable_unique_checking_of_unusable_index**.

- **REBUILD [PARTITION index_partition_name]**

  Rebuilds indexes on a table or an index partition. If the index contains the **lpi_parallel_method** option and the value is **PARTITION** when the index is rebuilt, and the **parallel_workers** value of the index's table is greater than 0, the index cannot be rebuilt in parallel. If the index does not contain the **lpi_parallel_method** option or the value of the option is set to **AUTO**, page-level parallel index is rebuilt by default. For details, see **LPI_PARALLEL_METHOD** in **CREATE INDEX**.

- **RENAME PARTITION index_partition_name TO new_index_partition_name**

  Renames an index partition.

- **MOVE PARTITION index_partition_name TABLESPACE new_tablespace**

  Modifies the tablespace to which an index partition belongs.

- **new_index_partition_name**

  Specifies a new index partition name.

- **index_partition_name**

  Specifies an index partition name.

- **new_tablespace**

  Specifies a new tablespace.

- **GSIVALID** | GSIREADY

  Internally called by the CREATE GLOBAL INDEX CONCURRENTLY or REINDEX INDEX CONCURRENTLY function to change the status of distributed global secondary indexes.

- **GSIUSABLE**

  This syntax is internally called by the VACUUM FULL function to modify the status of distributed global secondary indexes.

- **VISIBLE**

  Sets the index to be visible.

- **INVISIBLE**

  Set the index to be invisible.

- **GSIGETXID**

  Internally called by the CREATE GLOBAL INDEX CONCURRENTLY/REINDEX INDEX CONCURRENTLY function. It records the next unallocated xid and csn

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

in reloption of the temporary user table on each node for pruning when temporary tables are merged.

- **GSIMERGE**

  Internally called by the CREATE GLOBAL INDEX CONCURRENTLY/REINDEX INDEX CONCURRENTLY function to merge distributed global secondary indexes and temporary user tables.

- **GSISWITCH**

  Internally called by the CREATE GLOBAL INDEX CONCURRENTLY/REINDEX INDEX CONCURRENTLY function to switch between the old and new distributed global secondary indexes during online rebuilding of distributed global secondary indexes.

Examples

- Rename an index.

```
-- Create the test1 table and create an index for it.
gaussdb=# CREATE TABLE test1(col1 int, col2 int);
gaussdb=# CREATE INDEX aa ON test1(col1);

-- Rename the aa index as idx_test1_col1.
gaussdb=# ALTER INDEX aa RENAME TO idx_test1_col1;

-- Query the index information of the test1 table.
gaussdb=# SELECT tablename,indexname,tablespace FROM pg_indexes WHERE tablename = 'test1';
 tablename |   indexname    | tablespace
-----------+----------------+------------
 test1     | idx_test1_col1 |
(1 row)
```

- Change the tablespace to which the index belongs.

```
-- Create the tbs_index1 tablespace.
gaussdb=# CREATE TABLESPACE tbs_index1 RELATIVE LOCATION 'tablespace1/tbs_index1';

-- Change the tablespace to which the idx_test1_col1 index belongs to tbs_index1.
gaussdb=# ALTER INDEX IF EXISTS idx_test1_col1 SET TABLESPACE tbs_index1;

-- Query the index information of the test1 table.
gaussdb=# SELECT tablename,indexname,tablespace FROM pg_indexes WHERE tablename = 'test1';
 tablename |   indexname    | tablespace
-----------+----------------+------------
 test1     | idx_test1_col1 | tbs_index1
(1 row)
```

- Change and reset index storage parameters.

```
-- View details about the idx_test1_col1 index.
gaussdb=# \di idx_test1_col1                        List of relations
 Schema |      Name      | Type  | Owner | Table | Storage
--------+----------------+-------+-------+-------+---------
 public | idx_test1_col1 | index | omm   | test1 |
(1 row)

-- Change the fillfactor of the idx_test1_col1 index.
gaussdb=# ALTER INDEX IF EXISTS idx_test1_col1 SET (FILLFACTOR = 70);
-- View details about the idx_test1_col1 index.
gaussdb=# \di idx_test1_col1
                        List of relations
 Schema |      Name      | Type  | Owner | Table |     Storage
--------+----------------+-------+-------+-------+-----------------
 public | idx_test1_col1 | index | omm   | test1 | {fillfactor=70}
(1 row)
-- Reset the storage parameter of the idx_test1_col1 index.
gaussdb=# ALTER INDEX IF EXISTS idx_test1_col1 RESET (FILLFACTOR);
-- View details about the idx_test1_col1 index.
gaussdb=# \di idx_test1_col1
                    List of relations
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 Schema |      Name      | Type  | Owner | Table | Storage
--------+----------------+-------+-------+-------+---------
 public | idx_test1_col1 | index | omm   | test1 |
(1 row)
```

- Change the index availability.

```
-- Set the idx_test1_col1 index to be unavailable.
gaussdb=# ALTER INDEX IF EXISTS idx_test1_col1 UNUSABLE;
-- Check the availability of the idx_test1_col1 index.
gaussdb=# SELECT indisusable FROM pg_index WHERE indexrelid = 'idx_test1_col1'::regclass;
 indisusable
-------------
 f
(1 row)

-- Rebuild the idx_test1_col1 index.
gaussdb=# ALTER INDEX idx_test1_col1 REBUILD;
-- Check the availability of the idx_test1_col1 index.
gaussdb=# SELECT indisusable FROM pg_index WHERE indexrelid = 'idx_test1_col1'::regclass;
 indisusable
-------------
 t
(1 row)

-- Drop.
gaussdb=# DROP INDEX idx_test1_col1;
gaussdb=# DROP TABLE test1;
gaussdb=# DROP TABLESPACE tbs_index1;
```

- Rename an index partition.

```
-- Create a partitioned table named test2.
gaussdb=# CREATE TABLE test2(col1 int, col2 int) PARTITION BY RANGE (col1)(
    PARTITION p1 VALUES LESS THAN (100),
    PARTITION p2 VALUES LESS THAN (200)
);
-- Create a partitioned index.
gaussdb=# CREATE INDEX idx_test2_col1 ON test2(col1) LOCAL(
    PARTITION p1,
    PARTITION p2
);
-- Rename an index partition.
gaussdb=# ALTER INDEX idx_test2_col1 RENAME PARTITION p1 TO p1_test2_idx;
gaussdb=# ALTER INDEX idx_test2_col1 RENAME PARTITION p2 TO p2_test2_idx;

-- Query the partition name of the idx_test2_col1 index.
gaussdb=# SELECT relname FROM pg_partition WHERE parentid = 'idx_test2_col1'::regclass;
   relname
--------------
 p1_test2_idx
 p2_test2_idx
(2 rows)
```

- Change the tablespace to which an index partition belongs.

```
-- Create tablespaces tbs_index2 and tbs_index3.
gaussdb=# CREATE TABLESPACE tbs_index2 RELATIVE LOCATION 'tablespace1/tbs_index2';
gaussdb=# CREATE TABLESPACE tbs_index3 RELATIVE LOCATION 'tablespace1/tbs_index3';

-- Change the tablespace to which the idx_test2_col1 index partition belongs.
gaussdb=# ALTER INDEX idx_test2_col1 MOVE PARTITION p1_test2_idx TABLESPACE tbs_index2;
gaussdb=# ALTER INDEX idx_test2_col1 MOVE PARTITION p2_test2_idx TABLESPACE tbs_index3;

-- Query the tablespace to which the idx_test2_col1 index partition belongs.
gaussdb=# SELECT t1.relname index_name,
       t2.spcname tablespace_name
FROM pg_partition t1, pg_tablespace t2
WHERE t1.parentid = 'idx_test2_col1'::regclass AND
      t1.reltablespace = t2.oid;
  index_name  | tablespace_name
--------------+-----------------
 p1_test2_idx | tbs_index2
 p2_test2_idx | tbs_index3
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(2 rows)

-- Drop.
gaussdb=# DROP INDEX idx_test2_col1;
gaussdb=# DROP TABLE test2;
gaussdb=# DROP TABLESPACE tbs_index2;
gaussdb=# DROP TABLESPACE tbs_index3;
```

Helpful Links

CREATE INDEX, **DROP INDEX**, and **REINDEX**

1.13.7.17 ALTER LANGUAGE

Description

Modifies the definition of a procedural language.

Precautions

This syntax is not supported currently.

1.13.7.18 ALTER MASKING POLICY

Description

Modifies a masking policy.

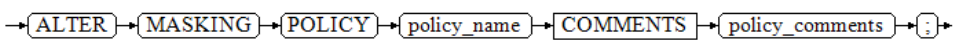
Precautions

- Only users with the POLADMIN or SYSADMIN permission, or the initial user can perform this operation.
- The masking policy takes effect only after the security policy is enabled, that is, **enable_security_policy** is set to **on**.

Syntax

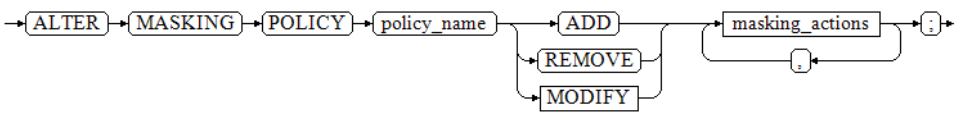
- Modify the policy description.

```
ALTER MASKING POLICY policy_name COMMENTS policy_comments;
```



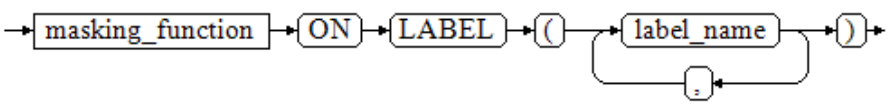
- Change the masking method.

```
ALTER MASKING POLICY policy_name { ADD | REMOVE | MODIFY } masking_actions[, ...];
```



  - The syntax of **masking_actions**

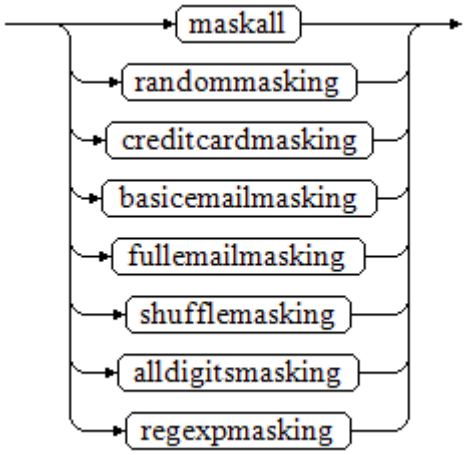
```
masking_function ON LABEL(label_name[, ...])
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

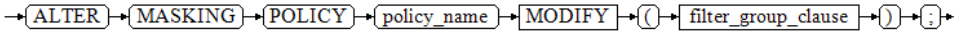
  - The syntax of **masking_function**

    ```
    { maskall | randommasking | creditcardmasking | basicemailmasking | fullemailmasking |
    shufflemasking | alldigitsmasking | regexpmasking }
    ```

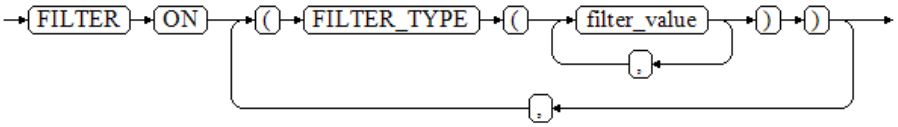
- Modify the scenarios where the masking policies take effect.

  ```
  ALTER MASKING POLICY policy_name MODIFY (filter_group_clause);
  ```

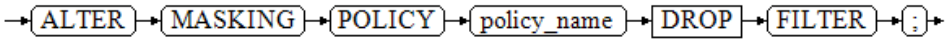
  - The syntax of filter_group_clause

    ```
    FILTER ON { ( FILTER_TYPE ( filter_value [, ... ] ) ) [, ... ] }
    ```

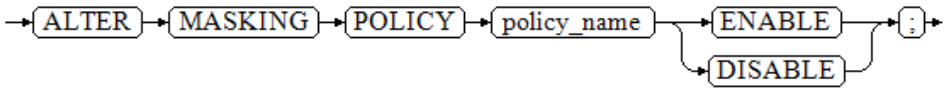
- Remove the filters of the masking policies for the policies to take effect.

  ```
  ALTER MASKING POLICY policy_name DROP FILTER;
  ```

- Enable or disable the masking policies.

  ```
  ALTER MASKING POLICY policy_name {ENABLE | DISABLE};
  ```

Parameters

- **policy_name**

  Specifies the masking policy name, which must be unique.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **policy_comments**

  Adds or modifies description of masking policies.

- **masking_function**

  Specifies eight preset masking methods or user-defined functions. Schema is supported.

  The maskall function is not preset. It is in the code and cannot be displayed by running **\df**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The preset masking methods are as follows:

```
{ maskall | randommasking | creditcardmasking | basicemailmasking | fullemailmasking |
shufflemasking | alldigitsmasking | regexpmasking }
```

- **label_name**

  Specifies the resource label name.

- **FILTER_TYPE**

  Specifies the types of information to be filtered by the policies: **IP**, **ROLES**, and **APP**.

- **filter_value**

  Indicates the detailed information to be filtered, such as the IP address, app name, and username.

- **ENABLE|DISABLE**

  Enables or disables the masking policy. If **ENABLE|DISABLE** is not specified, **ENABLE** is used by default.

Examples

- Modify the policy description.

```
-- Create table tb_for_masking.
gaussdb=# CREATE TABLE tb_for_masking(idx int, col1 text, col2 text, col3 text, col4 text, col5 text,
col6 text, col7 text,col8 text);

-- Insert data into the tb_for_masking table.
gaussdb=# INSERT INTO tb_for_masking VALUES(1, '9876543210', 'usr321usr', 'abc@huawei.com',
'abc@huawei.com', '1234-4567-7890-0123', 'abcdef 123456 ui 323 jsfd321 j3k2l3',
'4880-9898-4545-2525', 'this is a llt case');

-- View data.
gaussdb=# SELECT * FROM tb_for_masking;
 idx |    col1    |   col2    |      col3      |      col4      |       col5         |               col6
     |       col7         |       col8
-----+------------+-----------+----------------+----------------+--------------------
+------------------------------
------+--------------------+--------------------
   1 | 9876543210 | usr321usr | abc@huawei.com | abc@huawei.com | 1234-4567-7890-0123 | abcdef
123456 ui 323 jsfd321 j
3k2l3 | 4880-9898-4545-2525 | this is a llt case
(1 row)

-- Create a resource label for sensitive column col1.
gaussdb=# CREATE RESOURCE LABEL mask_lb1 ADD COLUMN(tb_for_masking.col1);

-- Create a data masking policy named maskpol1.
gaussdb=# CREATE MASKING POLICY maskpol1 maskall ON LABEL(mask_lb1);

-- Add description for masking policy maskpol1.
gaussdb=# ALTER MASKING POLICY maskpol1 COMMENTS 'masking policy for tb_for_masking.col1';

-- View the description of data masking policy maskpol1.
gaussdb=# SELECT * FROM GS_MASKING_POLICY;
 polname  |             polcomments              |         modifydate         | polenabled
----------+--------------------------------------+----------------------------+------------
 maskpol1 | masking policy for tb_for_masking.col1 | 2023-11-07 16:38:31.607374 | t
(1 row)
```

- Change a masking policy.

```
-- Create a resource label for sensitive column col2.
gaussdb=# CREATE RESOURCE LABEL mask_lb2 ADD COLUMN(tb_for_masking.col2);

-- Add the randommasking masking mode to the masking policy maskpol1.
gaussdb=# ALTER MASKING POLICY maskpol1 ADD randommasking ON LABEL(mask_lb2);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Access the tb_for_masking table. The masking policy is triggered for the col2 column.
gaussdb=# SELECT col2 FROM tb_for_masking;
    col2
------------
 27e8da66cc
(1 row)

-- Remove the randommasking masking mode from the masking policy maskpol1.
gaussdb=# ALTER MASKING POLICY maskpol1 REMOVE randommasking ON LABEL(mask_lb2);

-- Access the tb_for_masking table. The data in the col2 column is not masked, indicating that the
masking mode randommasking is invalid.
gaussdb=# SELECT col2 FROM tb_for_masking;
   col2
-----------
 usr321usr
(1 row)

-- Change the masking mode of the masking policy maskpol1 to randommasking.
gaussdb=# ALTER MASKING POLICY maskpol1 MODIFY randommasking ON LABEL(mask_lb1);

-- Access the tb_for_masking table. The masking policy is triggered for the col1 column.
gaussdb=# SELECT col1 FROM tb_for_masking;
    col1
------------
 5a03debac1
(1 row)
```

- Modify the scenarios where the masking policies take effect.

```
-- Create users dev_mask and bob_mask.
gaussdb=# CREATE USER dev_mask PASSWORD '********';
gaussdb=# CREATE USER bob_mask PASSWORD '********';

-- Create a resource label for sensitive column col8.
gaussdb=# CREATE RESOURCE LABEL mask_lb8 ADD COLUMN(tb_for_masking.col8);

-- Create a data masking policy named maskpol8.
gaussdb=# CREATE MASKING POLICY maskpol8 randommasking ON LABEL(mask_lb8) FILTER ON
ROLES(dev_mask, bob_mask), APP(gsql), IP('172.31.17.160', '127.0.0.0/24');

-- Change the filtering information by specifying ROLES of data masking policy maskpol8.
gaussdb=# ALTER MASKING POLICY maskpol1 MODIFY (FILTER ON ROLES(dev_mask));

-- Check tb_for_masking as the dev_mask user.
gaussdb=# GRANT ALL PRIVILEGES TO dev_mask;

-- Access the tb_for_masking table. The masking policy is triggered for the col8 column.
gaussdb=# SELECT col8 FROM tb_for_masking;
        col8
--------------------
 f134e06ef528013b46
(1 row)
```

- Remove the filters of the masking policies for the policies to take effect.

```
gaussdb=# ALTER MASKING POLICY maskpol1 DROP FILTER;
```

- Disable a masking policy.

```
-- Disable masking policy maskpol1.
gaussdb=# ALTER MASKING POLICY maskpol1 DISABLE;

-- Check the status of data masking policy maskpol1. If the value of polenabled is f, the data
masking policy is disabled successfully.
gaussdb=# SELECT * FROM GS_MASKING_POLICY;
 polname  | polcomments |         modifydate         | polenabled
----------+-------------+----------------------------+------------
 maskpol1 |             | 2023-11-07 17:22:54.594111 | f
```

- Delete data.

```
-- Drop masking policies.
gaussdb=# DROP MASKING POLICY maskpol1, maskpol8;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop resource labels.
gaussdb=# DROP RESOURCE LABEL mask_lb1, mask_lb2, mask_lb8;

-- Drop the tb_for_masking table.
gaussdb=# DROP TABLE tb_for_masking;

-- Drop the dev_mask and bob_mask users.
gaussdb=# DROP USER dev_mask, bob_mask;
```

Helpful Links

CREATE MASKING POLICY and DROP MASKING POLICY

1.13.7.19 ALTER MATERIALIZED VIEW

Description

Modifies multiple auxiliary attributes of an existing materialized view.

Statements and actions that can be used for ALTER MATERIALIZED VIEW are a subset of ALTER TABLE and have the same meaning when used for materialized views. For details, see ALTER TABLE.

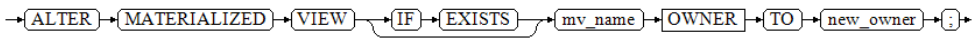
Precautions

- Only the materialized view owner can execute the **ALTER MATERIALIZED VIEW** command. System administrators have this permission by default.
- The materialized view structure cannot be modified.

Syntax

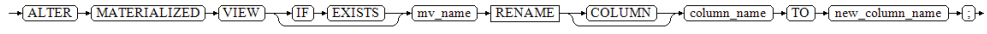
- Change the owner of a materialized view.

```
ALTER MATERIALIZED VIEW [ IF EXISTS ] mv_name
    OWNER TO new_owner;
```



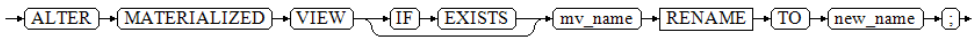
- Rename a column of a materialized view.

```
ALTER MATERIALIZED VIEW [ IF EXISTS ] mv_name
    RENAME [ COLUMN ] column_name TO new_column_name;
```



- Rename a materialized view.

```
ALTER MATERIALIZED VIEW [ IF EXISTS ] mv_name
    RENAME TO new_name;
```



Parameters

- **mv_name**

  Specifies the name of an existing materialized view, which can be schema-qualified.

  Value range: character strings complying with Identifier Naming Conventions.

- **column_name**

  Specifies the name of a new or existing column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: character strings complying with **Identifier Naming Conventions**.

- **new_column_name**

  Specifies the new name of an existing column.

- **new_owner**

  Specifies the username of the new owner of a materialized view.

- **new_name**

  Specifies the new name of a materialized view.

Examples

- Change the owner of a materialized view.

```
-- Create a table.
gaussdb=# CREATE TABLE my_table (c1 int, c2 int) WITH(STORAGE_TYPE=ASTORE);

-- Create a complete-refresh materialized view.
gaussdb=# CREATE MATERIALIZED VIEW foo AS SELECT * FROM my_table;

-- Create a user.
gaussdb=# CREATE USER test PASSWORD '********';

-- Change the owner of the complete-refresh materialized view.
gaussdb=# ALTER MATERIALIZED VIEW foo OWNER TO test;

-- View the information about the materialized view.
gaussdb=# \dm foo
                        List of relations
 Schema | Name |       Type        | Owner |             Storage
--------+------+-------------------+-------+-----------------------------------
 public | foo  | materialized view | test  | {orientation=row,compression=no}
(1 row)
```

- Rename a column of a materialized view.

```
-- Query the columns of the materialized view.
gaussdb=# \d foo;
Materialized view "public.foo"
 Column |  Type   | Modifiers
--------+---------+-----------
 c1     | integer |
 c2     | integer |
Rules:
    "_RETURN" AS
    ON SELECT TO foo DO INSTEAD  SELECT my_table.c1, my_table.c2
   FROM my_table
Replica Identity: NOTHING

-- Change column c1 of materialized view foo to col1 and column c2 to col2.
gaussdb=# ALTER MATERIALIZED VIEW foo RENAME c1 to col1;
gaussdb=# ALTER MATERIALIZED VIEW foo RENAME c2 to col2;

-- Use the SELECT statement to view the columns of the materialized view.
gaussdb=# SELECT * FROM foo WHERE 1=2;
 col1 | col2
------+------
(0 rows)
```

- Rename a materialized view.

```
-- Rename the materialized view foo to my_mview.
gaussdb=# ALTER MATERIALIZED VIEW foo RENAME TO my_mview;

-- Query information.
gaussdb=# \dm my_mview
                        List of relations
 Schema |   Name    |       Type        | Owner |             Storage
--------+-----------+-------------------+-------+-----------------------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 public | my_mview | materialized view | test  | {orientation=row,compression=no}
(1 row)

-- Delete.
gaussdb=# DROP MATERIALIZED VIEW my_mview;
gaussdb=# DROP TABLE my_table ;
gaussdb=# DROP USER test;
```

Helpful Links

CREATE INCREMENTAL MATERIALIZED VIEW, **CREATE MATERIALIZED VIEW**, **DROP MATERIALIZED VIEW**, **REFRESH INCREMENTAL MATERIALIZED VIEW**, and **REFRESH MATERIALIZED VIEW**

1.13.7.20 ALTER NODE

Description

Changes the definition of an existing cluster node.

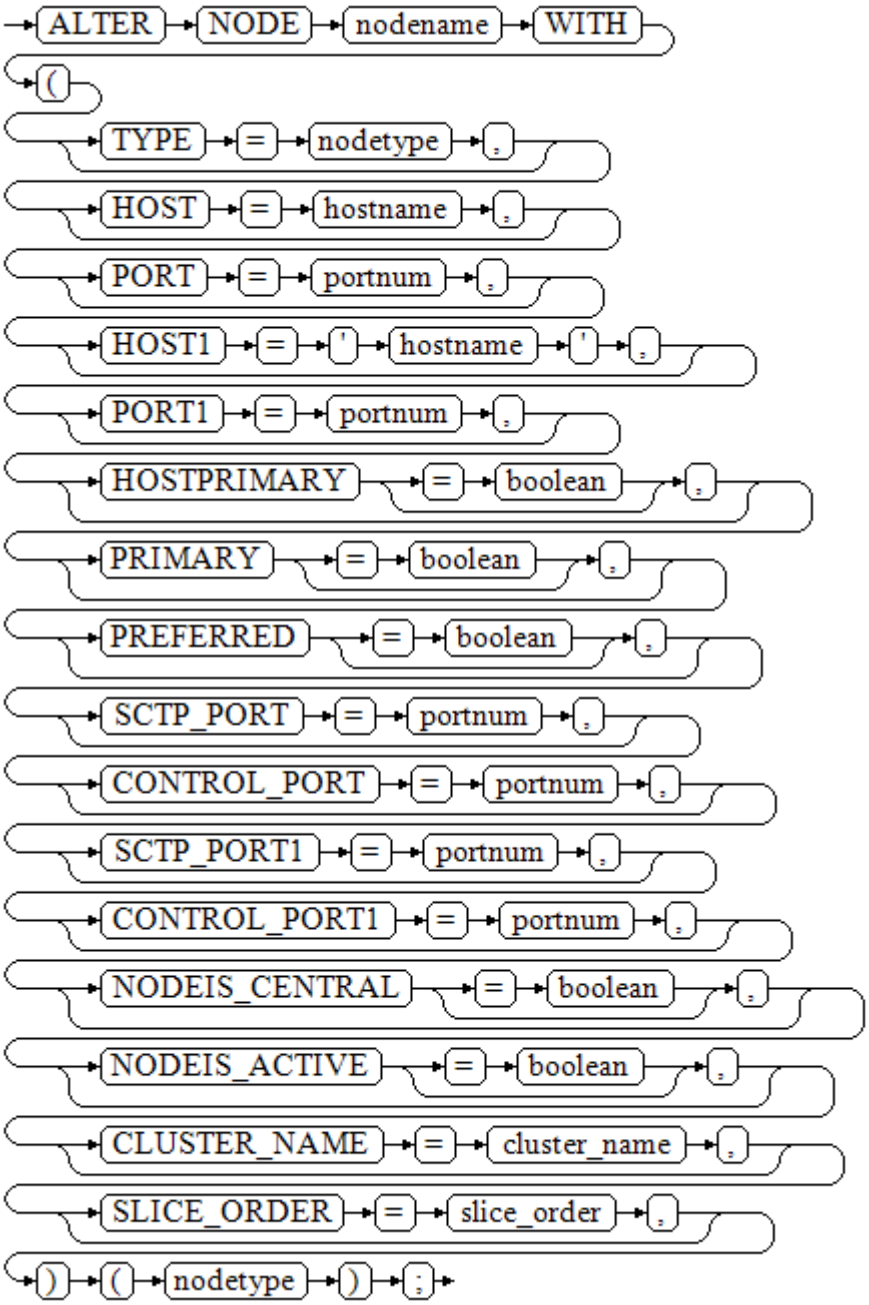
Precautions

ALTER NODE is an API of the cluster management tool and is used to manage clusters. Only administrators have the permission to use this API. Do not use this API directly. Otherwise, the cluster status may be affected and the database may be unavailable.

Syntax

```
ALTER NODE nodename WITH
  (
    [ TYPE = nodetype,]
    [ HOST = hostname,]
    [ PORT = portnum,]
    [ HOST1 = 'hostname',]
    [ PORT1 = portnum,]
    [ HOSTPRIMARY [ = boolean ],]
    [ PRIMARY [ = boolean ],]
    [ PREFERRED [ = boolean ],]
    [ SCTP_PORT = portnum,]
    [ CONTROL_PORT = portnum,]
    [ SCTP_PORT1 = portnum,]
    [ CONTROL_PORT1 = portnum, ]
    [ NODEIS_CENTRAL [ = boolean ],]
    [ NODEIS_ACTIVE [= boolean],]
    [ CLUSTER_NAME = cluster_name, ]
    [ SLICE_ORDER = slice_order, ]
  ) (nodetype);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



NOTE

The port whose number is specified by **PORT** is used for internal communications between nodes. Unlike the port connecting to an external client, it can be queried in the pgxc_node table.

Parameters

- **nodename**

  Node name.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **TYPE = nodetype**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the type of a node.

Value range:

- 'coordinator'
- 'datanode'
- 'coordinator_standbycluster'
- 'datanode_standbycluster_main'
- 'datanode_standbycluster_other'

- **HOST = hostname**

  Specifies the name or IP address of the primary node corresponding to a node.

- **PORT = portnum**

  Specifies the port number of the primary node to which a node is bound.

- **HOST1 = hostname**

  Specifies the name or IP address of the standby node corresponding to a node.

- **PORT1 = portnum**

  Specifies the port number of the standby node to which a node is bound.

- **HOSTPRIMARY**

  Specifies whether the host name or IP address is the primary one.

  Value range:

  - **true**
  - **false** (default value)

- **PRIMARY = boolean**

  Specifies whether the node is a primary node or not. A primary node allows read/write operations. A non-primary node allows only read operations.

  Value range:

  - **true**
  - **false** (default value)

- **PREFERRED = boolean**

  Specifies whether the node is a preferred node for read operations.

  Value range:

  - **true**
  - **false** (default value)

- **SCTP_PORT = portnum**

  Specifies the port used by the TCP proxy communications library of the primary node to listen on the data transmission channel. TCP is used to listen on connections.

- **CONTROL_PORT = portnum**

  Specifies the port used by the TCP proxy communications library of the primary node to listen on the control transmission channel. The TCP protocol is used to listen on connections.

- **SCTP_PORT1 = portnum**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the port used by the TCP proxy communications library of the standby node to listen on the data transmission channel. TCP is used to listen on connections.

- **CONTROL_PORT 1= portnum**

  Specifies the port used by the TCP proxy communications library of the standby node to listen on the control transmission channel. The TCP protocol is used to listen on connections.

- **NODEIS_CENTRAL**

  Specifies whether the current node is a central control node. It is used only for CNs and is invalid for DNs.

  – **true**

  – **false** (default value)

- **NODEIS_ACTIVE**

  Specifies whether the current node is normal. It is used to mark whether the CN is removed and is invalid for DNs.

  – **true**

  – **false** (default value)

- **CLUSTER_NAME**

  Cluster name.

- **SLICE_ORDER**

  Specifies the sequence number of the slice to which a node belongs.

- **nodetype**

  Type of the node to be changed, which is the value of **node_type** in the pgxc_node system catalog.

  Value range: **"C"** (CN)

  NOTE

  In multi-CN environments, when you create a cluster node, it is only generated on the CN that is currently connected and is not synchronized with other CNs. This requires you to repeat the creation process on all CNs. However, to drop the cluster node created, you only need to connect to one CN. If a cluster node is created on only one CN, you need to define the current CN name and drop the cluster node on the current CN.

Examples

```
-- Create cluster nodes on all CNs.
gaussdb=# CREATE NODE datanode1 WITH(
   TYPE = datanode,
   PREFERRED = false
);

-- Query the initial cluster DN status.
gaussdb=# SELECT node_name, nodeis_preferred FROM pgxc_node WHERE node_type = 'D' ORDER BY 1;
 node_name | nodeis_preferred
-----------+------------------
 datanode1 | f
 datanode2 | f
(2 rows)

-- Set datanode1 as the preferred DN.
gaussdb=# ALTER NODE datanode1 WITH(preferred = true);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Query the cluster DN status after the change.
gaussdb=# SELECT node_name, nodeis_preferred FROM pgxc_node WHERE node_type = 'D' ORDER BY 1;
 node_name | nodeis_preferred
-----------+------------------
 datanode1 | t
 datanode2 | f
(2 rows)

-- Connect to only one CN to drop cluster nodes.
gaussdb=# DROP NODE datanode1;
```

Helpful Links

CREATE NODE and **DROP NODE**

1.13.7.21 ALTER NODE GROUP

Description

Changes the information about a cluster node group.

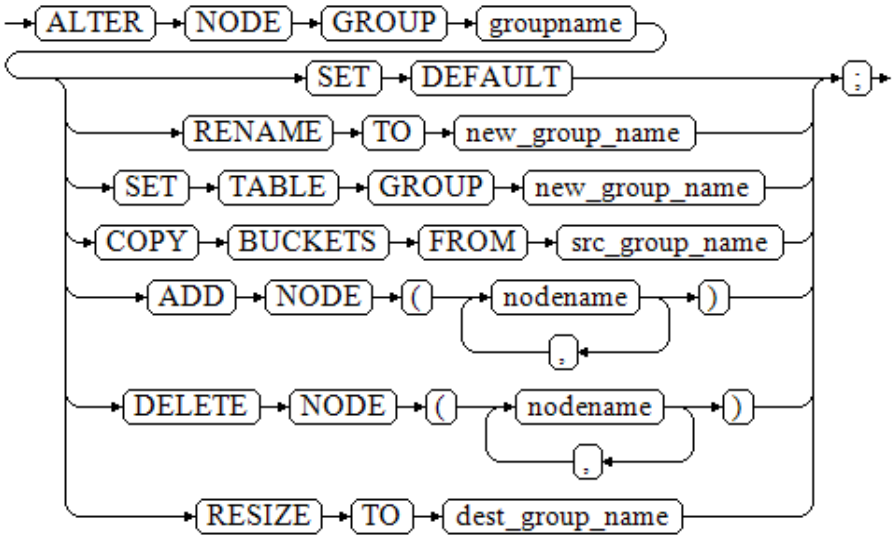
Precautions

- Only the system administrator or a user who has been granted the ALTER permission for the node group can modify the node group information.
- Node group modification is an internal operation of the system. Except SET DEFAULT, other operations must be performed in maintenance mode (by calling **SET xc_maintenance_mode=on;**).
- ALTER NODE GROUP can be used only within a database. To avoid data inconsistency in the database system, do not manually run this SQL statement.

Syntax

```
ALTER NODE GROUP groupname
{ SET DEFAULT
   | RENAME TO new_group_name
   | SET TABLE GROUP new_group_name
   | COPY BUCKETS FROM src_group_name
   | ADD NODE ( nodename [, ... ] )
   | DELETE NODE ( nodename [, ... ] )
   | RESIZE TO dest_group_name
};
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- **groupname**

  Specifies the name of the cluster node group to be altered.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **SET DEFAULT**

  Sets the **in_redistribution** column to **'y'** for all cluster node group objects except the one specified by **groupname**. To be compatible with earlier versions, this syntax is retained and does not need to be executed in maintenance mode.

- **RENAME TO new_group_name**

  Renames a specified cluster node group.

- **SET TABLE GROUP new_group_name**

  Changes all the **group_names** in the **pgroup** columns of the **pgxc_class** tables on all CNs to **new_group_name**.

- **COPY BUCKETS FROM src_group_name**

  Copies the content of the **group_members** and **group_buckets** columns from the cluster node group specified by **src_group_name** to the cluster node group specified by **groupname**.

- **ADD NODE (nodename [, ...])**

  Adds specified nodes to the cluster node group specified by **groupname**. These new nodes exist in the **PGXC_NODE** system catalog. This statement only modifies the system catalog and does not add nodes or redistribute data. Do not call this statement. You can observe the impact of the statement in the **PGXC_GROUP** system catalog.

- **DELETE NODE (nodename [, ...])**

  Deletes specified nodes from the cluster node group specified by **groupname**. The deleted nodes still exist in the **PGXC_NODE** system catalog. This statement only modifies the system catalog and does not delete nodes or redistribute data. Do not call this statement. You can observe the impact of the statement in the **PGXC_GROUP** system catalog.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **RESIZE TO dest_group_name**

  Resizes the cluster node group for redistribution from the source group (**groupname**) to the destination group (**dest_group_name**). The **is_installation** flag is removed from the source group and added to the destination group.

Examples

```
-- Query the initial cluster DN status.
gaussdb=# SELECT node_name, nodeis_preferred FROM pgxc_node WHERE node_type = 'D' ORDER BY 1;
 node_name | nodeis_preferred
-----------+------------------
 dn_6001_6002_6003 | f
 dn_6004_6005_6006 | f
 dn_6007_6008_6009 | f
(3 rows)

-- Create a node group and replace dn_6001_6002_6003 with the actual node name obtained in the previous step.
gaussdb=# CREATE NODE GROUP test_group WITH ( dn_6001_6002_6003 );

-- Changes the name of the created node group to test_group_new. This must be implemented in the maintenance mode (by calling SET xc_maintenance_mode=on;).
gaussdb=# SET xc_maintenance_mode=on;
gaussdb=# ALTER NODE GROUP test_group RENAME TO test_group_new;

-- Query the node group.
gaussdb=# SELECT group_name, group_members FROM pgxc_group;
   group_name   |   group_members
----------------+-------------------
 group_version1 | 16384 16388 16394
 test_group_new | 16384
(2 rows)

-- Delete the node group and disable the maintenance mode.
gaussdb=# DROP NODE GROUP test_group_new;
gaussdb=# SET xc_maintenance_mode=off;
```

Helpful Links

CREATE NODE GROUP and **DROP NODE GROUP**

1.13.7.22 ALTER PACKAGE

Description

Alters attributes of a package or recompiles a package.

Precautions

- Currently, only ALTER PACKAGE OWNER is supported. System administrators have this permission by default. To execute this command, you must be the owner of the package or a system administrator and a member of the new owner role.
- When separation of duties is enabled, to alter the owner of a package, you must have the user group permission, even as a system administrator. The owner of a package of the DEFINER type cannot be altered.
- Only the initial user can alter the owner of a package to the initial user.

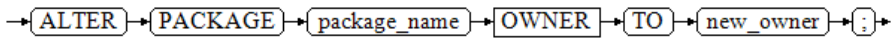
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- When separation of duties is disabled, only a system administrator or a user with higher permissions can alter the owner of a package.
- A system administrator is not allowed to alter the owner of a package of the DEFINER type to the initial user or OPRADMIN.

Syntax

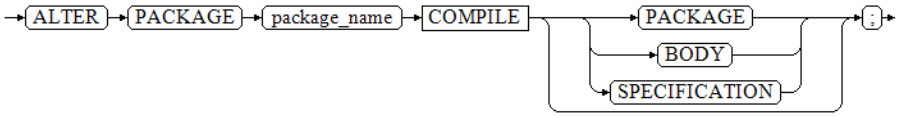
- Alter the owner of a package.

```
ALTER PACKAGE package_name OWNER TO new_owner;
```



- Recompile the package.

```
ALTER PACKAGE package_name COMPILE [PACKAGE | BODY | SPECIFICATION];
```



Parameters

- **package_name**

  Specifies the name of the package to be altered.

  Value range: an existing package name. Only one package can be altered at a time.

- **new_owner**

  Specifies the new owner of a package. To alter the owner of a package, the new owner must have the CREATE permission on the schema to which the package belongs.

  Value range: an existing user role.

Examples

- Change the owner of a package.

```
-- Create a database.
gaussdb=# CREATE DATABASE test DBCOMPATIBILITY 'ORA';
gaussdb=# \c test

-- Create a package.
test=# CREATE OR REPLACE PACKAGE test_pkg AS
    pkg_var int := 1;
    PROCEDURE test_pkg_proc(var int);
END test_pkg;
/

test=# CREATE OR REPLACE PACKAGE BODY test_pkg AS
    PROCEDURE test_pkg_proc(var int) AS
    BEGIN
        pkg_var := 1;
    END;
END test_pkg;
/

-- Create a user.
test=# CREATE ROLE test PASSWORD '********';

-- Change the owner of a package.
test=# ALTER PACKAGE test_pkg OWNER TO test;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Query the owner of test_pkg.
test=# SELECT t1.pkgname,t2.rolname FROM gs_package t1, gs_roles t2 WHERE t1.pkgname =
'test_pkg' AND t1.pkgowner = t2.oid;
 pkgname  | rolname
----------+---------
 test_pkg | test
(1 row)

-- Drop.
test=# DROP PACKAGE test_pkg;
test=# DROP ROLE test;
-- Switch to the default database. Change the database name based on the actual situation.
test=# \c postgres
gaussdb=# DROP DATABASE test;
```

Helpful Links

CREATE PACKAGE and **DROP PACKAGE**

1.13.7.23 ALTER PROCEDURE

Description

Changes the attributes of a user-defined stored procedure or recompiles a stored procedure.

Precautions

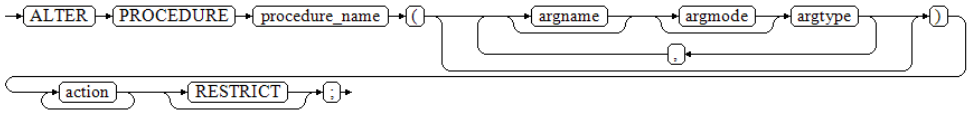
- Only the owner of a stored procedure or a user with the ALTER permission can run the **ALTER PROCEDURE** command. System administrators have this permission by default. The following are permission restrictions depending on attributes to be modified:
  - If a stored procedure involves operations on temporary tables, ALTER PROCEDURE cannot be used.
  - To modify the owner or schema of a stored procedure, you must be the owner of the stored procedure or system administrator and a member of the new owner role.
  - Only system administrators and the initial user can alter the schema of a stored procedure to public.
- Only the initial user or the user who creates the stored procedure can modify the stored procedure to a stored procedure that has the definer permission.
- When separation of duties is enabled, no role is allowed to alter the owner of a stored procedure with the definer permission.
- When separation of duties is disabled, only the initial user and system administrators can alter the owner of a stored procedure with the definer permission.
- Only the initial user can alter the owner of a stored procedure to the initial user.

Syntax

- Alter the additional parameters of a user-defined stored procedure.

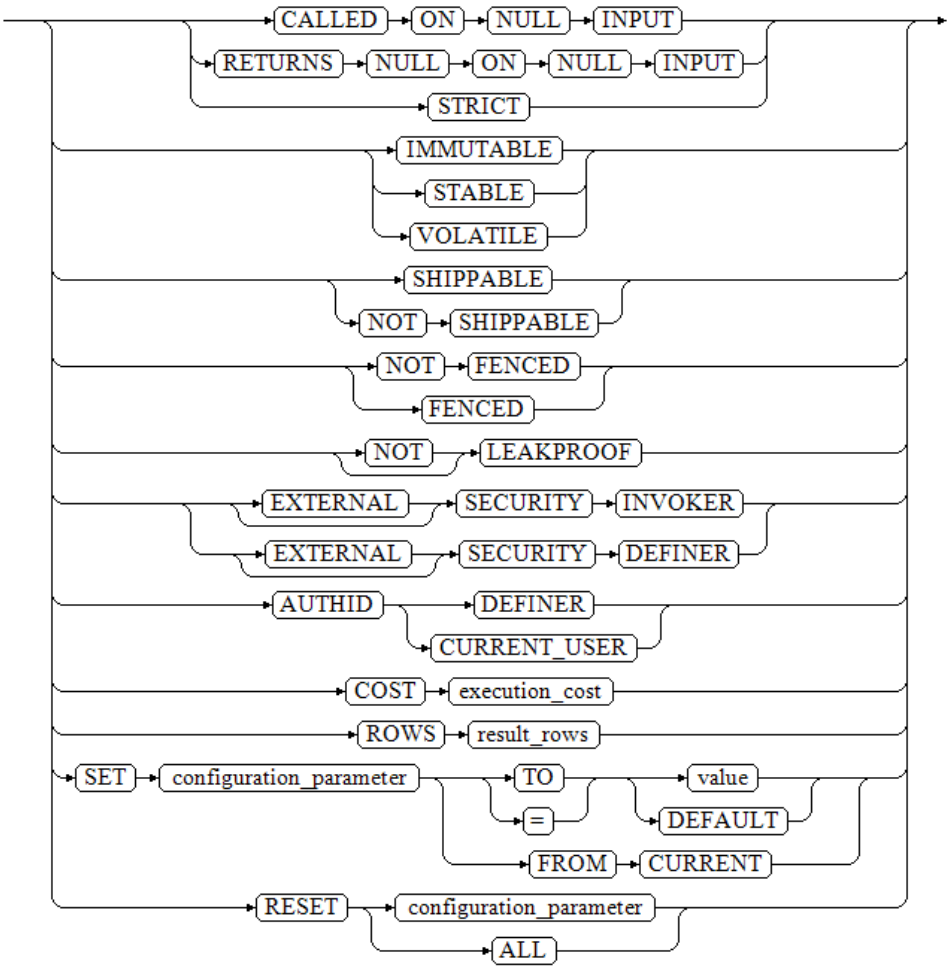
```
ALTER PROCEDURE procedure_name ( [ { [ argname ] [ argmode ] argtype} [, ...] ] )
    action [ ... ] [ RESTRICT ];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



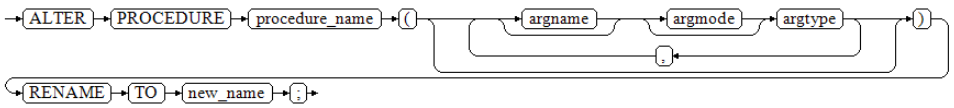
The syntax of the ACTION clause is as follows:

```
{{CALLED ON NULL INPUT  | RETURNS NULL ON NULL INPUT | STRICT}
 | {IMMUTABLE | STABLE | VOLATILE}
 | {SHIPPABLE | NOT SHIPPABLE}
 | {NOT FENCED | FENCED}
 | [ NOT ] LEAKPROOF
 | { [ EXTERNAL ] SECURITY INVOKER | [ EXTERNAL ] SECURITY DEFINER }
 | AUTHID { DEFINER | CURRENT_USER }
 | COST execution_cost
 | ROWS result_rows
 | SET configuration_parameter { { TO | = } { value | DEFAULT }| FROM CURRENT}
 | RESET {configuration_parameter | ALL}}
```



- Rename a user-defined stored procedure.

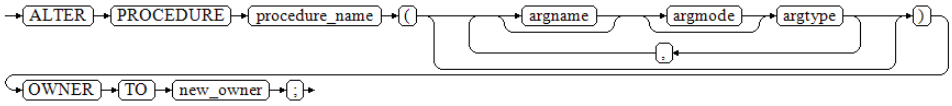
```
ALTER PROCEDURE procedure_name( [ { [ argname ] [ argmode ] argtype} [, ...] ] )
    RENAME TO new_name;
```



- Alter the owner of a user-defined stored procedure.

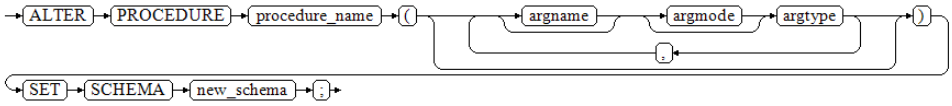
```
ALTER PROCEDURE procedure_name( [ { [ argname ] [ argmode ] argtype} [, ...] ] )
    OWNER TO new_owner;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



- Alter the schema of a user-defined stored procedure.

```
ALTER PROCEDURE procedure_name( [ { [ argname ] [ argmode ] argtype} [, ...] ] )
    SET SCHEMA new_schema;
```



Parameters

- **procedure_name**

  Specifies the name of the stored procedure to be modified.

  Value range: an existing stored procedure name

- **argmode**

  Specifies whether a parameter is an input or output parameter.

  Value range: **IN**, **OUT**, **INOUT**, and **VARIADIC**

- **argname**

  Parameter name.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **argtype**

  Specifies the type of the stored procedure parameter.

- **CALLED ON NULL INPUT**

  Specifies that the stored procedure can be called normally when some parameters of the stored procedure are **NULL**. Omitting this parameter is the same as specifying it.

- **RETURNS NULL ON NULL INPUT | STRICT**

  STRICT specifies that the stored procedure returns **NULL** when a parameter of the stored procedure is **NULL**. If this parameter is specified, the stored procedure is not executed when there are null parameters; instead a null result is returned automatically.

  RETURNS NULL ON NULL INPUT and **STRICT** have the same functions.

- **IMMUTABLE**

  Specifies that the stored procedure always returns the same result if the parameter values are the same.

- **STABLE**

  Specifies that the stored procedure cannot alter the database, and that within a single table scan it will consistently return the same result for the same parameter value, but its result varies by SQL statements.

- **VOLATILE**

  Specifies that the stored procedure value can change in a single table scan and no optimization is performed.

- **SHIPPABLE | NOT SHIPPABLE**, **FENCED | NOT FENCED**, etc.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies a constraint. The function of each parameter is similar to that of **CREATE FUNCTION**. For details, see **CREATE FUNCTION**.

- **LEAKPROOF**

  Specifies that the stored procedure has no side effect and the parameter contains only the return value. LEAKPROOF can be set only by system administrators.

- **EXTERNAL**

  (Optional) The purpose is to be compatible with SQL. This feature applies to all functions, not only external functions.

- **SECURITY INVOKER | AUTHID CURRENT_USER**

  Specifies that the stored procedure will be executed with the permissions of the user who calls it. Omitting this parameter is the same as specifying it.

  SECURITY INVOKER and **AUTHID CURRENT_USER** have the same functions.

- **SECURITY DEFINER | AUTHID DEFINER**

  Specifies that the stored procedure will be executed with the permissions of the user who created it.

  AUTHID DEFINER and **SECURITY DEFINER** have the same functions.

- **COST execution_cost**

  Estimates the execution cost of a stored procedure.

  The unit of **execution_cost** is **cpu_operator_cost**.

  Value range: a positive integer

- **ROWS result_rows**

  Estimates the number of rows returned by the stored procedure. This is only allowed when the stored procedure is declared to return a set.

  Value range: a positive number. The default value is **1000**.

- **configuration_parameter**
  - **value**

    Sets a specified database session parameter to a specific value.

    Value range: a string.

    - **DEFAULT**: The default system setting is used in the new session.
    - **OFF**: The setting is disabled.
    - **RESET**: The default system setting is used in the new session.
    - User-specified value: The value must meet the restriction of the modified parameter.

  - **DEFAULT**

    Default value.

  - **FROM CURRENT**

    Uses the value of **configuration_parameter** of the current session.

- **ALL**

  Specifies all GUC parameters.

- **new_name**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the new name of the stored procedure. To change the schema of a stored procedure, you must have the CREATE permission on the new schema.

Value range: character strings complying with **Identifier Naming Conventions**.

- **new_owner**

  Specifies the new owner of the stored procedure. To change the owner of a stored procedure, the new owner must have the CREATE permission on the schema to which the stored procedure belongs.

  Value range: an existing user role

- **new_schema**

  Specifies the new schema of the stored procedure.

  Value range: an existing schema name.

- **RESTRICT**

  It is used solely for compliance with the SQL standard and can be ignored.

Examples

```
-- Create a stored procedure.
gaussdb=# CREATE OR REPLACE PROCEDURE test_proc(a int)
IS
    proc_var int;
BEGIN
    proc_var := a;
END;
/

-- Change the execution rule of a stored procedure to IMMUTABLE. That is, the returned results are the same when the parameters remain unchanged.
gaussdb=# ALTER PROCEDURE test_proc(a int) IMMUTABLE;

-- Create a user.
gaussdb=# CREATE USER joe PASSWORD '********';

-- Change the owner of a stored procedure.
gaussdb=# ALTER PROCEDURE test_proc(a int)  OWNER TO joe;

-- Create a schema.
gaussdb=# CREATE SCHEMA test;

-- Modify the schema of a stored procedure.
gaussdb=# ALTER PROCEDURE  test_proc(a int)  SET SCHEMA test;

-- Drop a stored procedure. The schema to which the function belongs must be specified.
gaussdb=# DROP PROCEDURE test.test_proc;

-- Drop the new user and schema.
gaussdb=# DROP USER joe;
gaussdb=# DROP SCHEMA test;
```

Helpful Links

CREATE PROCEDURE and **DROP PROCEDURE**

1.13.7.24 ALTER RESOURCE LABEL

Description

Alters a resource label.

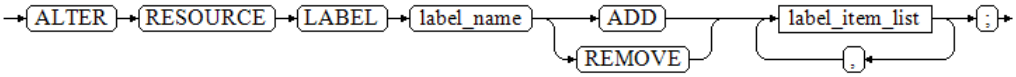
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

Only users with the POLADMIN or SYSADMIN permission or the initial user can run the **ALTER RESOURCE LABEL** command.

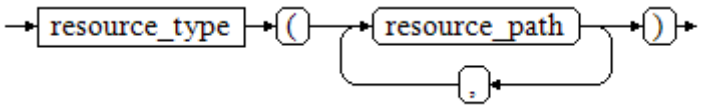
Syntax

```
ALTER RESOURCE LABEL label_name {ADD | REMOVE}
  label_item_list [, ...];
```



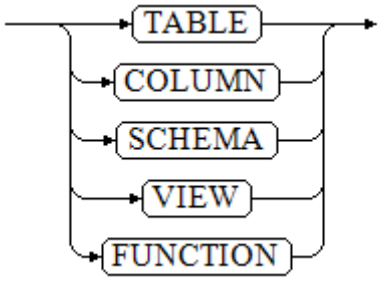
- label_item_list

  ```
  resource_type (resource_path [, ...])
  ```

- resource_type

  ```
  {TABLE | COLUMN | SCHEMA | VIEW | FUNCTION}
  ```

Parameters

- **label_name**

  Specifies a resource label name.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **resource_type**

  Specifies the type of database resources to be labeled.

- **resource_path**

  Specifies the path of a specific database resource.

Examples

```
-- Create basic table table_for_label.
gaussdb=# CREATE TABLE table_for_label(col1 int, col2 text);

-- Create resource label table_label.
gaussdb=# CREATE RESOURCE LABEL table_label ADD COLUMN(table_for_label.col1);

-- Attach resource label table_label to col2.
gaussdb=# ALTER RESOURCE LABEL table_label ADD COLUMN(table_for_label.col2);

-- Remove an item from table_label.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# ALTER RESOURCE LABEL table_label REMOVE COLUMN(table_for_label.col1);

-- Delete the resource label table_label.
gaussdb=# DROP RESOURCE LABEL table_label;

-- Delete the base table table_for_label.
gaussdb=# DROP TABLE table_for_label;
```

Helpful Links

CREATE RESOURCE LABEL and **DROP RESOURCE LABEL**

1.13.7.25 ALTER RESOURCE POOL

Description

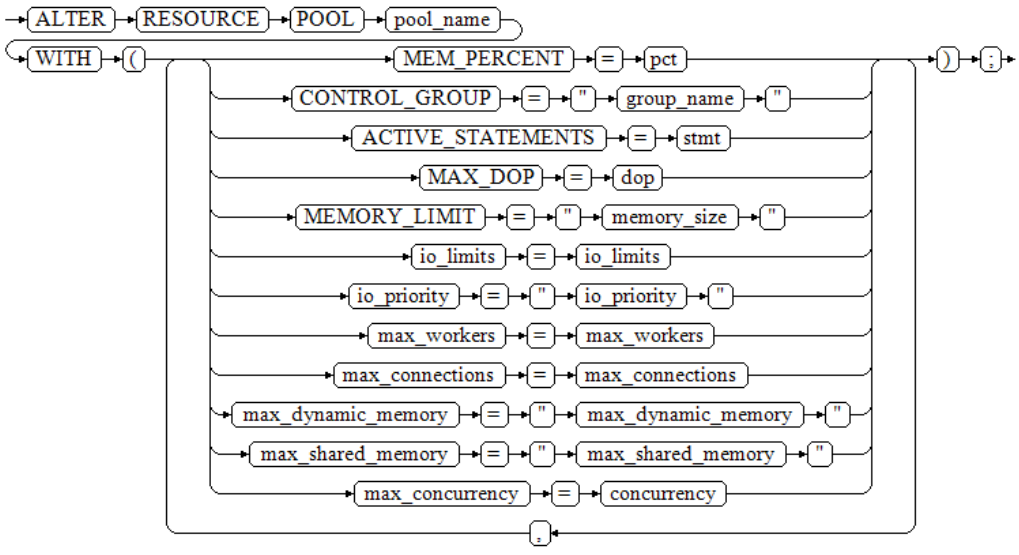
Changes the Cgroup of a resource pool.

Precautions

- Only users with the SYSADMIN permission or the initial user can run the **ALTER RESOURCE POOL** command.
- This syntax is not supported during upgrade.

Syntax

```
ALTER RESOURCE POOL pool_name
    WITH ({MEM_PERCENT = pct | CONTROL_GROUP = "group_name" | ACTIVE_STATEMENTS = stmt |
MAX_DOP = dop | MEMORY_LIMIT = "memory_size" | io_limits = io_limits | io_priority = "io_priority" |
max_worker = max_worker | max_connections = max_connections | max_dynamic_memory =
"max_dynamic_memory" | max_shared_memory = "max_shared_memory" | max_concurrency =
concurrency} [, ... ]);
```



Parameters

- **pool_name**

  Specifies the name of a resource pool. The resource pool must already exist.

  Value range: character strings complying with **Identifier Naming Conventions**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **group_name**

  Specifies the name of a Cgroup.

  Value range: an existing Cgroup.

  NOTE

  - You can use either double quotation marks ("") or single quotation marks (') in the syntax when setting the name of a Cgroup.
  - The value of **group_name** is case-sensitive.
  - If **group_name** is not specified, the default value **Medium** is used, indicating the **Medium Timeshare** Cgroup under the **DefaultClass** Cgroup.
  - If an administrator specifies a Workload Cgroup under Class, for example, **control_group** set to **class1:workload1**, the resource pool will be associated with the **workload1** Cgroup under **class1**. The level of **Workload** can also be specified. For example, **control_group** is set to **class1:workload1:1**.
  - If a database user specifies the Timeshare string (**Rush**, **High**, **Medium**, or **Low**) in the syntax, for example, **control_group** is set to **High**, the resource pool will be associated with the **High** Timeshare Cgroup under **DefaultClass**.
  - In multi-tenant scenarios, the Cgroup associated with a group resource pool is a Class Cgroup, and that associated with a service resource pool is a Workload Cgroup. Additionally, switching Cgroups between different resource pools is not allowed.

- **stmt**

  Specifies the maximum number of statements that can be concurrently executed in a resource pool.

  Value range: a number ranging from –1 to 2147483647. The value **–1** indicates no limit.

- **dop**

  Specifies the maximum statement concurrency degree for a resource pool, equivalent to the number of threads that can be created for executing a statement.

  Value range: numeric data ranging from 1 to 2147483647.

- **memory_size**

  Specifies the maximum memory size of a resource pool.

  Value range: a string of 1 KB to 2047 GB case-sensitive characters.

- **mem_percent**

  Specifies the proportion of available resource pool memory to the total memory or group user memory.

  – In multi-tenant scenarios, **mem_percent** of group users or service users ranges from 1 to 100. The default value is **20**.

  – In common scenarios, **mem_percent** of common users is an integer ranging from 0 to 100. The default value is **0**.

  NOTE

  When both of **mem_percent** and **memory_limit** are specified, only **mem_percent** takes effect.

- **io_limits**

  Specifies the maximum number of I/Os that can be triggered per second in a resource pool. The unit is 10,000.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: a number ranging from 0 to 2147483647. The value **0** indicates no limit.

- **io_priority**

  Specifies the I/O priority set for jobs that consume many I/O resources. It takes effect when the I/O usage reaches 90%.

  Value range: enumerated type. The options are **None**, **Low**, **Medium**, and **High**. If you do not want to control I/O resources, use the default value **None**.

  NOTE

  The settings of **io_limits** and **io_priority** are valid only for complex jobs, such as batch import (using INSERT INTO SELECT, COPY FROM, or CREATE TABLE AS), complex queries involving over 500 MB data on each DN, and VACUUM FULL.

- **max_worker**

  Concurrency in a table during data redistribution. It is used only for scaling.

- **max_connections**

  Maximum number of connections allowed by a resource pool.

  NOTE

  The total number of connections in all resource pools cannot exceed the value of the GUC parameter **max_connections** set for the entire GaussDB process.

- **max_dynamic_memory**

  Maximum dynamic memory allowed by a resource pool.

- **max_shared_memory**

  Maximum shared memory allowed by a resource pool.

- **max_concurrency**

  Maximum number of concurrent tasks allowed by a resource pool.

Examples

The following example assumes that the user has created the **class1** Cgroup and three Workload Cgroups under **class1**: **Low**, **wg1**, and **wg2**. Contact an administrator to create a Cgroup.

```
-- Create a resource pool.
gaussdb=# CREATE RESOURCE POOL pool1;

-- Specify the High Timeshare Workload Cgroup under the DefaultClass Cgroup.
gaussdb=# ALTER RESOURCE POOL pool1 WITH (CONTROL_GROUP="High");

-- Specify the Low Timeshare Workload Cgroup under the class1 Cgroup.
gaussdb=# ALTER RESOURCE POOL pool1 WITH (CONTROL_GROUP="class1:Low");

-- Specify the wg1 Workload Cgroup under the class1 Cgroup.
gaussdb=# ALTER RESOURCE POOL pool1 WITH (CONTROL_GROUP="class1:wg1");

-- Specify the wg2 Workload Cgroup under the class1 Cgroup.
gaussdb=# ALTER RESOURCE POOL pool1 WITH (CONTROL_GROUP="class1:wg2:3");

-- Drop the resource pool pool1.
gaussdb=# DROP RESOURCE POOL pool1;
```

Helpful Links

CREATE RESOURCE POOL and **DROP RESOURCE POOL**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.7.26 ALTER ROLE

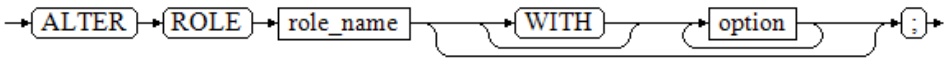
Description

Alters role attributes, including renaming a role, locking or unlocking a role, and altering permissions of a role.

Syntax

- Alter permissions of a role.

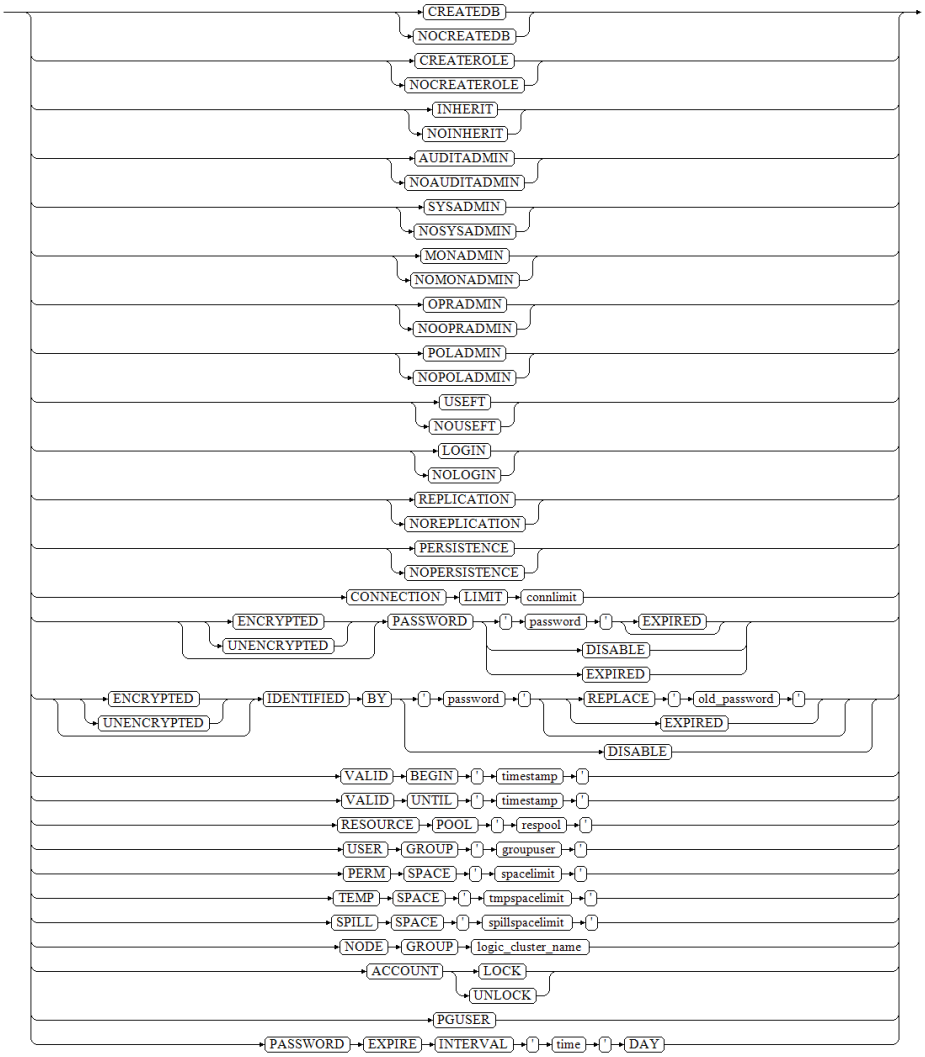
```
ALTER ROLE role_name [ [ WITH ] option [ ... ] ];
```



The option clause is as follows:

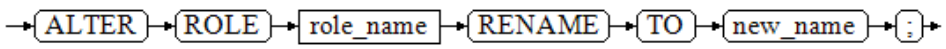
```
{
   {CREATEDB | NOCREATEDB}
   | {CREATEROLE | NOCREATEROLE}
   | {INHERIT | NOINHERIT}
   | {AUDITADMIN | NOAUDITADMIN}
   | {SYSADMIN | NOSYSADMIN}
   | {MONADMIN | NOMONADMIN}
   | {OPRADMIN | NOOPRADMIN}
   | {POLADMIN | NOPOLADMIN}
   | {USEFT | NOUSEFT}
   | {LOGIN | NOLOGIN}
   | {REPLICATION | NOREPLICATION}
   | {PERSISTENCE | NOPERSISTENCE}
   | CONNECTION LIMIT connlimit
   | [ ENCRYPTED | UNENCRYPTED ] PASSWORD { 'password' [EXPIRED] | DISABLE | EXPIRED }
   | [ ENCRYPTED | UNENCRYPTED ] IDENTIFIED BY { 'password' [ REPLACE 'old_password' |
EXPIRED ] | DISABLE }
   | VALID BEGIN 'timestamp'
   | VALID UNTIL 'timestamp'
   | RESOURCE POOL 'respool'
   | USER GROUP 'groupuser'
   | PERM SPACE 'spacelimit'
   | TEMP SPACE 'tmpspacelimit'
   | SPILL SPACE 'spillspacelimit'
   | NODE GROUP logic_cluster_name
   | ACCOUNT { LOCK | UNLOCK }
   | PGUSER
   | PASSWORD EXPIRE INTERVAL 'time' DAY
}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



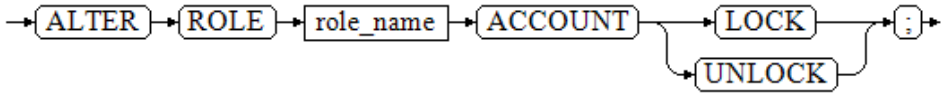
- Rename a role.

```
ALTER ROLE role_name
    RENAME TO new_name;
```



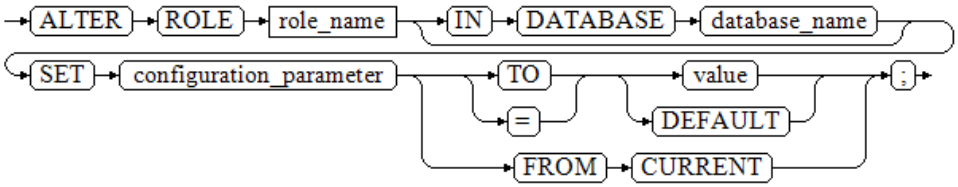
- Lock or unlock a role.

```
ALTER ROLE role_name
    ACCOUNT { LOCK | UNLOCK };
```



- Set parameters for a role.

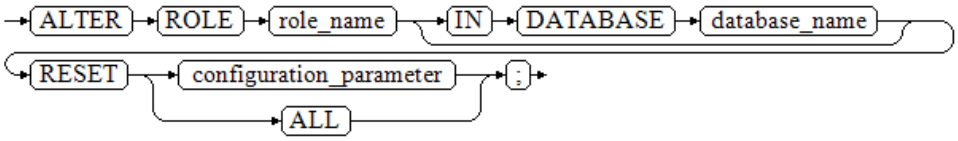
```
ALTER ROLE role_name [ IN DATABASE database_name ]
    SET configuration_parameter {{ TO | = } { value | DEFAULT } | FROM CURRENT};
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Reset parameters for a role.

```
ALTER ROLE role_name
    [ IN DATABASE database_name ] RESET {configuration_parameter|ALL};
```



Parameters

- **role_name**

  Specifies an existing role name.

  Value range: an existing role name. If a role name contains uppercase letters, enclose the name with double quotation marks ("").

  NOTICE

  Currently, the RENAME TO operation cannot be performed on **redisuser**, an object used for resource control during redistribution, and you also cannot rename other roles to **redisuser**.

- **IN DATABASE database_name**

  Modifies parameters of a role in a specified database.

- **SET configuration_parameter {{ TO | = } { value | DEFAULT } | FROM CURRENT}**

  Sets parameters for a role.

  NOTICE

  Session parameters modified using ALTER ROLE apply only to the specified role. The current version does not support user-level parameter settings.

  Value range:

  - **configuration_parameter**: For details about its value, see **SET**.
  - **value**: For details about its value, see **SET**.
  - **DEFAULT**: clears the value of **configuration_parameter**. **configuration_parameter** will inherit the default value of the new session generated for the role.
  - **FROM CURRENT**: uses the value of **configuration_parameter** in the current session.

- **RESET {configuration_parameter|ALL}**

  Clears the value of **configuration_parameter**. The statement has the same effect as that of **SET configuration_parameter TO DEFAULT**.

  NOTICE

  The current version does not support the resetting of user-level parameters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: **ALL** indicates that the values of all parameters are cleared.

- **ACCOUNT LOCK | ACCOUNT UNLOCK**
  - **ACCOUNT LOCK**: locks a role account to prevent the role from logging in to the database through password authentication.
  - **ACCOUNT UNLOCK**: unlocks a role account to allow the role to log in to the database.
- **PGUSER**

  In the current version, the **PGUSER** attribute of a role cannot be modified.
- **{PASSWORD|IDENTIFIED BY} 'password'**

  Resets or changes the user password.
  - Except for the initial user, other administrators and common users must enter the correct old passwords when changing their own passwords.
  - Only the initial user, SYSADMIN when separation of duties is disabled, or users who have the CREATEROLE permission can reset the password of a common user without entering the old password.
  - The initial user can reset passwords of system administrators. System administrators are prohibited from resetting passwords for each other.
  - User passwords must be enclosed within single quotation marks (') or double quotation marks (").
- **EXPIRED**

  Sets the password as expired.
  - Only the initial user, system administrators (SYSADMIN), and users with the CREATEROLE permission can set user passwords as expired. System administrator may perform this action for themselves or for other system administrators only when separation of duties is disabled.
  - The password of the initial user cannot be set as expired.
  - A user whose password has expired can log in to the database but cannot perform query operations. The user can perform query operations only after changing the password or asking an administrator to reset the password.
- For details about other parameters, see **CREATE ROLE** in section **Parameters**.

Examples

```
-- Create role test_role.
gaussdb=# CREATE ROLE test_role PASSWORD '********';

-- Set role test_role to log in to the database.
gaussdb=# ALTER ROLE test_role WITH LOGIN;

-- Create a database.
gaussdb=# CREATE DATABASE testdb1;

-- Lock role test_role.
gaussdb=# ALTER ROLE test_role ACCOUNT LOCK;

-- Unlock a locked role.
gaussdb=# ALTER ROLE test_role ACCOUNT UNLOCK;

-- Change the password of role test_role.
gaussdb=# ALTER ROLE test_role PASSWORD '********';

-- Rename role test_role to test_role2.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# ALTER ROLE test_role RENAME TO test_role2;

-- Change role test_role2 to SYSADMIN.
gaussdb=# ALTER ROLE test_role2 SYSADMIN;

-- Drop.
gaussdb=# DROP ROLE test_role2;
gaussdb=# DROP DATABASE testdb1;
```

Helpful Links

CREATE ROLE, **DROP ROLE**, and **SET ROLE**

1.13.7.27 ALTER ROW LEVEL SECURITY POLICY

Description

Alters an existing row-level security policy, including the policy name, users affected by the policy, and policy expression.

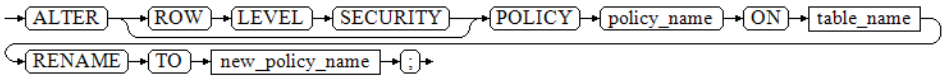
Precautions

Only the table owner or administrator can run the **ALTER ROW LEVEL SECURITY POLICY** command.

Syntax

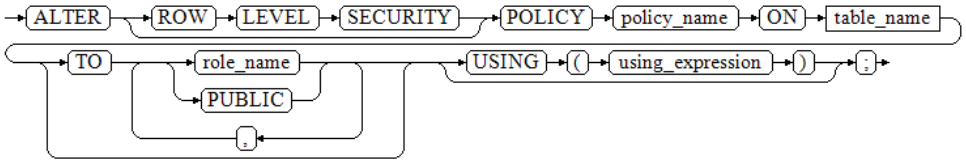
- Rename an existing row-level security policy.

```
ALTER [ ROW LEVEL SECURITY ] POLICY policy_name ON table_name RENAME TO new_policy_name;
```



- Alter the specified user and expression of an existing row-level security policy.

```
ALTER [ ROW LEVEL SECURITY ] POLICY policy_name ON table_name
    [ TO { role_name | PUBLIC } [, ...] ]
    [ USING ( using_expression ) ];
```



Parameters

- **policy_name**

  Specifies the name of a row-level security policy.

- **table_name**

  Specifies the name of a table to which a row-level security policy is applied.

- **new_policy_name**

  Specifies the new name of a row-level security policy.

- **role_name**

  Specifies names of users affected by a row-level security policy. **PUBLIC** indicates that the row-level security policy will affect all users.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **using_expression**

  Specifies a row-level security policy, which is similar to a Boolean expression in the WHERE clause.

Examples

```
-- Create the data table all_data.
gaussdb=# CREATE TABLE all_data(id int, role varchar(100), data varchar(100));

-- Create a row-level security policy to specify that the current user can view only their own data.
gaussdb=# CREATE ROW LEVEL SECURITY POLICY all_data_rls ON all_data USING(role = CURRENT_USER);
gaussdb=# \d+ all_data
                        Table "public.all_data"
 Column |          Type          | Modifiers | Storage  | Stats target | Description
--------+------------------------+-----------+----------+--------------+-------------
 id     | integer                |           | plain    |              |
 role   | character varying(100) |           | extended |              |
 data   | character varying(100) |           | extended |              |
Row Level Security Policies:
    POLICY "all_data_rls" FOR ALL
      TO public
      USING (((role)::name = "current_user"()))
Has OIDs: no
Distribute By: HASH(id)
Location Nodes: ALL DATANODES
Options: orientation=row, compression=no, storage_type=USTORE, segment=off

-- Change the name of the all_data_rls policy.
gaussdb=# ALTER ROW LEVEL SECURITY POLICY all_data_rls ON all_data RENAME TO all_data_new_rls;

-- Create users alice and bob.
gaussdb=# CREATE ROLE alice WITH PASSWORD '*********';
gaussdb=# CREATE ROLE bob WITH PASSWORD '*********';

-- Change the users affected by the row-level security policy.
gaussdb=# ALTER ROW LEVEL SECURITY POLICY all_data_new_rls ON all_data TO alice, bob;
gaussdb=# \d+ all_data
                        Table "public.all_data"
 Column |          Type          | Modifiers | Storage  | Stats target | Description
--------+------------------------+-----------+----------+--------------+-------------
 id     | integer                |           | plain    |              |
 role   | character varying(100) |           | extended |              |
 data   | character varying(100) |           | extended |              |
Row Level Security Policies:
    POLICY "all_data_new_rls" FOR ALL
      TO alice,bob
      USING (((role)::name = "current_user"()))
Has OIDs: no
Distribute By: HASH(id)
Location Nodes: ALL DATANODES
Options: orientation=row, compression=no, storage_type=USTORE, segment=off


-- Modify the expression of the row-level security to view only the data of the user whose column ID is greater than 100.
gaussdb=# ALTER ROW LEVEL SECURITY POLICY all_data_new_rls ON all_data USING (id > 100 AND role = current_user);
gaussdb=# \d+ all_data
                        Table "public.all_data"
 Column |          Type          | Modifiers | Storage  | Stats target | Description
--------+------------------------+-----------+----------+--------------+-------------
 id     | integer                |           | plain    |              |
 role   | character varying(100) |           | extended |              |
 data   | character varying(100) |           | extended |              |
Row Level Security Policies:
    POLICY "all_data_new_rls" FOR ALL
      TO alice,bob
      USING (((id > 100) AND ((role)::name = "current_user"())))
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
Has OIDs: no
Distribute By: HASH(id)
Location Nodes: ALL DATANODES
Options: orientation=row, compression=no, storage_type=USTORE, segment=off

-- Delete the policy.
gaussdb=# DROP ROW LEVEL SECURITY POLICY all_data_new_rls ON all_data;

-- Delete users alice and bob.
gaussdb=# DROP ROLE alice, bob;

-- Delete the all_data table.
gaussdb=# DROP TABLE all_data;
```

Helpful Links

CREATE ROW LEVEL SECURITY POLICY and **DROP ROW LEVEL SECURITY POLICY**

1.13.7.28 ALTER SCHEMA

Description

Alters schema attributes, including the name, owner, and tamper-proof attribute of a schema.

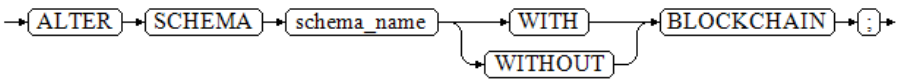
Precautions

- Only the owner of a schema and users with the ALTER permission on the schema are authorized to run the **ALTER SCHEMA** command. When separation of duties is disabled, system administrators have this authorization by default.
- To change the owner of a schema, you must be the owner of the schema or a system administrator and a member of the new owner role.
- Only the initial user can change the owner of a system schema (such as pg_catalog and sys) except PUBLIC. Changing the names of built-in system schemas may make some functions unavailable or even affect the normal running of the database. By default, the names of built-in system schemas cannot be changed. To ensure compatibility with earlier versions, you can change the names of the built-in system schemas only when the system is being started or upgraded, or when **allow_system_table_mods** is set to **on**.
- Only the initial user can change the owner of a schema to an O&M administrator.

Syntax

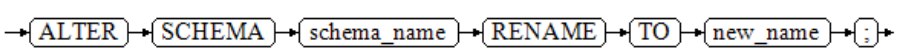
- Alter the tamper-proof attribute of a schema.

```
ALTER SCHEMA schema_name { WITH | WITHOUT } BLOCKCHAIN;
```



- Rename a schema.

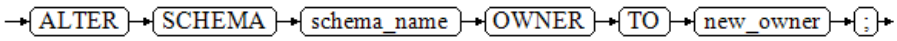
```
ALTER SCHEMA schema_name
  RENAME TO new_name;
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

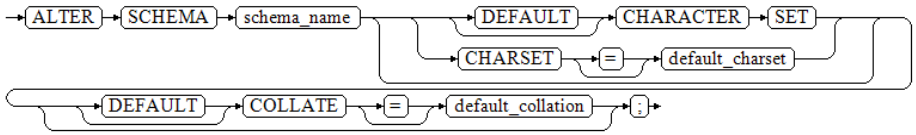
- Alter the owner of a schema.

```
ALTER SCHEMA schema_name
  OWNER TO new_owner;
```



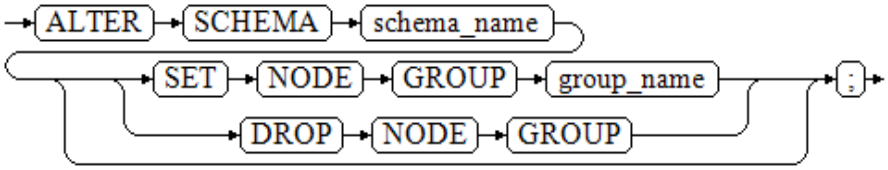
- Alter the default character set and collation of a schema.

```
ALTER SCHEMA schema_name
  [ [DEFAULT] CHARACTER SET | CHARSET [ = ] default_charset ] [ [DEFAULT] COLLATE [ = ]
default_collation ];
```



- Alter the single-node attribute of the schema.

```
ALTER SCHEMA schema_name
  [ SET NODE GROUP group_name | DROP NODE GROUP ];
```



Parameters

- **schema_name**

  Specifies the name of an existing schema.

  Value range: an existing schema name.

- **RENAME TO new_name**

  Rename a schema. **new_name** indicates the new name of a schema.

  Value range: character strings complying with **Identifier Naming Conventions**.

  The rules for naming a schema are as follows:

  – The schema name must be unique in the current database.

  – The new name cannot be the same as the initial username of the current database.

  – Do not use a new name starting with pg_, gs_role_, or data_redis.

- **OWNER TO new_owner**

  Change the owner of a schema. **new_owner** indicates the new owner of a schema.

  Value range: an existing username or role name.

  NOTE

  To do this as a non-SYSADMIN user, you must be a direct or indirect member of the new owner role, and that role must have the CREATE permission on the database.

- **{ WITH | WITHOUT } BLOCKCHAIN**

  Alter the tamper-proof attribute of a schema. **WITH BLOCKCHAIN** indicates the tamper-proof schema.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- To change a normal schema to a tamper-proof schema, set the GUC parameter **enable_ledger** to **on**. The default value is **off**, and the level is SIGHUP.
- Ordinary row-store tables with the tamper-proof attribute are tamper-proof history tables, excluding foreign tables, temporary tables, and system catalogs.
- You can modify or switch the tamper-proof attribute of a schema only when the schema does not contain any table.
- The tamper-proof attribute cannot be modified for the temporary table schema, TOAST table schema, dbe_perf schema, or blockchain schema.

- **default_charset**

  Changes the default character set of a schema.

  Value range: For details about the supported character sets, see **Table 1-398**.

  NOTICE

  - This syntax is supported only when **sql_compatibility** is set to **'MYSQL'** or **'M'**. The preceding information is displayed when **sql_compatibility** is set to **'MYSQL'**. For details about the information when **sql_compatibility** is set to **'M'**, see "SQL Reference > SQL Syntax > SQL Statements > A > ALTER SCHEMA" in *M Compatibility Developer Guide*.
  - If the default collation is not specified, the default collation of the specified character set is used as the default collation of the schema.

- **default_collate**

  Changes the default collation of the schema.

  Value range: For details about the supported collations, see **Table 1-398**.

  NOTICE

  - This syntax is supported only when **sql_compatibility** is set to **'MYSQL'** or **'M'**. The preceding information is displayed when **sql_compatibility** is set to **'MYSQL'**. For details about the information when **sql_compatibility** is set to **'M'**, see "SQL Reference > SQL Syntax > SQL Statements > A > ALTER SCHEMA" in *M Compatibility Developer Guide*.
  - If the default character set is not specified, the character set corresponding to the specified collation is used as the default character set of the schema.

- **SET NODE GROUP group_name | DROP NODE GROUP**
  - **SET NODE GROUP group_name**: sets the single-node attribute of the schema. **group_name** indicates the name of a single-node group. Only the single-node group whose type is 's' can be set.
  - **DROP NODE GROUP**: drops the single-node attribute of the schema.

  NOTE

  This syntax is supported only in M-compatible databases (**sql_compatibility = 'M'**).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

- Alter the tamper-proof attribute of a schema.

```
-- Create a schema test_schema1.
gaussdb=# CREATE SCHEMA test_schema1;

-- Change the schema test_schema1 to the tamper-proof mode.
gaussdb=# ALTER SCHEMA test_schema1 WITH BLOCKCHAIN;

-- Query mode information. The tamper-proof attribute is true.
gaussdb=# \dn+ test_schema1
                          List of schemas
     Name     | Owner | Access privileges | Description | WithBlockChain
--------------+-------+-------------------+-------------+----------------
 test_schema1 | omm   |                   |             | t
(1 row)
```

- Rename a schema.

```
-- Rename the schema test_schema1 to test_sch1.
gaussdb=# ALTER SCHEMA test_schema1 RENAME TO test_sch1;

-- Query schema information.
gaussdb=# \dn+ test*
                        List of schemas
   Name    | Owner | Access privileges | Description | WithBlockChain
-----------+-------+-------------------+-------------+----------------
 test_sch1 | omm   |                   |             | t
(1 row)
```

- Change the owner of a schema.

```
-- Create user test_user.
gaussdb=# CREATE ROLE test_user PASSWORD '********';

-- Change the owner of schema test_sch1 to test_user.
gaussdb=# ALTER SCHEMA test_sch1 OWNER TO test_user;

-- Query schema information.
gaussdb=# \dn+  test_sch1;
                          List of schemas
   Name    |   Owner   | Access privileges | Description | WithBlockChain
-----------+-----------+-------------------+-------------+----------------
 test_sch1 | test_user |                   |             | t
(1 row)

-- Drop the new schema and user.
gaussdb=# DROP SCHEMA test_sch1;
gaussdb=# DROP ROLE test_user;
```

- Alter the default character set and collation.

NOTICE

Except the binary character set and collation, only the character set that is the same as the database encoding can be specified.

```
-- Create and switch to the test database.
gaussdb=# CREATE DATABASE test1 WITH DBCOMPATIBILITY = 'MYSQL' ENCODING = 'UTF8'
LC_COLLATE = 'zh_CN.utf8' LC_CTYPE = 'zh_CN.utf8';
gaussdb=# \c test1

-- Create a schema test_sch2.
test1=# CREATE SCHEMA test_sch2;
-- Change the default character set to utf8mb4 and the default collation of characters to
utf8mb4_bin.
test1=# ALTER SCHEMA test_sch2 CHARACTER SET utf8mb4 COLLATE utf8mb4_bin;

-- Drop the new schema.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
test1=# DROP SCHEMA test_sch2;

-- Switch to the default database (change the database name based on the actual situation) and drop the new database.
test1=# \c postgres
gaussdb=# DROP DATABASE test1;
```

- Modify the single-node attribute of the schema.

```
-- Create a schema test_schema1.
gaussdb=# CREATE SCHEMA test_schema1;

-- Modify the single-node attribute of the test_schema1 schema.
gaussdb=# ALTER SCHEMA test_schema1 SET NODE GROUP sys_single_group1;

-- Query the single-node attribute of the test_schema1 schema. The nspgroupoid column in the pg_namespace system catalog records the OID of the node group corresponding to the single-node attribute.
gaussdb=# SELECT * FROM pg_namespace WHERE nspname='test_schema1';
   nspname    | nspowner | nsptimeline | nspacl | in_redistribution | nspblockchain | nspcollation | nspgroupoid
--------------+----------+-------------+--------+-------------------+---------------+--------------+-------------
 test_schema1 |       10 |           0 |        | n                 | f             |              |       16389
(1 row)

-- Drop the single node attribute of the test_schema1 schema.
gaussdb=# ALTER SCHEMA test_schema1 DROP NODE GROUP;

-- Query the single node attribute of the test_schema1 schema.
gaussdb=# SELECT * FROM pg_namespace WHERE nspname='test_schema1';
   nspname    | nspowner | nsptimeline | nspacl | in_redistribution | nspblockchain | nspcollation | nspgroupoid
--------------+----------+-------------+--------+-------------------+---------------+--------------+-------------
 test_schema1 |       10 |           0 |        | n                 | f             |              |
(1 row)
```

Helpful Links

CREATE SCHEMA and **DROP SCHEMA**

1.13.7.29 ALTER SEQUENCE

Description

Alters parameters of a specified sequence.

Precautions

- To run the **ALTER SEQUENCE** command, you must be the sequence owner, a user with the ALTER permission on the sequence, or a user with the ALTER ANY SEQUENCE permission. When separation of duties is disabled, system administrators have this authorization by default. To modify the owner of a sequence, you must be the sequence owner (or a system administrator) and also a member of the new owner's role.
- In the current version, you can modify only the owner, owning column, and maximum value. To alter other parameters, delete the sequence and rebuild it. Then, use the setval function to restore parameter values.
- ALTER SEQUENCE MAXVALUE cannot be used in transactions, functions, or stored procedures.
- After the maximum value of a sequence is altered, the cache of the sequence in all sessions is cleared.

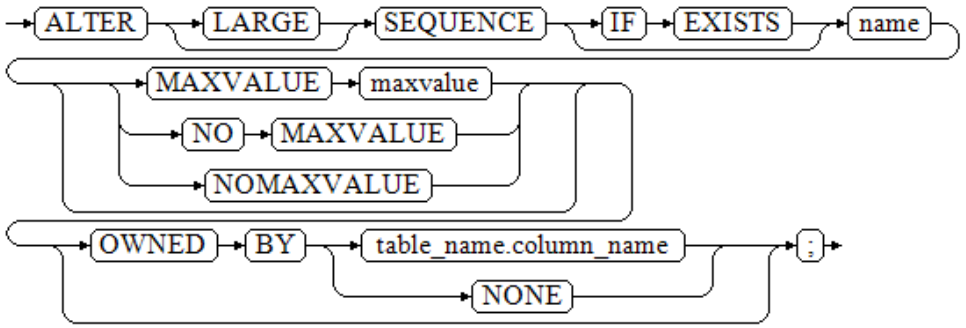
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If the LARGE flag is specified when a sequence is created, you also need to specify the LARGE flag when performing the ALTER operation.
- ALTER SEQUENCE blocks calls to the nextval, setval, currval, and lastval functions.

Syntax

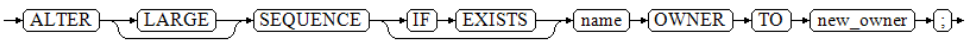
- Alter the owning column and maximum value of a sequence.

```
ALTER [LARGE] SEQUENCE [ IF EXISTS ] name
    [MAXVALUE maxvalue | NO MAXVALUE | NOMAXVALUE]
    [ OWNED BY { table_name.column_name | NONE } ] ;
```



- Alter the owner of a sequence.

```
ALTER [LARGE] SEQUENCE [ IF EXISTS ] name OWNER TO new_owner;
```



Parameters

- **LARGE**

  If the LARGE flag is specified when a sequence is created, you also need to specify the LARGE flag when performing the ALTER operation.

- **name**

  Specifies the name of the sequence to be modified.

- **IF EXISTS**

  Issues a notice rather than an error if the specified sequence does not exist.

- **MAXVALUE maxvalue | NO MAXVALUE | NOMAXVALUE**

  Specifies the maximum value of the sequence. The new maximum value must be greater than the current maximum value stored by GTM. If this parameter is not specified, the original maximum value is retained.

  Value range: (gtm_last_value, 2^{63} – 1]. **gtm_last_value** indicates the maximum value that has been allocated in the sequence. You can run the **\d** *Sequence name* command to query the value. If the LARGE flag is specified when a sequence is created, the value range is (gtm_last_value, 2^{127} – 1].

- **OWNED BY { table_name.column_name | NONE }**

  Associates a sequence with a specified column of a table. When the column or the table is deleted, the associated sequence is automatically deleted.

  – If the sequence was previously associated with another table, this new association will overwrite the existing one.

  – The associated table and sequence must be owned by the same user and reside in the same schema.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Using OWNED BY NONE removes any existing associations.
- **new_owner**

  Specifies the username of the new sequence owner. To change the owner, you must also be a direct or indirect member of the new role, and this role must have the CREATE permission on the sequence's schema.

Examples

```
-- Create an ascending sequence named serial, starting from 101.
gaussdb=# CREATE SEQUENCE serial START 101;

-- Create a table and specify default values for the sequence.
gaussdb=# CREATE TABLE t1(c1 bigint default nextval('serial'));

-- Change the owning column of serial to T1.C1.
gaussdb=# ALTER SEQUENCE serial OWNED BY t1.c1;

-- Delete a sequence and a table.
gaussdb=# DROP SEQUENCE serial CASCADE;
gaussdb=# DROP TABLE t1;

-- Create the seq_test sequence.
gaussdb=# CREATE SEQUENCE seq_test;

-- Query sequence information.
gaussdb=# \ds
              List of relations
 Schema |   Name   |   Type   | Owner | Storage
--------+----------+----------+-------+---------
 public | seq_test | sequence | omm   |
(1 row)

-- Create the user u_test and modify the sequence owner.
gaussdb=# CREATE USER u_test PASSWORD '********';
gaussdb=# ALTER SEQUENCE seq_test OWNER TO u_test;
gaussdb=# \ds
              List of relations
 Schema |   Name   |   Type   | Owner  | Storage
--------+----------+----------+--------+---------
 public | seq_test | sequence | u_test |
(1 row)

-- Delete the user and sequence.
gaussdb=# DROP SEQUENCE seq_test;
gaussdb=# DROP USER u_test;
```

Helpful Links

CREATE SEQUENCE and **DROP SEQUENCE**

1.13.7.30 ALTER SERVER

Description

Adds, alters, and deletes parameters of a specified server. You can query existing servers from the pg_foreign_server system catalog.

Precautions

- Only the server owner or a user granted the ALTER permission can run the **ALTER SERVER** command. By default, system administrators have the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

permission. To alter a server owner, you must be the server owner or system administrator and a member of the new owner role.

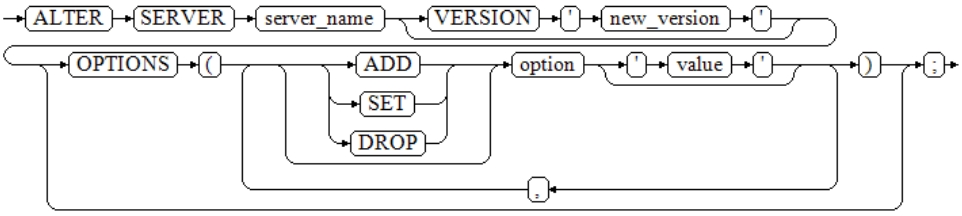
- When multi-layer quotation marks are used for sensitive columns (such as **password** and **secret_access_key**) in OPTIONS, the semantics is different from that in the scenario where quotation marks are not used. Therefore, sensitive columns are not identified for anonymization.

Syntax

- Alter parameters of a foreign server.

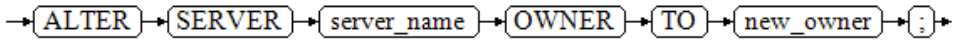
```
ALTER SERVER server_name [ VERSION 'new_version' ]
    [ OPTIONS ( {[ ADD | SET | DROP ] option ['value']} [, ... ] ) ];
```

In **OPTIONS**, **ADD**, **SET**, and **DROP** are operations to be performed. If these operations are not specified, ADD operations will be performed by default. **option** and **value** are the parameters of the corresponding operation.



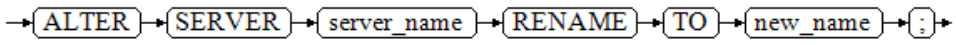
- Alter the owner of a foreign server.

```
ALTER SERVER server_name
    OWNER TO new_owner;
```



- Alter the name of a foreign server.

```
ALTER SERVER server_name
    RENAME TO new_name;
```



Parameters

- **server_name**

  Specifies the name of the foreign server to be altered.

- **new_version**

  Specifies the new version of the altered foreign server.

- **OPTIONS**

  Options for altering the server. The option name must be unique. The actions are ADD, SET, and DROP. If no action is specified, ADD is used. The name and value are also validated with the foreign data wrapper library of the server.

  In addition to the connection parameters supported by libpq, the following parameters are provided:

  - **fdw_startup_cost**

    Estimates the startup time required for a foreign table scan. This value usually contains the time taken to establish a connection, analyze the request at the remote end, and generate a plan. The default value is **100**. The value is a real number greater than 0.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - **fdw_tuple_cost**

    Specifies the additional consumption when each tuple is scanned on a remote server. The value specifies the extra consumption of data transmission between servers. The default value is **0.01**. The value is a real number greater than 0.

- **new_owner**

  Specifies the new owner of the server.

  NOTE

  To alter the owner of a server, you must be the owner of the foreign server and a direct or indirect member of the new owner role, and you must have the USAGE permission on the foreign data wrapper of the foreign server.

- **new_name**

  Specifies the new name of the server.

Examples

```
-- Create my_server.
gaussdb=# CREATE SERVER my_server FOREIGN DATA WRAPPER log_fdw;

-- Change the name of an external server.
gaussdb=# ALTER SERVER my_server RENAME TO my_server_1;

-- Create a user.
gaussdb=# CREATE USER jack PASSWORD '********';

-- Change the owner of the external server.
gaussdb=# ALTER SERVER my_server_1 OWNER TO jack;

-- Add the dbname and dbuser information to the external server.
gaussdb=# ALTER SERVER my_server_1 OPTIONS(ADD dbname 'test',dbuser' test_user');

-- Query the external server information.
gaussdb=# SELECT * FROM pg_foreign_server WHERE srvname='my_server_1';
   srvname   | srvowner | srvfdw | srvtype | srvversion | srvacl |            srvoptions
-------------+----------+--------+---------+------------+--------+----------------------------------
 my_server_1 |    22060 |  11826 |         |            |        | {dbname=test,"dbuser= test_user"}
(1 row)

-- Change the dbname in the external server to 'db_test'.
gaussdb=# ALTER SERVER my_server_1 OPTIONS (SET dbname 'db_test');

-- Query the external server information.
gaussdb=# SELECT srvname,srvoptions FROM pg_foreign_server WHERE srvname='my_server_1';
   srvname   |             srvoptions
-------------+-----------------------------------
 my_server_1 | {dbname=db_test,"dbuser= test_user"}
(1 row)

-- Delete the dbname information from the external server.
gaussdb=# ALTER SERVER my_server_1 OPTIONS (DROP dbname);

-- Query the external server information.
gaussdb=# SELECT srvname,srvoptions FROM pg_foreign_server WHERE srvname='my_server_1';
   srvname   |     srvoptions
-------------+--------------------
 my_server_1 | {"dbuser= test_user"}
(1 row)

-- Delete my_server_1.
gaussdb=# DROP SERVER my_server_1;

-- Drop the user.
gaussdb=# DROP USER jack;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

CREATE SERVER and **DROP SERVER**

1.13.7.31 ALTER SESSION

Description

Defines or alters the conditions or parameters that affect the current session. Any modified session parameters remain effective until the session is disconnected.

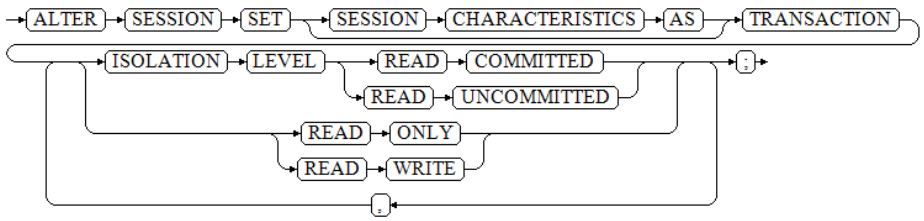
Precautions

- If SET TRANSACTION is executed without a prior START TRANSACTION, its effects will not be displayed, and the transaction will terminate immediately.
- To avoid using SET TRANSACTION, you may declare the required transaction mode(s) directly within START TRANSACTION. For details, see **START TRANSACTION**.

Syntax

- Set transaction parameters of a session.

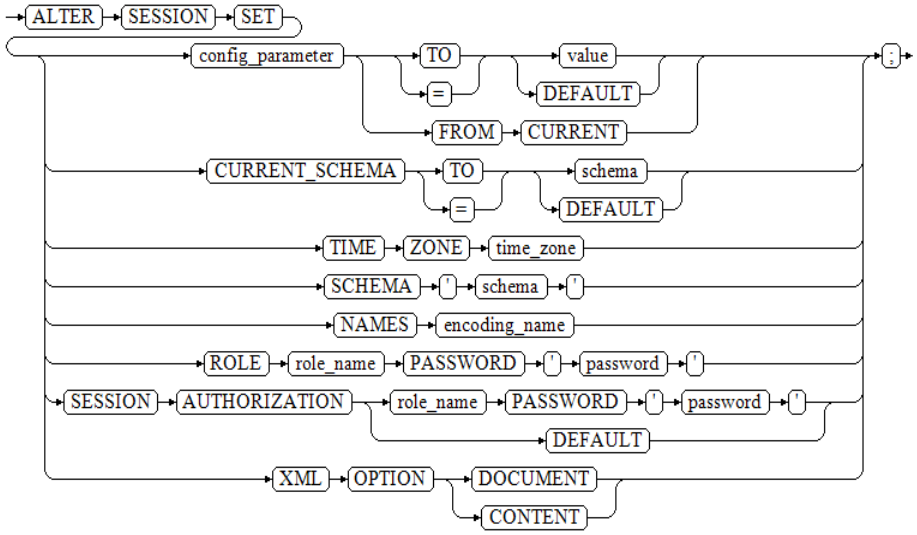
```
ALTER SESSION SET [ SESSION CHARACTERISTICS AS ] TRANSACTION
   { ISOLATION LEVEL { READ COMMITTED | READ UNCOMMITTED } | { READ ONLY  | READ
WRITE } } [, ...] ;
```



- Set other GUC parameters of a session.

```
ALTER SESSION SET
   {{config_parameter { { TO  | =  }  { value | DEFAULT } | FROM CURRENT }}
     | CURRENT_SCHEMA { TO | = } { schema | DEFAULT }
     | TIME ZONE time_zone
     | SCHEMA 'schema'
     | NAMES encoding_name
     | ROLE role_name PASSWORD 'password'
     | SESSION AUTHORIZATION { role_name PASSWORD 'password' | DEFAULT }
     | XML OPTION { DOCUMENT | CONTENT }
   } ;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- **config_parameter**

  Specifies the name of a configurable GUC parameter. You can use SHOW ALL to view available GUC parameters.

  - **value**

    Specifies a new value for **config_parameter**. It can be a string constant, identifier, number, or comma-separated list.

    - **DEFAULT**
    - **OFF**
    - **RESET**
    - User-specified value: The value must meet the restriction of the modified parameter.

  - **DEFAULT**

    Sets the parameter specified by **config_parameter** to its default value.

  - **FROM CURRENT**

    Assigns the value in the current session to **configuration_parameter**.

- **CURRENT_SCHEMA**

  Specifies the current schema.

  Value range: existing schema names. If the schema name does not exist, the value of **CURRENT_SCHEMA** will be empty.

- **TIME ZONE timezone**

  Specifies the local time zone for the current session.

  Value range: valid local time zones. The corresponding GUC parameter is **TimeZone**. The default value is **PRC**.

- **SCHEMA schema**

  Specifies the schema. Same as **CURRENT_SCHEMA**, the schema here is a string.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **NAMES encoding_name**

  Specifies the client-side encoding. It is equivalent to **set client_encoding to encoding_name**.

  Value range: valid encoding names. The corresponding GUC parameter is **client_encoding**, and the default encoding is **UTF8**.

- **role_name**

  Name of a role.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **password**

  Specifies the password for the role.

  Value range: It must comply with the password naming rules.

- **SESSION AUTHORIZATION**

  Specifies a user identifier for the current session.

- **XML OPTION { DOCUMENT | CONTENT }**

  Specifies the XML parsing mode.

  Value range: **CONTENT** (default value) or **DOCUMENT**.

- **ISOLATION LEVEL**

  Specifies the transaction isolation level, which determines the data that can be read (that is, data visibility) by a transaction when other concurrent transactions exist.

  Value range:

  – **READ COMMITTED** (default): Only committed data is read.

  – **READ UNCOMMITTED**: This isolation level behaves identically to **READ COMMITTED**.

Examples

- Set transaction parameters for a session.

  The keyword ALTER SESSION can be omitted in the example.

```
-- Start a transaction and set the transaction level.
gaussdb=# START TRANSACTION;
gaussdb=# ALTER SESSION SET TRANSACTION READ ONLY;
gaussdb=# END;
```

- Set other GUC parameters for a session.

  The keyword ALTER SESSION can be omitted in the example.

```
-- Create the ds schema.
gaussdb=# CREATE SCHEMA ds;

-- Set the search path of the schema.
gaussdb=# SET SEARCH_PATH TO ds, public;

-- Set the time/date type to the traditional postgres format (date before month).
gaussdb=# SET DATESTYLE TO postgres, dmy;

-- Set the encoding of the current session to UTF8.
gaussdb=# ALTER SESSION SET NAMES 'UTF8';

-- Set the time zone to PST8PDT.
gaussdb=# SET TIME ZONE 'PST8PDT';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Set the time zone to Europe/Rome.
gaussdb=# SET TIME ZONE 'Europe/Rome';

-- Set the current schema.
gaussdb=# ALTER SESSION SET CURRENT_SCHEMA TO tpcds;

-- Set XML OPTION to DOCUMENT.
gaussdb=# ALTER SESSION SET XML OPTION DOCUMENT;

-- Create the role joe, and set the session role to joe.
gaussdb=# CREATE ROLE joe WITH PASSWORD '********';
gaussdb=# ALTER SESSION SET SESSION AUTHORIZATION joe PASSWORD '********';

-- Switch to the default user.
gaussdb=> ALTER SESSION SET SESSION AUTHORIZATION default;

-- Drop the ds schema.
gaussdb=# DROP SCHEMA ds;

-- Drop joe.
gaussdb=# DROP ROLE joe;
```

Helpful Links

SET

1.13.7.32 ALTER SYNONYM

Description

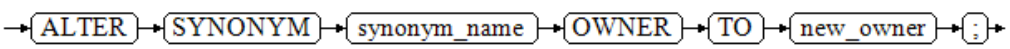
Alters attributes of a synonym.

Precautions

- Currently, only the owner of a synonym can be altered.
- Only system administrators have the permission to alter the owner of a synonym. When separation of duties is enabled, system administrators do not have the permission to alter the owner of a synonym by default.
- The new owner must have the CREATE permission on the schema where the synonym resides.
- PUBLIC synonyms cannot be modified.

Syntax

```
ALTER SYNONYM synonym_name
    OWNER TO new_owner;
```



Parameters

- **synonym_name**

  Specifies the name (optionally schema-qualified) of the synonym to be altered.

  Value range: character strings complying with **Identifier Naming Conventions**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **new_owner**

  Specifies the new owner of the synonym object.

  Value range: a string. It must be a valid username.

Examples

```
-- Create a system administrator.
gaussdb=# CREATE USER sysadmin WITH SYSADMIN PASSWORD '********';

-- Switch the system administrator.
gaussdb=#  \c - sysadmin

-- Create synonym t1.
gaussdb=#  CREATE OR REPLACE SYNONYM t1 FOR ot.t1;

-- Create user u1.
gaussdb=# CREATE USER u1 PASSWORD '********';

-- Assign permissions to the new user.
gaussdb=# GRANT ALL ON SCHEMA sysadmin TO u1;

-- Change the owner of synonym t1 to u1.
gaussdb=# ALTER SYNONYM t1 OWNER TO u1;

-- Delete synonym t1.
gaussdb=# DROP SYNONYM t1;

-- Revoke permissions from user u1.
gaussdb=# REVOKE ALL ON SCHEMA sysadmin FROM u1;

-- Delete user u1.
gaussdb=# DROP USER u1;

-- Switch to the initial user init_user. Replace init_user with the actual initial username.
gaussdb=# \c - init_user

-- Delete user sysadmin.
gaussdb=# DROP USER sysadmin;
```

Helpful Links

CREATE SYNONYM and **DROP SYNONYM**

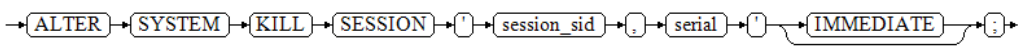
1.13.7.33 ALTER SYSTEM KILL SESSION

Description

ALTER SYSTEM KILL SESSION ends a session.

Syntax

```
ALTER SYSTEM KILL SESSION 'session_sid, serial' [ IMMEDIATE ];
```



Parameters

- **session_sid, serial**

  Specifies the SID and SERIAL of a session (for the format, see the example).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

You can use the pg_stat_activity system catalog to query the current active threads (see the examples). However, when you run the **ALTER SYSTEM KILL SESSION** command, the threads may have ended.

Value range: SIDs and SERIALs of all sessions that can be queried from the system catalog dv_sessions.

- **IMMEDIATE**

  Specifies that a session will be ended instantly after the statement is executed.

Examples

```
-- Start two sessions, and create a table and start a transaction to insert data in the first session.
gaussdb=# CREATE TABLE tbl_test(id int);
gaussdb=# BEGIN;
gaussdb=# INSERT INTO tbl_test VALUES (1);

-- Query the session information in the second session. If the value of state is idle in transaction, the transaction is waiting to be committed.
gaussdb=# SELECT t1.datname,
       t1.usename,
       t1.pid,
       t2.serial#,
       t1.state
FROM pg_stat_activity t1,
     dv_sessions t2
WHERE t1.query LIKE 'INSERT INTO tbl_test%'
  AND t1.sessionid = t2.sid;
 datname  | usename |       pid       | serial# |        state
----------+---------+-----------------+---------+---------------------
 postgres | omm     | 139802072635136 |       0 | idle in transaction
(1 row)

-- End a session. If the IMMEDIATE parameter is not specified, the session and transactions in the session are forcibly ended.
gaussdb=# ALTER SYSTEM KILL SESSION '139802072635136,0';
 pg_terminate_backend
----------------------
 t
(1 row)

-- Perform the reconnection and query the tbl_test table. The transaction is forcibly ended and the data is rolled back.
gaussdb=# SELECT * FROM tbl_test;
 id
----
(0 rows)

-- Drop the table.
gaussdb=# DROP TABLE tbl_test;
```

1.13.7.34 ALTER TABLE

Description

Alters table attributes, including altering table definition, renaming a table, renaming a specified column in a table, renaming a table constraint, setting table schemas, adding or updating multiple columns, and enabling or disabling row-level security policies.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

- If the base table is distributed in hash mode and you want to create a primary key index or unique index that does not contain the distribution key of the base table, you need to use a global secondary index (specify the **BY GLOBAL INDEX** column). If you want to create a primary key index or unique index that contains the distribution key of the base table, you need to use an ordinary index (do not specify the **BY GLOBAL INDEX** column). In single-DN deployment mode, you can use either a global secondary index or an ordinary index to create a primary key index or unique index that does not contain the distribution key of the base table. If the base table is not distributed in hash mode, you can only use an ordinary index to create a primary key index or unique index that does not contain the distribution key of the base table. That is, the index key must contain the distribution key of the base table.
- The table owner, users with the ALTER permission on the table, and users with the ALTER ANY TABLE permission are authorized to run the **ALTER TABLE** command. System administrators have this authorization by default. To modify the owner or schema of a table, you must be the table owner or a system administrator. When separation of duties is enabled, you must also be a member of the new owner's role.
- You cannot modify the tablespace of a partitioned table, but you can modify the tablespaces of individual partitions.
- The storage parameter **ORIENTATION** cannot be modified.
- SET SCHEMA does not allow switching to system-internal schemas; it currently supports switching only between user-defined schemas.
- The distribution key (or column) of a table cannot be modified.
- The partition key columns of a partitioned table cannot be changed or the character set cannot be converted.
- The database does not allow adding an auto-increment column or a column with a default value that includes the NEXTVAL() expression.
- The **rowid** and **rowno** system columns cannot be renamed or deleted, their default values cannot be set, and their definitions cannot be modified.
- Row-level security cannot be enabled for foreign tables and temporary tables.
- Constraints cannot be added to system columns.
- Dropping a PRIMARY KEY constraint by name does not remove the NOT NULL constraint. If necessary, drop the NOT NULL constraint manually.
- When JDBC is used, the default value can be set through PrepareStatement.
- When a new column is added using ADD COLUMN, the existing rows associated with that column in the table will receive the default value specified in the DEFAULT clause. However, if no default value is specified, the existing rows will default to **NULL**.
- If the default value of the new column is **NULL**, no full table update will be triggered.
- If a default value is specified for a new column, the following requirements must be met. Otherwise, additional full table update overhead will be generated, affecting online services.
  - The length of the default value does not exceed 128 bytes.
  - The default value does not contain volatile functions. If you are not sure whether it contains volatile functions, check the **provolatile** attribute of

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

the functions in the PG_PROC system catalog. Volatile functions will have a value of **'v'**.

- If you add or modify an ILM policy for a data object and include a row-level expression, you must use only the whitelisted functions (in **Row Expression Function Whitelist**) as the expression.

  NOTE

  Among the functions supported by the row-level expression ON (EXPR) in the ILM policy, the output of some may be influenced by compatibility parameters. For instance, after you set **b_format_version** to **'5.7'** and **b_format_dev_version** to **'s2'** in MySQL-compatible mode, the upper function will be unable to convert data to uppercase as usual.

- The number of table constraints cannot exceed 32,767.
- When performing online DDL operations, reserve sufficient disk space. That is, the remaining space must be the total space of the tables to be operated and the accessories such as indexes on the tables.
- Online DDL operations cannot be executed within a transaction or a stored procedure. If you use the JDBC driver to run multiple SQL statements, a transaction is implicitly started, and these SQL statements cannot be executed together.
- If online DDL execution is canceled, the residual data clearance process starts first. If the execution is canceled again, the clearance process ends and residual data remains.
- Resource contention exists between online DDL operations and user services. Therefore, online DDL operations are not suitable for heavy-traffic scenarios. In scenarios where resources are limited, fine-grained resource control is recommended. Parallel acceleration is not recommended.
- Online DDL operations are not supported in M-compatible databases.
- In the following cases, online DDL operations do not take effect. By default, DDL operations are performed offline. If **ONLINE** is specified, a message is displayed, indicating that the operation is not allowed.
  - A clause contains column_constraint or any of the keywords COMMENT, AUTO_INCREMENT, CHARACTER SET, CHARSET, and COLLATE when you change column types, precisions, and lengths online by using either of the following syntaxes: MODIFY column_name data_type { { [COMMENT 'string'] [...] } |AUTO_INCREMENT } or MODIFY [COLUMN] column_name data_type [CHARACTER SET |CHARSET charset] [{[COLLATE collation] | [column_constraint]} [...]].
  - The following syntax is used to change column types, precisions, or lengths online: CHANGE [COLUMN] column_name new_column_name data_type[CHARACTER SET | CHARSET charset] [{[COLLATE collation] |[column_constraint]} [...]].
  - A column type is changed or converted to the SET data type online.
  - The ALTER TABLE statement contains operations that are not supported by the current online DDL feature. (The supported online DDL operations include changing column type, precision, or length, moving table-level tablespaces, and adding NOT NULL, CHECK, PRIMARY KEY, or UNIQUE constraints.)
  - A table on which online DDL operations (including changing column types, precisions, and lengths, and moving table-level tablespaces) are

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

performed contains GSI, or a statement contains the BY GLOBAL INDEX keyword.

- Online DDL operations do not take effect when they are conducted on database-level, index-level, subpartitioned, segment-page, hash bucket, temporary, or unlogged tables. (Such DDL operations include modifying the column type, precision, and length, moving tablespaces at the table level, and adding the NOT NULL, CHECK, PRIMARY KEY, or UNIQUE constraints.) By default, these DDL operations are executed offline, so a message will indicate that the current DDL statements cannot be executed online.
- Long transactions may block online DDL operations (including changing the column type, precision, and length online, moving table-level tablespaces online, and adding NOT NULL, CHECK, PRIMARY KEY, and UNIQUE constraints online). Therefore, you are advised not to perform online DDL operations when long transactions exist.
- At the later stage of execution, online DDL operations (including changing the column type, precision, or length, moving table-level tablespaces, and adding NOT NULL, CHECK, PRIMARY KEY, or UNIQUE constraints) of this feature will block DQL and DML operations. Therefore, for a service model with a timeout error reporting mechanism, requests may fail due to blocking. If such an issue occurs, you can retry the requests.
- During the execution of online DDL statements, avoid executing other DDL statements concurrently. If exceptions like no response or error reporting occur, reconnect to the client and try again.
- During the upgrade from a version that does not support the online DDL operations to a version that supports it, the operations (including changing the column type, precision, or length, moving table-level tablespaces, and adding NOT NULL, CHECK, PRIMARY KEY, or UNIQUE constraints) are not supported because they are under observation. By default, DDL operations are performed offline, and a message is displayed indicating that online DDL operations are not supported during the upgrade.
- During the execution of an online DDL operation for altering table partitions (including changing the column type, precision, or length, moving table-level tablespaces, and adding NOT NULL, CHECK, PRIMARY KEY, or UNIQUE constraints), a specific schema is generated. The schema naming format is online$$ddl$$[hash value of (*Name of the schema where the table is located Table name*)].
  - By default, the owner of the schema is the initial user. You are advised not to operate the schema during online DDL operations; otherwise, the online DDL operations may be terminated and an unknown error may occur. You are advised not to query tables in the schema. By default, using gs_dump will not export the schema.
  - Do not create a schema with the prefix online$$. Otherwise, an error message is displayed. If a schema with the same name already exists, online DDL operations will not take effect. By default, DDL operations are executed offline, and a message is displayed indicating that the current DDL statement cannot be executed online due to a schema name conflict.
- If an online DDL operation (including modifying the column type, precision, or length, or moving table-level tablespaces) fails or an online DDL operation (adding NOT NULL, CHECK, PRIMARY KEY, or UNIQUE constraints) is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

performed, the table's storage space may bloat significantly. The bloat degree is proportional to the DDL statement execution duration and the number of concurrent write operations. You are advised not to perform online DDL operations when long transactions exist.

- If online DDL operations include adding NOT NULL, CHECK, PRIMARY KEY, or UNIQUE constraints, the error reported when data that violates the constraints may be different from that for offline DDL operations.
- In an online DDL operation for adding a PRIMARY KEY or UNIQUE constraint, if the constraint contains an expression, the DDL operation is performed offline by default. If the ONLINE keyword exists, a notice is displayed, indicating that the DDL operation cannot be performed online.
- Online DDL operations (including changing the column type, precision, or length, moving table-level tablespaces, and adding NOT NULL, CHECK, PRIMARY KEY, or UNIQUE constraints) inherit the usage restrictions from **CREATE TABLE LIKE**.
- Avoid performing online DDL operations and scaling operations simultaneously. Otherwise, such operations may fail.

Syntax

- Modify the definition of a table.

```
ALTER TABLE  [ OFFLINE | [ ONLINE [ WITH ( { online_parameter = value } [, ... ] ) ] ] ] [ IF EXISTS ]
 { table_name [*] | ONLY table_name | ONLY ( table_name ) }
    action [, ... ];

ALTER TABLE  [ OFFLINE | [ ONLINE [ WITH ( { online_parameter = value } [, ... ] ) ] ] ] [ IF EXISTS ]
table_name
    ADD ( { column_name data_type [ COLLATE collation ] [ column_constraint [ ... ] ] } [, ... ] );

ALTER TABLE  [ OFFLINE | [ ONLINE [ WITH ( { online_parameter = value } [, ... ] ) ] ] ] [ IF EXISTS ]
table_name
    MODIFY ( { column_name data_type | column_name [ CONSTRAINT constraint_name ] NOT NULL
[ ENABLE ] | column_name [ CONSTRAINT constraint_name ] NULL } [, ... ] );

ALTER TABLE  [ OFFLINE | [ ONLINE [ WITH ( { online_parameter = value } [, ... ] ) ] ] ] [ IF EXISTS ]
table_name
    RENAME [TO | AS | = ] new_table_name;
ALTER TABLE  [ OFFLINE | [ ONLINE [ WITH ( { online_parameter = value } [, ... ] ) ] ] ] [ IF EXISTS ]
    { table_name [*] | ONLY table_name | ONLY ( table_name ) }
    RENAME [ COLUMN ] column_name TO new_column_name;

ALTER TABLE  [ OFFLINE | [ ONLINE [ WITH ( { online_parameter = value } [, ... ] ) ] ] ] [ IF EXISTS ]
    { table_name [*] | ONLY table_name | ONLY ( table_name ) }
    RENAME CONSTRAINT constraint_name TO new_constraint_name;

ALTER TABLE  [ OFFLINE | [ ONLINE [ WITH ( { online_parameter = value } [, ... ] ) ] ] ] [ IF EXISTS ]
table_name
    SET SCHEMA new_schema;

ALTER TABLE  [ OFFLINE | [ ONLINE [ WITH ( { online_parameter = value } [, ... ] ) ] ] ] [ IF EXISTS ]
table_name
    GSIWAITALL;
```

  The table operation ACTION is as follows:

```
{ distribute_clause
    [ DISTRIBUTE BY { REPLICATION | HASH ( column_name [,...] )
    | RANGE ( column_name [,...] ) { SLICE REFERENCES tablename | ( slice_less_than_item [,...] )
    | ( slice_start_end_item [,...] ) }
    | LIST ( column_name [,...] ) { SLICE REFERENCES tablename | ( slice_values_item [,...] ) }
    } ]
    [ TO { GROUP groupname | NODE ( nodename [,...] ) } ]
column_clause
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    | ADD table_constraint [ NOT VALID ]
    | ADD table_constraint_using_index
    | VALIDATE CONSTRAINT constraint_name
    | DROP CONSTRAINT [ IF EXISTS ]  constraint_name [ RESTRICT | CASCADE ]
    | DROP PRIMARY KEY
    | CLUSTER ON index_name
    | SET WITHOUT CLUSTER
    | SET WITH ROWID
    | SET ( {storage_parameter = value} [, ... ] )
    | RESET ( storage_parameter [, ... ] )
    | OWNER TO new_owner
    | SET TABLESPACE new_tablespace
    | SET { COMPRESS | NOCOMPRESS }
    | ADD NODE ( nodename [, ... ] )
    | DELETE NODE ( nodename [, ... ] )
    | UPDATE SLICE LIKE table_name
    | DISABLE TRIGGER { trigger_name | ALL | USER }
    | ENABLE TRIGGER { trigger_name | ALL | USER }
    | ENABLE REPLICA TRIGGER trigger_name
    | ENABLE ALWAYS TRIGGER trigger_name
    | ENABLE ROW LEVEL SECURITY
    | DISABLE ROW LEVEL SECURITY
    | FORCE ROW LEVEL SECURITY
    | NO FORCE ROW LEVEL SECURITY
    | ENCRYPTION KEY ROTATION
    | REPLICA IDENTITY { DEFAULT | USING INDEX index_name | FULL | NOTHING }
    | AUTO_INCREMENT [ = ] value
    | COMMENT [ = ] 'string'
    | [ [ DEFAULT ] CHARACTER SET | CHARSET [ = ] default_charset ] [ [ DEFAULT ] COLLATE [ = ]
default_collation ]
    | ILM ADD POLICY ROW STORE COMPRESS ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day |
days | month | months | year | years } OF NO MODIFICATION [ ON ( EXPR )]
    | [ MODIFY { PARTITION partition_name | SUBPARTITION subpartition_name }] ILM { ENABLE |
DISABLE | DELETE } POLICY policy_name
    | [ MODIFY { PARTITION partition_name | SUBPARTITION subpartition_name }] ILM { ENABLE_ALL
| DISABLE_ALL | DELETE_ALL}
    | [ COLVIEW | COLVIEW NOCOLVIEW | NOCOLVIEW COLVIEW] [ ( { column_name [, ... ]} ) ]
[ PRIORITY { HIGH | LOW | NONE } ]
    | NOCOLVIEW }
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- **ADD table_constraint [NOT VALID]**

  Adds a table constraint.

- **ADD table_constraint_using_index**

  Adds a PRIMARY KEY constraint or UNIQUE constraint to a table based on the existing unique index. Currently, an existing GSI index cannot be used to perform the ADD table_constraint_using_index operation on a table.

- **VALIDATE CONSTRAINT constraint_name**

  Validates a CHECK constraint created with the **NOT VALID** option, and scans the entire table to ensure that all rows meet the constraint. If a constraint is marked as valid, the syntax will not alter its behavior.

- **DROP CONSTRAINT [IF EXISTS] constraint_name [RESTRICT | CASCADE]**

  Drops a table constraint.

- **DROP PRIMARY KEY**

  Drops a PRIMARY KEY constraint from a table. This syntax is supported only in MySQL-compatible and ORA-compatible modes.

- **CLUSTER ON index_name**

  Selects the default index for CLUSTER operations. It does not actually re-cluster the table.

- **SET WITHOUT CLUSTER**

  Deletes the most recently used CLUSTER index from the table. This syntax affects CLUSTER operations where the default index is not declared.

- **SET WITH ROWID**

  Adds the system columns **rowid** and **rowno** to a table and sets the storage option **hasrowid** to **on** for the table.

  - This syntax cannot be used for system catalogs, local temporary tables, global temporary tables, unlogged tables, foreign tables, hash bucket tables, or replication tables.
  - This syntax cannot be used for tables that have the **OID** system column or the **gs_tuple_uid** system column enabled, that is, tables with the **oids** storage option (or the WITH OIDS clause) or the **hasuids** storage option specified.
  - This syntax cannot be used for tables that contain columns named **rowid** or **rowno**.
  - This syntax is valid only in ORA-compatible mode.
  - After the SET WITH ROWID operation is performed on compressed data, the data is automatically decompressed.

- **SET ({storage_parameter = value} [, ...])**

  Changes one or more storage parameters for the table.

  If the value of **table_name** is an index name, **ACTIVE_PAGES** specifies the number of index pages, which may be less than the actual number of physical file pages and can be used for optimizer optimization. Currently, this parameter takes effect only on the local indexes of Ustore partitioned tables and will be updated by ANALYZE and VACUUM (including AUTOVACUUM). You are advised not to manually set this parameter because it is invalid in distributed mode.

- **RESET (storage_parameter [, ...])**

  Resets one or more storage parameters to their defaults. As with SET, re-altering a table based on different parameters might be needed to meet the expectations.

- **OWNER TO new_owner**

  Changes the owner of a table, sequence, or view to the specified user.

  The following operations are not allowed. If you perform them, an error will be reported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1. A SYSADMIN changes the owner of the table to the initial user.

2. In separation-of-duties mode, a SYSADMIN changes the owner of a table to a CREATEROLE or an AUDITADMIN.

- **SET TABLESPACE new_tablespace**

  Changes the table's tablespace to the specified tablespace and moves the data files associated with the table to the specified tablespace. Indexes on the table are not moved; the index's tablespace can be modified using the **SET TABLESPACE** option of the ALTER INDEX syntax.

- **DISTRIBUTE BY { REPLICATION | HASH (column_name [, ...]) | RANGE (column_name [, ...]) { SLICE REFERENCES tablename | (slice_less_than_item [, ...]) | (slice_start_end_item [, ...]) } | LIST (column_name [, ...]) { SLICE REFERENCES tablename | (slice_values_item [, ...]) }}**

  Specifies how a table is distributed or replicated among nodes. For details, see the description of DISTRIBUTE BY in **CREATE TABLE**.

- **TO { GROUP groupname | NODE (nodename [, ...]) }**

  Only available in extended mode (when GUC parameter **support_extended_features** is **on**). Exercise caution when enabling the mode. It is mainly used for internal scale-out tools. You are advised not to use the mode. This command only modifies the logical mapping relationship of the table distribution nodes and does not migrate the table's metadata and data on the DN.

- **ADD NODE (nodename [, ...])**

  This syntax is primarily intended for internal scale-out tools. You are advised not to use it directly.

- **DELETE NODE (nodename [, ...])**

  This syntax is primarily intended for internal scale-in tools. You are advised not to use it directly.

- **UPDATE SLICE LIKE table_name**

  This syntax is primarily intended for internal scale-in and scale-out tools. You are advised not to use it directly.

- **DISABLE TRIGGER { trigger_name | ALL | USER }**

  Disables a specified trigger. **trigger_name** specifies a trigger to be disabled. **ALL** disables all triggers, and **USER** disables only user triggers (excluding internally generated constraint triggers, such as those for deferrable unique and exclusion constraints).

  Exercise caution when using this functionality. Once a trigger is disabled, the integrity of the expected constraints cannot be ensured.

- **ENABLE TRIGGER { trigger_name | ALL | USER }**

  Enables the specified trigger. **trigger_name** specifies a trigger to be enabled. **ALL** enables all triggers, and **USER** enables only user triggers.

- **ENABLE REPLICA TRIGGER trigger_name**

  The trigger mechanism is affected by the GUC parameter **session_replication_role**. When a session is in "replica" mode, only triggers configured with ENABLE REPLICA are triggered. When a session is in the "origin" (default) or "local" mode, only triggers that are not configured with ENABLE REPLICA are triggered.

- **ENABLE ALWAYS TRIGGER trigger_name**

  Specifies a trigger as **ENABLE ALWAYS TRIGGER**, so that it will be triggered consistently despite the current setting of **session_replication_role** in the **replica** mode.

- **{ DISABLE | ENABLE } [REPLICA | ALWAYS] RULE**

  Disables or enables a specified rule.

  - **Disable [Replica | Always] Rule**: disables a specified rule.
  - **ENABLE REPLICA**: The session is started only when the session is in **replica** mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **ENABLE ALWAYS**: specifies a trigger as **ENABLE ALWAYS TRIGGER**, so that it will be triggered consistently despite the current setting of **session_replication_role** in the **replica** mode.

The ON SELECT rule cannot be disabled because it is related to the view implementation. If a rewriting rule is configured for a table, disabled rules are visible in the system and are disabled only during query rewriting. The rule triggering mechanism is simultaneously affected by the configuration variable **session_replication_role**.

- **{ DISABLE | ENABLE } ROW LEVEL SECURITY**

  Enables or disables row-level security for a table. The setting status will be updated to PG_CLASS.RELOPTIONS on each execution node to ensure that the system catalog metadata is consistent.

  If row-level security is enabled for a data table but no row-level security policy is defined, the row-level access to the data table is not affected. If row-level security for a table is disabled, the row-level access to the table is not affected even if a row-level security policy has been defined. For details, see **CREATE ROW LEVEL SECURITY POLICY**.

- **{NO FORCE|FORCE} ROW LEVEL SECURITY**

  Forcibly enables or disables row-level security for a table. The setting status will be updated to the PG_CLASS.RELOPTIONS of each execution node synchronously to ensure the metadata consistency of system catalogs.

  By default, a table owner is not affected by row-level security feature. However, if the feature is forcibly enabled for a table, the table owner (unless this user is a system administrator) will be affected.

- **ENCRYPTION KEY ROTATION**

  Rotation of the transparent data encryption key.

  - The data encryption key of a table can be rotated only when the database's TDE feature is enabled and the table's **enable_tde** is set to **on**.
  - After the transparent data encryption key rotation operation, the system automatically requests KMS to create a key. Data encrypted with the old key is still decrypted using the old key, while newly written data requires decryption with the new key. To ensure the security of encrypted data, you can periodically update the key based on the amount of new data in the encrypted table.

- **REPLICA IDENTITY { DEFAULT | USING INDEX index_name | FULL | NOTHING }**

  Specifies the record level of old tuples in UPDATE and DELETE statements on a table in logical replication scenarios.

  - **DEFAULT**: records the old value of the primary key column. If there is no primary key, **DEFAULT** does not record the old value.
  - **USING INDEX**: records the old values of columns covered by **index_name** (columns specified as **NOT NULL**), only including unique, global, and final values.
  - **FULL**: records the old values of all columns in the row.
  - **NOTHING**: does not record information in old rows.

  In logical replication scenarios, when the UPDATE and DELETE statements of a table are parsed, the parsed old tuples consist of the information recorded. For tables with primary keys, this option can be set to **DEFAULT** or **FULL**. For a table without a primary key, set this parameter to **FULL**. Otherwise, the old tuple will be parsed as empty during decoding. In general scenarios, you are advised not to set this parameter to **NOTHING**, as old tuples will always be parsed as empty. You are advised to define PRIMARY KEY or UNIQUE constraints for user tables to ensure deterministic replay. Otherwise, SQL statement replay may modify multiple rows, compromising data accuracy. For a Ustore table, setting **NOTHING** has no effect; the actual outcome is the same as setting **FULL**. For tables without a primary key, setting **DEFAULT** records all columns in the row.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **AUTO_INCREMENT [=] value**

  Sets the next increment of the auto-increment column.

  - This syntax takes effect only when **sql_compatibility** is set to **'MYSQL'**.
  - The configured **value** takes effect only when it is greater than the current value of the auto-increment counter and must be a non-negative number less than $2^{127} - 1$.

- **[[DEFAULT] CHARACTER SET | CHARSET [=] default_charset] [[DEFAULT] COLLATE [=] default_collation]**

  Modifies the character set and collation of a table to the default ones. The modification does not affect existing columns in the table.

- **ILM ADD POLICY ROW STORE COMPRESS ADVANCED [MEDIUM | HIGH] ROW AFTER n { day | days | month | months | year | years } OF NO MODIFICATION [ON (EXPR)]**

  Adds a specified ILM policy to a table. A complete ILM policy comprises ILM actions and ILM conditions. An ILM action defines a specific data compression or movement behavior, while an ILM condition defines a data compression or movement behavior that triggers the ILM action. The ILM condition is a row-level condition, applying to each row in a table. EXPR supports only table columns and the whitelisted functions. For details, see **Row Expression Function Whitelist**. Adding an ILM policy between the flashback point and the current point does not affect the flashback function.

- **[MODIFY { PARTITION partition_name | SUBPARTITION subpartition_name }] ILM { ENABLE | DISABLE | DELETE } POLICY policy_name**

  Modifies an ILM policy of a table (partition or subpartition). **policy_name** is the POLICY_NAME information queried from the system view GS_ADM_ILMOBJECTS or GS_MY_ILMOBJECTS.

- **[MODIFY { PARTITION partition_name | SUBPARTITION subpartition_name }] ILM { ENABLE_ALL | DISABLE_ALL | DELETE_ALL}**

  Modifies all ILM policies of a table (partition or subpartition).

– column_clause is as follows:

```
ADD [ COLUMN ] [ IF NOT EXISTS ] column_name data_type [ CHARACTER SET | CHARSET
charset ] [ COLLATE collation ] [ column_constraint [ ... ] ]
| MODIFY column_name data_type
| MODIFY column_name [ CONSTRAINT constraint_name ] NOT NULL [ ENABLE ]
| MODIFY column_name [ CONSTRAINT constraint_name ] NULL
| MODIFY column_name data_type { { [ COMMENT 'string' ] [ ... ] }
| DROP [ COLUMN ] [ IF EXISTS ] column_name [ RESTRICT | CASCADE ]
| ALTER [ COLUMN ] column_name [ SET DATA ] TYPE data_type [ ENCRYPTED WITH
( COLUMN_ENCRYPTION_KEY = column_encryption_key, ENCRYPTION_TYPE =
encryption_type_value ) ] [ COLLATE collation ] [ USING expression ]
| ALTER [ COLUMN ] column_name { SET DEFAULT expression | DROP DEFAULT }
| ALTER [ COLUMN ] column_name { SET | DROP } NOT NULL
| ALTER [ COLUMN ] column_name SET STATISTICS [PERCENT] integer
| ADD STATISTICS (( column_1_name, column_2_name [, ...] ))
| DELETE STATISTICS (( column_1_name, column_2_name [, ...] ))
| DISABLE STATISTICS (( column_1_name, column_2_name [, ...] ))
| ENABLE STATISTICS (( column_1_name, column_2_name [, ...] ))
| ALTER [ COLUMN ] column_name SET ( {attribute_option = value} [, ... ] )
| ALTER [ COLUMN ] column_name RESET ( attribute_option [, ... ] )
| ALTER [ COLUMN ] column_name SET STORAGE { PLAIN | EXTERNAL | EXTENDED | MAIN }
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- **ADD [COLUMN] [IF NOT EXISTS] column_name data_type [CHARACTER SET | CHARSET [=] charset] [COLLATE collation] [column_constraint [...]]**

  Adds a column to a table. When a new column is added using ADD COLUMN, the existing rows associated with that column in the table will receive the default value of the new column. (However, if the DEFAULT clause is not declared, the existing rows will default to **NULL**.) When IF NOT EXISTS is specified and columns with the same name exist, a notice is returned, indicating that the column already exists. When IF NOT EXISTS is not specified and columns with the same name exist, an error is returned.

- **ADD ({ column_name data_type } [, ...])**

  Adds multiple columns to a table.

- **MODIFY ({ column_name data_type | column_name [CONSTRAINT constraint_name] NOT NULL [ENABLE] | column_name [CONSTRAINT constraint_name] NULL } [, ...])**

  Modifies the data type of an existing column in a table. Running this command will clear the statistics of the specified column. It is advisable to re-collect the statistics after the modification.

- **MODIFY column_name data_type { { [COMMENT 'string'] [...] }**
  - Replaces the definition of existing column in a table with a new definition. The indexes and independent object constraints, such as primary keys, unique key and CHECK constraints in the previous column are not deleted.
  - This syntax can be used only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).
  - This syntax cannot modify foreign tables or encrypted columns.
  - This syntax cannot modify the data types or collations of partition key columns, or the data types or collations of columns referenced by rules or materialized views.
  - If a column referenced by a generated column undergoes a change in its data type or collation, data in the generated column will be regenerated.
  - If some objects (such as indexes, independent object constraints, views, triggers, and row-level security policies) depend on a modified column, these objects are rebuilt during column modification. If the definition of the modified column violates the constraints of this type of object, the modification fails. For example, the data type of the column that is used as the view result column cannot be modified. Evaluate the impact before making modifications.
  - In HASH, LIST, and RANGE distribution scenarios, the data type and comments of distribution keys cannot be modified.
  - If a modified column is called by some objects (such as user-defined functions and stored procedures), the modified column does not process these objects. After the column names are modified, these objects may be unavailable. Please evaluate the impact before modification.
  - This clause's syntax partially matches the MODIFY column_name data_type syntax from the previous clause but has different semantics. When the GUC parameter b_format_behavior_compat_options includes enable_modify_column, the system applies this clause's semantics.
  - Running this command will clear the statistics of the specified column. It is advisable to re-collect the statistics after the modification.

- **DROP [COLUMN] [IF EXISTS] column_name [RESTRICT | CASCADE]**

  Drops a column from a table. Indexes and constraints related to the column are automatically dropped. If an object not belonging to the table depends on the column (for example, view), CASCADE must be specified.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The DROP COLUMN statement does not physically remove the column, but simply makes it invisible to SQL operations. Subsequent INSERT and UPDATE operations on the table will store a **NULL** value for the column. Therefore, dropping a column is a quick process, but it does not immediately free up the disk space occupied by the dropped column (because the space has not been recycled yet). The space will be recycled when VACUUM is executed.

- **ALTER [COLUMN] column_name [SET DATA] TYPE data_type [ENCRYPTED WITH (COLUMN_ENCRYPTION_KEY = column_encryption_key, ENCRYPTION_TYPE = encryption_type_value)] [COLLATE collation] [USING expression]**

  Modifies the type of a column in a table. Indexes and simple table constraints on the column will automatically use the new data type by reparsing the originally supplied expression.

  If the original data type of a column and the modified data type are binary compatible, you do not need to rewrite the entire table when running this statement. In other scenarios, the entire table is rewritten. You can check whether the original type and target type are binary compatible in the PG_CAST system catalog. If **castmethod** is **b**, they are binary compatible. For example, if the data type of the source table is text and is converted to int, table rewriting is triggered. If it is converted to clob, table rewriting is not triggered. If table rewriting is triggered, the deleted space on the table is recycled immediately.

  Running this command will clear the statistics of the specified column. It is advisable to re-collect the statistics after the modification.

  Encrypted columns cannot be modified.

- **ALTER [COLUMN] column_name { SET DEFAULT expression | DROP DEFAULT }**

  Sets or removes the default value for a column. This default applies only to subsequent INSERT operations and does not affect existing rows in the table. Defaults can also be created for views, in which case they are inserted into INSERT statements on the view before the view's ON INSERT rule is applied.

- **ALTER [COLUMN] column_name { SET | DROP } NOT NULL**

  Marks a column as allowing or rejecting NULL values. You can only use SET NOT NULL when the column contains no null values.

- **ALTER [COLUMN] column_name SET STATISTICS [PERCENT] integer**

  Specifies the per-column statistics-collection target for subsequent ANALYZE operations. The target can be set in the range from 0 to 10000. Set it to **–1** to revert to using the default system statistics target.

- **{ADD | DELETE} STATISTICS ((column_1_name, column_2_name [, ...]))**

  Adds or deletes the declaration of collecting multi-column statistics to collect multi-column statistics as needed when ANALYZE is performed for a table or a database. The statistics about a maximum of 32 columns can be collected at a time. You are not allowed to add or delete such declaration for system catalogs or foreign tables.

- **{DISABLE | ENABLE} STATISTICS ((column_1_name, column_2_name [, ...]))**

  Sets the status of multi-column statistics. If multi-column statistics are disabled, related statistics are not collected, and the optimizer does not use the multi-column statistics during plan generation.

- **ALTER [COLUMN] column_name SET ({attribute_option = value} [, ...])**

  ALTER [COLUMN] column_name RESET (attribute_option [, ...])

  Sets or resets per-attribute options.

  Currently, the only defined per-attribute options are **n_distinct** and **n_distinct_inherited**. **n_distinct** affects statistics of a table, while **n_distinct_inherited** affects the statistics of the table and its subtables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Currently, only **SET/RESET n_distinct** is supported, and **SET/RESET n_distinct_inherited** is forbidden.

- **ALTER [COLUMN] column_name SET STORAGE { PLAIN | EXTERNAL | EXTENDED | MAIN }**

  Sets the storage mode for a column. It specifies whether this column is held inline or in an attached table, and whether the data should be compressed. SET STORAGE itself does not change anything in the table. It sets the strategy to be pursued during future table updates.

- **column_constraint**:

```
[ CONSTRAINT constraint_name ]
  { NOT NULL |
    NULL |
    CHECK ( expression ) |
    DEFAULT default_expr |
    ON UPDATE update_expr |
    AUTO_INCREMENT |
    COMMENT 'string' |
    COLVIEW |
    NOCOLVIEW |
    UNIQUE [KEY] index_parameters |
    PRIMARY KEY index_parameters |
    ENCRYPTED WITH ( COLUMN_ENCRYPTION_KEY = column_encryption_key,
ENCRYPTION_TYPE = encryption_type_value ) |
    REFERENCES reftable [ ( refcolumn ) ] [ MATCH FULL | MATCH PARTIAL | MATCH
SIMPLE ]
      [ ON DELETE action ] [ ON UPDATE action ] }
  [ DEFERRABLE | NOT DEFERRABLE ] [ INITIALLY DEFERRED | INITIALLY IMMEDIATE ]
```

  - **index_parameters**:

```
[ WITH ( {storage_parameter = value} [, ... ] ) ]
  [ USING INDEX TABLESPACE tablespace_name ]
```

  - **update_expr**:

```
{ CURRENT_TIMESTAMP | LOCALTIMESTAMP | NOW() }
```

- **table_constraint_using_index** used to add the PRIMARY KEY constraint or UNIQUE constraint based on the unique index is as follows:

```
[ CONSTRAINT constraint_name ]
  { UNIQUE | PRIMARY KEY } USING INDEX index_name
  [ DEFERRABLE | NOT DEFERRABLE ] [ INITIALLY DEFERRED | INITIALLY IMMEDIATE ]
```

- **table_constraint**:

```
[ CONSTRAINT [ constraint_name ] ]
  { CHECK ( expression ) |
    UNIQUE [ idx_name ] [ USING method ] ( { { column_name [ ( length ) ] | ( expression ) }
[ ASC | DESC ] } [, ... ] ) index_parameters |
    PRIMARY KEY [ USING method ] ( { column_name [ ASC | DESC ] } [, ... ] )
index_parameters |
    FOREIGN KEY [ idx_name ] ( column_name [, ... ] ) REFERENCES reftable [ ( refcolumn
[, ... ] ) ] }
  [ DEFERRABLE | NOT DEFERRABLE ] [ INITIALLY DEFERRED | INITIALLY IMMEDIATE ]
  { [ COMMENT 'string' ] [ ... ] }
```

index_parameters:

```
[ WITH ( {storage_parameter = value} [, ... ] ) ]
  [ USING INDEX TABLESPACE tablespace_name ][BY GLOBAL INDEX]
```

NOTE

If **index_parameters** is set to **BY GLOBAL INDEX**, the global secondary index is used to create constraints.

- Rename a table. The renaming does not affect stored data.

```
ALTER TABLE [ IF EXISTS ] table_name
  RENAME [ TO | AS | = ] new_table_name;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

If you specify this parameter in a version 5.7 MySQL-compatible database (**sql_compatibility** set to **'MYSQL'**, **b_format_version** set to **'5.7'**, and **b_format_dev_version** set to **'s2'**), the following situations may occur:

- If the character string corresponding to the new table name starts with "#mysql50#" and is followed by other characters, "#mysql50#" will be ignored.
- If the old and new table names are the same, no error is reported.

- Rename the specified column in the table.

```
ALTER TABLE [ IF EXISTS ] { table_name [*] | ONLY table_name | ONLY ( table_name ) }
    RENAME [ COLUMN ] column_name TO new_column_name;
```

- Rename the constraint of the table.

```
ALTER TABLE [ IF EXISTS ] { table_name [*] | ONLY table_name | ONLY ( table_name ) }
    RENAME CONSTRAINT constraint_name TO new_constraint_name;
```

- Set the schema of a table.

```
ALTER TABLE [ IF EXISTS ] table_name
    SET SCHEMA new_schema;
```

NOTE

- This form moves the table into another schema. Associated indexes and constraints are moved as well. Currently, modifying schemas for sequences is not supported. If the table has sequences, drop and rebuild them, or remove the ownership between the table and sequences to successfully modify the table's schema.
- To modify the schema of a table, you must have the CREATE permission on the new schema. To add the table as a new child of a parent table, you must own the parent table as well. To alter the owner, you must also be a direct or indirect member of the new owning role, and that member must have the CREATE permission on the table's schema. These restrictions ensure that you can only rebuild or drop the table. However, a system administrator can alter all permissions of any table anyway.
- All the forms of ALTER TABLE that act on a single table, except RENAME and SET SCHEMA, can be combined into a list of multiple modifications to be applied together. For example, it is possible to add several columns or modify the type of several columns in a single command. This is particularly useful with large tables, since only one pass over the tables need to be made.
- Adding a CHECK or NOT NULL constraint will scan the table to validate that existing rows meet the constraint.
- Adding a column with a non-null default or changing the type of an existing column will rewrite the entire table. Rewriting a large table may take much time and temporarily needs doubled disk space.

- Update columns.

```
ALTER TABLE [ IF EXISTS ] table_name
    MODIFY ( { column_name data_type | column_name [ CONSTRAINT constraint_name ] NOT NULL
[ ENABLE ] | column_name [ CONSTRAINT constraint_name ] NULL } [, ...] );
```

- Commit all DML transactions in the table when the SQL statement is executed.

```
ALTER TABLE [ IF EXISTS ] table_name GSIWAITALL IN lock_mode MODE;
```

NOTE

This syntax is primarily called by the internal CREATE GLOBAL INDEX CONCURRENTLY/REINDEX INDEX CONCURRENTLY to synchronize lock waits during online creation of GSIs. Based on the given lock level, you can obtain the list of transactions that conflict with that lock level on each node and wait for those transactions to complete. You are advised not to use it, as doing so may trigger an error.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **ONLINE | OFFLINE**

  Controls whether to enable online DDL operations.

  The keywords ONLINE and OFFLINE are used.

  – ONLINE: DDL operations are executed online. In this case, if the ALTER TABLE statement contains a scenario that is not supported by the current DDL feature, the DDL operations are executed offline and a notice is displayed, indicating that the ONLINE keyword is ignored.

  – OFFLINE: DDL operations are executed offline.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- Whether the data meets the data type conversion, length modification, or constraint requirements is checked during online DDL operations (including changing the column type, precision, and length, and adding NOT NULL, CHECK, PRIMARY KEY, and UNIQUE constraints). If the data does not meet the requirements, an error is reported during DDL operations. Online DDL operations support concurrent DML operations. Concurrent DML operations may delete data that does not meet conversion requirements or violates constraints, but an error may be still reported for the DDL operations.

  For example, when online DDL scans a table to change the column type, if the data of tuple A in the table does not meet data conversion requirements, an error is reported immediately when the online DDL transaction scans tuple A. At the same time, if tuple A is deleted by a parallel transaction DELETE after tuple A has been scanned by an online DDL operation, tuple A is not considered as deleted.
- Online DDL error troubleshooting: The system automatically creates a temporary schema as the DDL implementation tool and sets the table to the **append_mode** state during online DDL operations (including changing the column type, precision, and length, and adding NOT NULL, CHECK, PRIMARY KEY, and UNIQUE constraints). If the online DDL execution is complete, the temporary mode is automatically cleared and restored. Automatic cleanup may fail due to reasons such as lock timeout. In this case, manual cleanup is required. If a severe fault occurs, such as FATAL, PANIC, database fault, or node disconnection, the created temporary schema and table appending status may remain. In this case, you need to manually clear and restore the temporary schema and table. The scenarios are as follows:
  - Canceling online DDL operations: If you cancel online DDL operations during the execution, the system automatically clears the data. If the execution is canceled, the residual data clearance process starts first. If the execution is canceled again, the clearance process ends and residual data remains.
  - Operation error: If an error needs to be reported during the execution of a DDL command, for example, type conversion fails or length modification damages data, the system reports an error and automatically clears the data.
  - Major faults:
    1. Breakdown of a distributed system with one primary node and no standby node: If a CN or DN breaks down, you can manually clear the data using **online_ddl_cleanup()**, or the data will be automatically cleared when you run online DDL operations or online VACUUM FULL next time.
    2. Breakdown of a distributed system with one primary node and multiple standby nodes:
       - Breakdown of the CN where DDL operations are performed: You can manually clear the data using **online_ddl_cleanup()**, or the data will be automatically cleared when you run online DDL operations or online VACUUM FULL next time.
       - A CN where no DDL operation is executed is disconnected and breaks down in a distributed system with one primary node and multiple standby nodes and HA switchover. You need to wait until the switchover is complete and the cluster recovers (online DDL operation waits for 60s). You can retry up to three times. If the cluster recovers, you do not need to manually clear the fault. If the cluster fails to recover, you need to manually clear the fault using **online_ddl_cleanup()** after the cluster becomes normal. If the cluster is abnormal, retry may fail. In this case, you need to manually clear the fault.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

    - Adding PRIMARY KEY and UNIQUE constraints online: If an exception (for example, manual cancellation, duplicate unique index key values, insufficient resources, thread startup failure, or lock timeout) occurs during online index creation, the index fails to be created online. In this case, residual indexes and temporary tables may exist, occupying system resources. For details, see **CONCURRENTLY** in **CREATE INDEX**.
  - You are advised not to call **online_ddl_cleanup()** during online DDL execution. Otherwise, the online DDL operations may be terminated or an unknown error may occur.
  - When the column char type extension is modified online, the underlying data is not modified and rewritten. After online DDL operations, the underlying storage of existing data still retains the old length, and the underlying storage of new and modified data caused by DML operations complies with the extended length.

- **online_parameter**

  Specifies the online DDL option when the ONLINE keyword is specified. If the ONLINE keyword is not specified or online DDL conditions are not met, this parameter does not take effect.

  Value range:

  – **parallel_threads**: specifies the number of threads when parallelism is enabled. The value is an integer from 1 to 32, with **1** being the default.

    ▪ When **parallel_threads** is set to a value greater than **1**, multiple threads are enabled and the number of threads is specified.

    ▪ If **parallel_threads** is not set or is set to **1**, multi-thread acceleration is disabled.

      When a user enables resource and workload management, for example, I/O resource management (for example, **io_limits** > **0** or **io_priority** = **high**/**medium**/**low**), **parallel_threads** ignores resource and workload management under the user or session. In addition, a notice is displayed, indicating that the current resource and workload management is invalid and multi-thread acceleration is enabled. The multi-thread execution process of online DDL operations actually follows the resource and workload management of the initial user.

    ▪ This parameter can be used together with **parallel_workers** for acceleration. In this scenario, **parallel_workers** determines the maximum number of parallel threads within an index, while **parallel_threads** determines the maximum number of parallel threads between indexes.

  – **max_catchup_times**: specifies the maximum number of online DDL catchups to prevent long online DDL execution time. The value is an integer from 1 to *INT_MAX*, with **50** being the default.

    ▪ Online DDL execution records the DML incremental data generated during data reconstruction to a temporary table and performs multiple cyclic catchups to reconstruct the incremental data. If the amount of concurrent DML data is too large, online DDL catchup fails. To prevent this problem, you can set this parameter to limit the number of catchups.

  – **last_catchup_threshold**: specifies the upper limit of the time for triggering the last catchup. After a catchup round is complete, if the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

duration of this round is less than the threshold, the last round of catchup and switchover is triggered. Value range: [1000,1200000] (unit: ms). The default value is **5000**.

- **online_ddl_lockwait_timeout**: specifies the lock wait timeout interval for online DDL operations. If the thread waiting for the lock fails to obtain the lock within the specified time, a lock timeout error is reported. When the **CANCEL** mode is used to obtain the lock, the actual timeout interval is twice the value of this parameter. Value range: [1,1200000] (unit: ms). The default value is **10000**.
- **lockwait_interval**: indicates the lock wait increment, that is, the retry interval after the lock wait times out. If the lock wait times out in the current round, the system waits for the period specified by this parameter and then attempts to obtain the lock in the next round. Value range: [0,300] (unit: seconds). The default value is **5**.
- **trylock_threshold**: indicates the maximum number of lock wait attempts. It must be used together with **enable_cancel**. If the lock cannot be obtained due to lock wait timeout after the maximum number of attempts is reached, the **CANCEL** mode will be used for the next attempt. If **enable_cancel** is not enabled, the lock fails to be obtained and an error is reported after the maximum number of attempts is reached. Value range: [1,*INT_MAX*] or **-1** (indicating that there is no upper limit). The default value is **-1**.
- **enable_cancel**: specifies whether services can be canceled. That is, when the number of lock attempts reaches the upper limit or the level-8 lock of the user table is obtained during the table switching transaction, user services are quickly terminated to ensure that the online DDL operation can successfully obtain the lock. The value can be **false** or **true**. The default value is **false**.

- **IF EXISTS**

  Issues a notice rather than an error if no tables have identical names. The notice indicates that the table does not exist.

- **table_name [*] | ONLY table_name | ONLY (table_name)**

  table_name is the name of the table to be modified.

  If **ONLY** is specified before the table name, only that table is modified. If **ONLY** is not specified, the table and all its descendant tables (if any) are modified. You can explicitly add the asterisk (*) option following the table name to specify that all subtables are scanned, which is the default operation.

- **constraint_name**
  - Specifies the name of an existing constraint to be dropped in the DROP CONSTRAINT operation.
  - Specifies the name of a new constraint in the ADD CONSTRAINT operation.

  NOTICE

  For each new constraint, **constraint_name** is optional in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**) but required in other databases.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **index_name**

  Specifies the index name.

  NOTICE

  In the ADD CONSTRAINT operation:

  - **index_name** is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).
  - If both **constraint_name** and **index_name** are specified for a FOREIGN KEY constraint, the value of **constraint_name** will be used as the index name.
  - If both **constraint_name** and **index_name** are specified for a UNIQUE constraint, the value of **index_name** will be used as the index name.

- **USING method**

  Specifies the method used to create the index.

  For details about the value range, see **USING method** in **CREATE INDEX**.

  NOTICE

  In the ADD CONSTRAINT operation:

  - **USING method** is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).
  - In MySQL-compatible mode, when **USING method** is not specified, Astore will default to B-tree indexing, and Ustore will default to UB-tree indexing.
  - If the storage engine of a table is Ustore and the USING BTREE constraint is specified in an SQL statement, the underlying layer will automatically treat it as USING UBTREE.

- **ASC | DESC**

  ASC specifies an ascending (default) sort order. **DESC** specifies a descending sort order.

  NOTICE

  ADD CONSTRAINT supports **ASC | DESC** only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).

- **expression**

  Specifies an expression based on one or more columns of the table. It must be written in parentheses.

  NOTICE

  Expression indexes in the UNIQUE constraint are supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **storage_parameter**

  Specifies the name of a storage parameter.

  The following options are added for online scale-out:

  - **append_mode** (enumerated type)

    Scales out a table online or offline, or stops scaling it. You can modify certain content in the table during online scale-out but cannot do so during offline scaling.

    To modify a table that is being scaled, append new data so that they can be recorded as incremental data.

    - **on**: scales out a table online. New data will be appended.
    - **off**: stops scaling. New data will be written in normal mode, and options for online scaling will not be displayed in **pg_class.reloptions**.
    - **read_only**: scales a table offline, during which no other operations can be performed on the table.
    - **end_catchup**: reports errors for the write service in the last catchup. The read service is executed normally.

  - **rel_cn_oid** (OID type)

    Records the OID of tables on the current CN to generate **delete_delta** on the DNs.

    If **append_mode** is set to **on**, **rel_cn_oid** must be specified.

    The **append_mode** and **rel_cn_oid** options are used only for online scale-out tools.

  - **exec_step** (integer)

    Records resumable transmission steps in **relOptions** of the temporary table.

    Value range: [1,4]

    It can be used only for data redistribution.

  - **create_time** (long integer)

    Records the time when the temporary table is created during resumable transmission in **relOptions** of the temporary table.

    It can be used only for data redistribution.

  - **redistribute_version** (long integer)

    Specifies the number of times that a table is successfully redistributed during logical scale-out. The value is used by the standby node to read and verify consistency. The value is incremented by 1 each time the table is redistributed.

    This option is set only in the scale-out process and is not recommended.

  - **enable_update_distkey**

    Determines whether the table supports the UPDATE operation on distribution keys. The distribution keys can be updated only when this attribute of the table is set to **on** and constraints are met. Otherwise, distribution keys do not support the UPDATE operation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: **on** and **off**.

Default value: **off**.

NOTE

The constraints on updating distribution keys are as follows:

- Distribution keys can be updated only when **enable_update_distkey** is set to **on**.
- The UPDATE statement cannot be pushed down to DNs for execution. Instead, a PGXC plan is directly generated, and it remains unchanged regardless of the values before and after the distribution key updates.
- Tables with row-level UPDATE TRIGGER are not supported. Otherwise, the execution fails and an error message is displayed. The row-level INSERT/DELETE TRIGGER does not take effect, and the update-statement-level TRIGGER is executed normally.
- Concurrent update of the same row is not supported. You need to obtain a lock first. The result (returning **0** or reporting an error) of obtaining a lock on DNs depends on the setting of the GUC parameter **concurrent_dml_mode**. If an error is reported, the possible causes are as follows: (1) The error message "update distribute column conflict" is reported. (2) When the lock obtaining time exceeds the threshold, an error is reported, indicating that the lock times out.
- Tables with global secondary indexes (GSIs) are not supported. Otherwise, an error is reported.
- Only hash distribution is supported. LIST/RANGE distributed tables are not supported. Otherwise, an error is reported.
- MERGE INTO and UPSERT cannot be used to update distribution keys. Otherwise, an error is reported.
- GTM_FREE is not supported. Otherwise, an error is reported.
- UPDATE RETURNING is not supported. Otherwise, an error is reported.
- Statements with joined tables are not supported. Otherwise, an error is reported.
- UPDATE+LIMIT is not supported. Otherwise, an error is reported.

The following option is added for creating an index:

– **parallel_workers** (int type)

  Specifies the number of bgworker threads started for index creation. For example, a value of **2** indicates that two bgworker threads will be started to create indexes in parallel.

  Value range: [0,32]. **0** indicates that parallel index creation is disabled.

  Default value: unspecified, indicating that parallel index creation is disabled.

The following option is added to the replication table:

– **primarynode** (Boolean type)

  Default value: **off**.

  When **primarynode** is set to **on**, the primary node is selected for the replication table. Typically, the primary node is the first entry in the **nodeoids** column of the pgxc_class system catalog. When the IUD operation is performed on the replication table, the operation is delivered to the primary node first. After the result is received, the operation is delivered to other DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- logical_repl_node (string type)

  Specifies the name of the DN responsible for returning logical logs to the CN during logical decoding of a distributed replication table. For the replication table, if this parameter is not specified, the first node in the node group where the current table is located is used by default. When the RESET operation is performed on this option, the node is reset to the first node of current table.

  Value range: character strings.

  Default value: For the non-replication table, this parameter is empty by default. For the replication table, this parameter is set to the name of the first node by default.

TDE options:

- **enable_tde** (Boolean type)

  Specifies whether to set a table as an encrypted table. This parameter supports only row-store tables, segment-page tables, hash bucket tables, temporary tables, and unlogged tables. When setting **enable_tde** to **on**, ensure that the GUC parameter **enable_tde** is enabled and **tde_key_info** is properly set. For detailed usage, consult "Feature Usage Guidelines > Transparent Data Encryption" in *Developer Guide*.

  Value range: **on** and **off**.

  on: TDE is enabled.

  off: TDE is disabled.

  NOTE

  - When the value is switched from **on** to **off**, any inserted or updated data will remain encrypted when written to existing pages, but will no longer be encrypted on newly generated pages.
  - Conversely, switching the value from **off** to **on** means that inserted or updated data will not be encrypted when written to existing pages, but will be automatically encrypted on newly generated pages.

  In summary, newly generated pages follow the post-switch encryption status, whereas existing pages maintain the encryption status they had before the switch. You are advised to manually perform VACUUM FULL on the table after the encryption switchover to ensure that the encryption status of all data pages is consistent.

  Default value: **off**.

- **encrypt_algo** (string type)

  Specifies the algorithm for encrypting a table.

  Value range: **AES_128_CTR**, **SM4_CTR**, **AES_128_CTR_SHA256_HMAC**, **SM4_CTR_SM3_HMAC**, and **AES_128_GCM**.

  Default value: If **enable_tde** is **on** and the **table_algorithm** subparameter within the GUC parameter **tde_encrypt_config** is not empty, the default value is the same as that of **table_algorithm**. If **table_algorithm** is empty, the default value is **AES_128_CTR**. When **enable_tde** is **off**, the default is empty.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- Similar to **enable_tde**, after **encrypt_algo** is used, the old algorithm is still used when inserted or updated data is written to the old page, and the new algorithm is used when the data is written to the new page.
- You can also manually perform VACUUM FULL on the table to ensure that all data pages are encrypted using the new algorithm.

– **hasuids** (Boolean type)

Default value: **off**.

If this parameter is set to **on**, a unique table-level ID is allocated to a tuple when the tuple is updated.

– **statistic_granularity**

Records the default **partition_mode** when the table analyzes statistics. For details about **PARTITION_MODE**, see **ANALYZE | ANALYSE**. This parameter setting does not work for non-partitioned tables.

Value range: See the value range of **partition_mode**.

Default value: **AUTO**.

The following options are added for the ROWID feature:

– hasrowid

Specifies whether the table contains the **ROWID** and **ROWNO** system columns. If **hasrowid** is set to **on**, the **rowid** and **rowno** system columns are added to the table, and sequences and indexes are created on the **rowno** system column.

Value range: **on** and **off**.

Default value: **off**.

NOTE

- This parameter supports only ORA-compatible databases (that is, **sql_compatibility** is set to **ORA**).
- This parameter does not support local temporary tables, unlogged tables, global temporary tables, materialized views, hash bucket tables, or replication tables.
- This parameter cannot be specified together with **hasuids**.
- When this parameter is enabled, the ordinary column name cannot be defined as **rowno** or **rowid**.
- The **hasrowid** storage option of a table cannot be changed to a different value using the SET ({storage_parameter = value} [, ...]) or RESET (storage_parameter [, ...]) clause.

The following options are added to online DDL operations:

– **online_ddl_status** (enumerated type)

This parameter is used to update the online DDL execution status of a target table.

It is called in online DDL operations. You are advised not to use it.

- **on**: enables the online DDL status.
- **off**: disables the online DDL status.
- **init**: initializes online DDL temporary objects.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - **catchup**: synchronizes concurrent data increment for DDL operations.
  - **catchup_basedata**: synchronizes basic data of online DDL operations.
  - **scan_catchup**: performs catchup when NOT NULL or CHECK constraints are added online.
  - **scan_catchup_basedata**: checks the basic data about adding NOT NULL or CHECK constraints online.
  - **vacuum_catchup**: performs catchup when VACUUM FULL is performed online.
  - **vacuum_verbose_catchup**: performs catchup when VACUUM FULL VERBOSE is performed online.
  - **last_catchup**: reports errors for the write service in the last online DDL catchup.
  - **switch**: exchanges temporary tables in the last online DDL catchup.
  - **copy_switch**: exchanges temporary tables in the last online tablespace movement catchup.
  - **record_toast_oid**: records the OID of a TOAST table attached to a target table of online DDL operations.
  - **abort**: specifies that online DDL operations enter the abort state.
  - **flush_buffer**: forcibly flushes buffers to disks when tablespaces are moved online.
  - **tmp_rel**: indicates a temporary table created during an online DDL process.

- **online_ddl_lock_tag** (int type)

  Unique ID of a target table of online DDL operations, which is used to generate temporary objects. The ID is the hash value of (Table schema + Table name).

  It is called in online DDL operations. You are advised not to use it.

New options for VACUUM:

- vacuum_truncate

  Specifies whether to enable the VACUUM TRUNCATE function of a table.

  Value range: **on** and **off**.

  Default value: **on**

- **new_owner**

  Specifies the name of the new table owner.

- **new_tablespace**

  Specifies the new name of the tablespace to which the table belongs.

- **column_name**, **column_1_name**, **column_2_name**

  Specifies the name of a new or existing column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **data_type**

  Specifies the type of a new column or a new type of an existing column.

- **charset**

  Assigns a character set to the column. If a character set is explicitly specified, the column will use the default collation associated with that character set.

  This syntax is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).

- **collation**

  Assigns a collation to the column. The optional COLLATE clause specifies a collation for the new column; if omitted, the collation is the default for the new column. You can run the **select * from pg_collation** command to query collations from the pg_collation system catalog. The row starting with **default** in the query result represents the default collation.

  MySQL-compatible databases (**sql_compatibility = 'MYSQL'**) also support the utf8mb4_bin, utf8mb4_general_ci, utf8mb4_unicode_ci, binary, gbk_chinese_ci, gbk_bin, gb18030_chinese_ci, and gb18030_bin collations. For details, see **Table 1-398**.

  NOTE

  - Only the character type supports specifying a character set. When you specify the binary character set or collation, the system will convert the character type to the binary type. If the system cannot find a matching type mapping, it will report an error. Currently, only the TEXT-to-BLOB mapping is supported.
  - In addition to the binary character set and collation, you can only specify the character set that matches the database encoding.
  - If the character set or collation of a column is not explicitly specified but the default character set or collation of the table is specified, the column will inherit the character set and collation from the table. If the default character set or collation of a table does not exist, the character set and collation of table columns inherit the character set and collation of the current database when **b_format_behavior_compat_options** contains **'default_collation'**.
  - If the modified character set or the character set corresponding to the modified collation differs from the character set of the current column, data in the column will be converted to the specified character set for encoding.

- **USING expression**

  Specifies how to compute the new column value from the old; if omitted, the default conversion is an assignment cast from old data type to new. A USING clause must be provided if there is no implicit or assignment cast from the old to new type.

  NOTE

  USING in ALTER TYPE can specify any expression involving the old values of the row; that is, it can refer to any columns other than the one being cast. This allows general casting to be done with the ALTER TYPE syntax. Because of this flexibility, the USING expression is not applied to the column's default value (if any); the result might not be a constant expression as required for a default. This means that when there is no implicit or assignment cast from old to new type, ALTER TYPE might fail to convert the default even though a USING clause is supplied. In such cases, drop the default with DROP DEFAULT, perform ALTER TYPE, and then use SET DEFAULT to add a suitable new default. Similar considerations apply to indexes and constraints involving the column.

- **NOT NULL | NULL**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies whether the column allows null values.

- **ENABLE**

  Enables the constraint. This is the default.

- **integer**

  Specifies the constant value of a signed integer. When using **PERCENT**, the range of **integer** is from 0 to 100.

- **attribute_option**

  Specifies an attribute option.

- **PLAIN | EXTERNAL | EXTENDED | MAIN**

  Specifies a column-store mode.

  – **PLAIN** must be used for fixed-length values (such as integers). It must be inline and uncompressed.

  – **MAIN** is for inline, compressible data.

  – **EXTERNAL** is for external, uncompressed data. Use of **EXTERNAL** will make substring operations on **text** and **bytea** values run faster, at the penalty of increased storage space.

  – **EXTENDED** is for external, compressed data. **EXTENDED** is the default for most data types that support non-**PLAIN** storage.

- **CHECK (expression)**

  Ensures that for any inserted or updated row, the expression must evaluate to true. Otherwise, an exception is raised and no changes are made to the database.

  A CHECK constraint specified as a column constraint should reference that column's value only, while an expression appearing in a table constraint can reference multiple columns.

  Currently, **CHECK (expression)** does not include subqueries and cannot use variables apart from the current column.

- **DEFAULT default_expr**

  Assigns a default data value to a column.

  The data type of the default expression must match the data type of the column.

  The default expression will be used in any insert operation that does not specify a value for the column. If there is no default value for a column, then the default value is **NULL**.

- **ON UPDATE update_expr**

  Specifies an attribute constraint for a column.

  During an UPDATE operation on a tuple in a table, if the new values of the updated columns differ from the old values, columns (in the tuple) that have this attribute but are not part of the UPDATE will be automatically updated to the current timestamp. However, if the new values of the updated columns are the same as the old values, those columns will remain unchanged. If columns with this attribute are included in the UPDATE, their values will be updated to the specified value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This attribute can be specified in MySQL-compatible databases version 5.7 (**sql_compatibility = 'MYSQL'**, **b_format_version='5.7'**, and **b_format_dev_version='s1'**). It can also be specified in PG-compatible databases, but the syntax is not supported during upgrade observation.
- In terms of syntax, **update_expr** supports three keywords: CURRENT_TIMESTAMP, LOCALTIMESTAMP, and NOW(). You can also optionally specify the precision of a keyword with parentheses. The three types of keywords are synonyms of each other and have the same attribute effect. For example, ON UPDATE CURRENT_TIMESTAMP(), ON UPDATE CURRENT_TIMESTAMP(5), ON UPDATE LOCALTIMESTAMP(), and ON UPDATE LOCALTIMESTAMP(6). If a keyword does not contain parentheses or contains empty parentheses, the precision is 0. The NOW keyword cannot work without parentheses. In PG-compatible mode, ON UPDATE CURRENT_TIMESTAMP(), ON UPDATE LOCALTIMESTAMP(), and ON UPDATE NOW(5) are not supported.
- This attribute can be specified only for columns of the following types: timestamp, datetime, date, time without time zone, smalldatetime, and abstime.
- The CREATE TABLE AS syntax does not inherit the column attributes.
- The CREATE TABLE LIKE syntax can use **INCLUDING UPDATE** or **EXCLUDING UPDATE** to inherit or exclude a constraint. In PG-compatible mode, only the **EXCLUDING UPDATE** option can be used to exclude a constraint. The **INCLUDING ILM** option is added to copy ILM policy information from the source table. When combined with the **INCLUDING PARTITION** option, it can also copy the policy information of partition objects from the source table.
- The precision specified by this attribute can be different from the precision specified by the type in the corresponding column. After the column value is updated through this attribute, the minimum precision is displayed. For example, **ALTER TABLE t1 ADD col1 timestamp(6) ON UPDATE CURRENT_TIMESTAMP(3);**. If the UPDATE syntax triggers the attribute to take effect, three decimal places in the value of **col1** are displayed after the update.
- A column cannot be designated as both this attribute and a generated column constraint simultaneously.
- Specifying this attribute within the partition key of a partitioned table is not allowed.
- In distributed scenarios, this attribute cannot be specified for distribution keys or the primary key.

- **AUTO_INCREMENT**

  Marks a column as auto-increment.

  For details, see **AUTO_INCREMENT** in **CREATE TABLE**.

- **COLUMN_ENCRYPTION_KEY = column_encryption_key**

  Specifies the name of the column encryption key in the ENCRYPTED WITH constraint.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **ENCRYPTION_TYPE = encryption_type_value**

  When setting the encryption type in the ENCRYPTED WITH constraint, you can choose between **DETERMINISTIC** or **RANDOMIZED** as **encryption_type_value**.

- **COMMENT [=] 'string'**
  - The COMMENT [=] 'string' clause is used to add comments to a table.
  - The **COMMENT 'string'** in **column_constraint** indicates that comments are added to a column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - The **COMMENT 'string'** in **table_constraint** indicates that comments are added to the indexes corresponding to the primary key and unique key.

  For details, see **COMMENT [=] 'string'** in **CREATE TABLE**.

- **UNIQUE [KEY] index_parameters**

  Specifies that a column or a combination of columns must be unique across the entire table.

  For the purpose of a UNIQUE constraint, NULL is not considered equal.

  Semantically equivalent to UNIQUE, UNIQUE KEY is available only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).

- **UNIQUE [index_name][USING method] ({ {column_name [(length)] | (expression) } [ASC | DESC] }[, ...]) index_parameters**

  Specifies that a column or a combination of columns must be unique across the entire table.

  column_name (length) serves as a prefix key. For details, see **column_name (length)** in **CREATE INDEX**.

  For the purpose of a UNIQUE constraint, NULL is not considered equal.

  NOTICE

  - **index_name** is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).
  - If both **constraint_name** and **index_name** are specified for a UNIQUE constraint, the value of **index_name** will be used as the index name.

- **PRIMARY KEY index_parameters**

  PRIMARY KEY [USING method] ({ column_name [ASC | DESC] } [, ...]) index_parameters

  Specifies that one or more columns in the table require unique, non-NULL values.

- **USING method**

  Specifies the method used to create the index.

  For details about the value range, see **Parameters** in **CREATE INDEX**.

  NOTICE

  - **USING method** is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).
  - In MySQL-compatible mode, when **USING method** is not specified, Astore will default to B-tree indexing, and Ustore will default to UB-tree indexing.
  - Since the underlying layer of GSIs is based on UB-tree storage, GSIs will be built with UB-tree storage, even if users specify B-tree storage.
  - If the storage engine of a table is Ustore and the USING BTREE constraint is specified in an SQL statement, the underlying layer will automatically treat it as USING UBTREE.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **ASC | DESC**

  ASC specifies an ascending (default) sort order. **DESC** specifies a descending sort order.

  NOTICE

  ASC | DESC is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).

- **expression**

  Specifies an expression based on one or more columns of the table. It must be written in parentheses.

  NOTICE

  Expression indexes in the UNIQUE constraint are supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).

- **DEFERRABLE | NOT DEFERRABLE**

  Specifies whether the constraint can be deferred. A constraint that is not deferrable will be checked immediately after every command. Checking of constraints that are deferrable can be postponed until the end of the transaction (using the **SET CONSTRAINTS** command). **NOT DEFERRABLE** is the default. Currently, only the UNIQUE, PRIMARY KEY, and FOREIGN KEY constraints accept this clause. (That is, other constraints are not deferrable.)

- **INITIALLY IMMEDIATE | INITIALLY DEFERRED**

  Specifies the default time to check the constraint if it is deferrable. A constraint that is not deferrable is always **INITIALLY IMMEDIATE** and cannot be declared as **INITIALLY DEFERRED**.

  - If the constraint is **INITIALLY IMMEDIATE**, it is checked after each statement. This is the default.
  - If a constraint is **INITIALLY DEFERRED**, it is checked only at the end of the transaction.

- **WITH ({storage_parameter = value} [, ...])**

  Specifies an optional storage parameter for the table or an index.

- **tablespace_name**

  Specifies the name of the tablespace where the index locates.

- **NOCOMPRESS**

  Specifies that the existing compression feature of the table will not be modified.

- **new_table_name**

  Specifies a new name for the table.

- **new_column_name**

  Specifies a new name for a specific column in the table.

- **new_constraint_name**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the new name of a table constraint.

- **new_schema**

  Specifies a new schema name.

- **CASCADE**

  Automatically drops objects that depend on the dropped column or constraint (for example, views referencing the column).

- **RESTRICT**

  Refuses to drop the column if the column is referenced by other columns or constraints. **RESTRICT** is the default option. If **CASCADE** is not specified, the value is **RESTRICT**. An example of the statement is as follows:

  ```
  ALTER TABLE table_name [DROP [column] col_name [CASCADE | RESTRICT]];
  ```
  table_name indicates the table name, and **col_name** indicates the column name.

- **schema_name**

  Specifies the schema name of the table.

- **IF NOT EXISTS**

  When IF NOT EXISTS is specified and columns with the same name exist, a notice is returned, indicating that the column already exists. When IF NOT EXISTS is not specified and columns with the same name exist, an error is returned.

- **[DEFAULT] CHARACTER SET | CHARSET [=] default_charset**

  Changes the default character set of the table. If this parameter is specified separately, the table will use the default collation of the specified character set.

- **[DEFAULT] COLLATE [=] default_collation**

  Changes the default collation of the table. If this parameter is specified separately, the table will default to the character set corresponding to the specified collation.

  This syntax is supported only when **sql_compatibility** is set to **'MYSQL'**. For details about the collation, see **Table 1-398**.

  NOTE

  If the character set or collation of a table is not explicitly specified but the default character set or collation of the schema is specified, the table will inherit the character set and collation from the schema. If the default character set or collation of the schema cannot be found, the table will inherit the character set and its corresponding collation from the current database when **b_format_behavior_compat_options** includes **'default_collation'**.

- **GSIWAITALL**

  It is primarily called by the internal CREATE GLOBAL INDEX CONCURRENTLY/ REINDEX INDEX CONCURRENTLY to synchronize lock waits during online creation of GSIs. Based on the given lock level, you can obtain the list of transactions that conflict with that lock level on each node and wait for those transactions to complete.

Examples of Modifying a Table

- Rename a table.

  ```
  gaussdb=# CREATE TABLE aa(c1 int, c2 int);
  gaussdb=# ALTER TABLE IF EXISTS aa RENAME TO test_alt1;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Change the schema of a table.

```
-- Create the test_schema schema.
gaussdb=# CREATE SCHEMA test_schema;

-- Change the schema of the test_alt1 table to test_schema.
gaussdb=# ALTER TABLE test_alt1 SET SCHEMA test_schema;

-- Query table information.
gaussdb=# SELECT schemaname,tablename FROM pg_tables WHERE tablename = 'test_alt1';
 schemaname  | tablename
-------------+-----------
 test_schema | test_alt1
(1 row)
```

- Change the owner of a table.

```
-- Create user test_user.
gaussdb=# CREATE USER test_user PASSWORD '********';

-- Change the owner of the test_alt1 table to test_user.
gaussdb=# ALTER TABLE IF EXISTS test_schema.test_alt1 OWNER TO test_user;

-- Query.
gaussdb=# SELECT tablename, schemaname, tableowner FROM pg_tables WHERE tablename =
'test_alt1';
 tablename | schemaname  | tableowner
-----------+-------------+------------
 test_alt1 | test_schema | test_user
(1 row)
```

- Change the tablespace of a table.

```
-- Create the tbs_data1 tablespace.
gaussdb=# CREATE TABLESPACE tbs_data1 RELATIVE LOCATION 'tablespace1/tbs_data1';

-- Change the tablespace of the test_alt1 table to tbs_data1.
gaussdb=# ALTER TABLE test_schema.test_alt1 SET TABLESPACE tbs_data1;

-- Query.
gaussdb=# SELECT tablename, tablespace FROM pg_tables WHERE tablename = 'test_alt1';
 tablename | tablespace
-----------+------------
 test_alt1 | tbs_data1
(1 row)

-- Create a table.
gaussdb=# CREATE TABLE test(c1 int, c2 int);
-- Change the space of the test table to tbs_data1 online.
gaussdb=# ALTER TABLE ONLINE test SET TABLESPACE tbs_data1;

-- Query.
gaussdb=# SELECT tablename, tablespace FROM pg_tables WHERE tablename = 'test';
 tablename | tablespace
-----------+------------
 test| tbs_data1
(1 row)

-- Drop.
gaussdb=# DROP TABLE test ;
gaussdb=# DROP TABLE test_schema.test_alt1;
gaussdb=# DROP TABLESPACE tbs_data1;
gaussdb=# DROP SCHEMA test_schema;
gaussdb=# DROP USER test_user;
```

Examples of Modifying Columns

- Change column names.

```
-- Create a table.
gaussdb=# CREATE TABLE test_alt2(c1 INT,c2 INT);

-- Change column names.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# ALTER TABLE test_alt2 RENAME c1 TO id;
gaussdb=# ALTER TABLE test_alt2 RENAME COLUMN c2 to areaid;

-- Query.
gaussdb=# \d test_alt2
   Table "public.test_alt2"
 Column |  Type   | Modifiers
--------+---------+-----------
 id     | integer |
 areaid | integer |
```

- Add a column.

```
-- Add a column to the test_alt2 table.
gaussdb=# ALTER TABLE IF EXISTS test_alt2 ADD COLUMN name VARCHAR(20);

-- Query.
gaussdb=# \d test_alt2
           Table "public.test_alt2"
  Column  |         Type          | Modifiers
----------+-----------------------+-----------
 id       | integer               |
 areaid   | integer               |
 name     | character varying(20) |
```

- Add an auto-increment column.

```
-- Add the AUTO_INCREMENT column to the local_autoinc table. (Change the DN name by running SELECT node_name FROM pgxc_node WHERE node_type = 'D' based on the actual situation.)
gaussdb=# CREATE TABLE local_autoinc(col1 int)
DISTRIBUTE BY LIST(col1)(
    SLICE s1 VALUES (1) DATANODE datanode1,
    SLICE s2 VALUES (2) DATANODE datanode2
);
-- Distribute data to DN1.
gaussdb=# INSERT INTO local_autoinc(col1) VALUES(1);

-- Add a local auto-increment column. The value starts from 1 for each DN.
gaussdb=# ALTER TABLE local_autoinc ADD COLUMN col int AUTO_INCREMENT;
gaussdb=# SELECT col,col1 FROM local_autoinc ORDER BY 2,1;
 col | col1
-----+------
   1 |    1
(1 row)

-- Set the next auto-increment value of all DNs to 10.
gaussdb=# ALTER TABLE local_autoinc AUTO_INCREMENT = 10;

-- Distribute data to DN1. NULL triggers auto-increment, and the auto-increment value is 10.
gaussdb=# INSERT INTO local_autoinc(col, col1) VALUES(NULL,1);

-- Distribute data to DN2. 0 triggers auto-increment, and the auto-increment value is 10.
gaussdb=# INSERT INTO local_autoinc(col, col1) VALUES(0,2);
gaussdb=# SELECT col,col1 FROM local_autoinc ORDER BY 2,1;
 col | col1
-----+------
   1 |    1
  10 |    1
  10 |    2
(3 rows)
```

- Change the data type of a column.

```
-- Change the type of the name column in the test_alt2 table.
gaussdb=# ALTER TABLE test_alt2 MODIFY name VARCHAR(50);



-- Query.
gaussdb=# \d test_alt2
          Table "public.test_alt2"
 Column |         Type          | Modifiers
--------+-----------------------+-----------
 id     | integer               |
 areaid | integer               |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 name   | character varying(50) |

-- Change the type of the name column in the test_alt2 table online.
gaussdb=# ALTER TABLE ONLINE test_alt2 MODIFY name VARCHAR(60);


-- Query.
gaussdb=# \d test_alt2
          Table "public.test_alt2"
 Column |         Type          | Modifiers
--------+-----------------------+-----------
 id     | integer               |
 areaid | integer               |
 name   | character varying(60) |
-- Change the type of the name column in the test_alt2 table.
gaussdb=# ALTER TABLE test_alt2 ALTER COLUMN name TYPE VARCHAR(25);

-- Query.
gaussdb=# \d test_alt2
          Table "public.test_alt2"
 Column |         Type          | Modifiers
--------+-----------------------+-----------
 id     | integer               |
 areaid | integer               |
 name   | character varying(25) |

-- Change the type of the name column in the test_alt2 table online.
gaussdb=# ALTER TABLE ONLINE test_alt2 ALTER COLUMN name TYPE VARCHAR(35);

-- Query.
gaussdb=# \d test_alt2
          Table "public.test_alt2"
 Column |         Type          | Modifiers
--------+-----------------------+-----------
 id     | integer               |
 areaid | integer               |
 name   | character varying(35) |
```

- Drop a column.

```
-- Drop the areaid column from test_alt2.
gaussdb=# ALTER TABLE test_alt2 DROP COLUMN areaid;

-- Query.
gaussdb=# \d test_alt2
          Table "public.test_alt2"
 Column |         Type          | Modifiers
--------+-----------------------+-----------
 id     | integer               |
 name   | character varying(25) |
```

- Change the column-store mode.

```
-- View table details.
gaussdb=# \d+ test_alt2
                          Table "public.test_alt2"
 Column |         Type          | Modifiers | Storage  | Stats target | Description
--------+-----------------------+-----------+----------+--------------+-------------
 id     | integer               |           | plain    |              |
 name   | character varying(35) |           | extended |              |
Has OIDs: no
Distribute By: HASH(id)
Location Nodes: ALL DATANODES
Options: orientation=row, compression=no, storage_type=USTORE, segment=off

-- Change the storage mode of the name column in the test_alt2 table.
gaussdb=# ALTER TABLE test_alt2 ALTER COLUMN name SET STORAGE PLAIN;

-- Query.
gaussdb=# \d+ test_alt2
                         Table "public.test_alt2"
 Column |         Type          | Modifiers | Storage | Stats target | Description
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------+-----------------------+-----------+---------+--------------+-------------
 id     | integer               |           | plain   |              |
 name   | character varying(35) |           | plain   |              |
Has OIDs: no
Distribute By: HASH(id)
Location Nodes: ALL DATANODES
Options: orientation=row, compression=no, storage_type=USTORE, segment=off


-- Drop.
gaussdb=# DROP TABLE test_alt2;
```

Examples of Modifying Constraints

- Add a NOT NULL constraint to a column.

```
-- Create a table.
gaussdb=# CREATE TABLE test_alt3(pid INT, areaid CHAR(5), name VARCHAR(20));

Add a NOT NULL constraint to pid.
gaussdb=# ALTER TABLE test_alt3 MODIFY pid NOT NULL;

-- Query.
gaussdb=# \d test_alt3
          Table "public.test_alt3"
 Column |         Type          | Modifiers
--------+-----------------------+-----------
 pid    | integer               | not null
 areaid | character(5)          |
 name   | character varying(20) |

-- Cancel the NOT NULL constraint on a column.
gaussdb=# ALTER TABLE test_alt3 MODIFY pid NULL;

-- Add a NOT NULL constraint to the PID online.
gaussdb=# ALTER TABLE ONLINE test_alt3 MODIFY pid NOT NULL;

-- Query.
gaussdb=# \d test_alt3
          Table "public.test_alt3"
 Column |         Type          | Modifiers
--------+-----------------------+-----------
 pid    | integer               | not null
 areaid | character(5)          |
 name   | character varying(20) |
```

- Cancel the NOT NULL constraint on a column.

```
gaussdb=# ALTER TABLE test_alt3 MODIFY pid NULL;
-- Query.
gaussdb=# \d test_alt3
          Table "public.test_alt3"
 Column |         Type          | Modifiers
--------+-----------------------+-----------
 pid    | integer               |
 areaid | character(5)          |
 name   | character varying(20) |
```

- Change the default value of a column.

```
-- Change the default value of id in the test_alt3 table.
gaussdb=# ALTER TABLE test_alt3 ALTER COLUMN areaid SET DEFAULT '00000';

-- Query.
gaussdb=# \d test_alt3
                Table "public.test_alt3"
 Column |         Type          |        Modifiers
--------+-----------------------+-------------------------
 pid    | integer               |
 areaid | character(5)          | default '00000'::bpchar
 name   | character varying(20) |
-- Drop the default value of id.
gaussdb=# ALTER TABLE test_alt3 ALTER COLUMN areaid DROP DEFAULT;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Query.
gaussdb=# \d test_alt3
          Table "public.test_alt3"
 Column |          Type          | Modifiers
--------+------------------------+-----------
 pid    | integer                |
 areaid | character(5)           |
 name   | character varying(20) |
```

- Add a table-level constraint.
  - Directly add a constraint.

    ```
    -- Add a PRIMARY KEY constraint to a table.
    gaussdb=# ALTER TABLE test_alt3 ADD CONSTRAINT pk_test3_pid PRIMARY KEY (pid);

    -- Query.
    gaussdb=# \d test_alt3
              Table "public.test_alt3"
     Column |          Type          | Modifiers
    --------+------------------------+-----------
     pid    | integer                | not null
     areaid | character(5)           |
     name   | character varying(20) |
    Indexes:
        "pk_test3_pid" PRIMARY KEY, ubtree (pid) WITH (storage_type=USTORE) TABLESPACE
    pg_default

    -- Drop the table.
    gaussdb=# DROP TABLE IF EXISTS test_alt3;

    -- Create a table.
    gaussdb=# CREATE TABLE test_alt3(pid INT, areaid CHAR(5), name VARCHAR(20));

    -- Add a primary key constraint to the table online.
    gaussdb=# ALTER TABLE ONLINE test_alt3 ADD CONSTRAINT pk_test3_pid PRIMARY KEY (pid);

    -- Query.
    gaussdb=# \d test_alt3
              Table "public.test_alt3"
     Column |          Type          | Modifiers
    --------+------------------------+-----------
     pid    | integer                | not null
     areaid | character(5)           |
     name   | character varying(20) |
    Indexes:
        "pk_test3_pid" PRIMARY KEY, ubtree (pid) WITH (storage_type=USTORE) TABLESPACE
    pg_default
    ```

  - Create an index and then add constraints.

    ```
    -- Create a table.
    gaussdb=# CREATE TABLE test_alt4(c1 INT, c2 INT);

    -- Create an index.
    gaussdb=# CREATE UNIQUE INDEX pk_test4_c1 ON test_alt4(c1);

    -- Associate the created index when adding a constraint.
    gaussdb=# ALTER TABLE test_alt4 ADD CONSTRAINT pk_test4_c1 PRIMARY KEY USING INDEX
    pk_test4_c1;

    -- Query.
    gaussdb=# \d test_alt4
       Table "public.test_alt4"
     Column |  Type   | Modifiers
    --------+---------+-----------
     c1     | integer | not null
     c2     | integer |
    Indexes:
        "pk_test4_c1" PRIMARY KEY, ubtree (c1) WITH (storage_type=USTORE) TABLESPACE
    pg_default
    ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop.
gaussdb=# DROP TABLE test_alt4;
```

- Drop a table-level constraint.

```
-- Drop a constraint.
gaussdb=# ALTER TABLE test_alt3 DROP CONSTRAINT IF EXISTS pk_test3_pid;

-- Query.
gaussdb=# \d test_alt3
          Table "public.test_alt3"
 Column |         Type          | Modifiers
--------+-----------------------+-----------
 pid    | integer               | not null
 areaid | character(5)          |
 name   | character varying(20) |

-- Drop.
gaussdb=# DROP TABLE test_alt3;

-- Delete a primary key constraint. (This syntax is supported only in MySQL-compatible and ORA-compatible modes.)
gaussdb=# CREATE TABLE test_drop_primary_key(c1 INT PRIMARY KEY);
gaussdb=# \d test_drop_primary_key
Table "public.test_drop_primary_key"
 Column |  Type   | Modifiers
--------+---------+-----------
 c1     | integer | not null
Indexes:
    "test_drop_primary_key_pkey" PRIMARY KEY, ubtree (c1) WITH (storage_type=USTORE) TABLESPACE pg_default


gaussdb=# ALTER TABLE test_drop_primary_key DROP PRIMARY KEY;
gaussdb=# \d test_drop_primary_key
Table "public.test_drop_primary_key"
 Column |  Type   | Modifiers
--------+---------+-----------
 c1     | integer | not null
```

- Rename a constraint.

```
-- Create the tbl_test1 table and specify the primary key constraint.
gaussdb=# CREATE TABLE tbl_test1(
    id int,
    name varchar(50),
    CONSTRAINT aaa PRIMARY KEY (id)
);

-- Rename the constraint.
gaussdb=# ALTER TABLE tbl_test1 RENAME CONSTRAINT aaa TO pk_test1_id;

-- View the information.
gaussdb=# \d tbl_test1;
          Table "public.tbl_test1"
 Column |         Type          | Modifiers
--------+-----------------------+-----------
 id     | integer               | not null
 name   | character varying(50) |
Indexes:
    "pk_test1_id" PRIMARY KEY, ubtree (id) WITH (storage_type=USTORE) TABLESPACE pg_default


-- Drop the table.
gaussdb=# DROP TABLE tbl_test1;
```

Helpful Links

CREATE TABLE and **DROP TABLE**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.7.35 ALTER TABLE PARTITION

Description

Modifies table partitions, including adding, deleting, splitting, merging, clearing, exchanging, and renaming partitions, moving partition tablespaces, and modifying partition attributes.

Precautions

- Only the partitioned table owner or a user with the ALTER permission can run the **ALTER TABLE PARTITION** command. When separation of duties is disabled, system administrators have this permission by default.
- The tablespace of the added partition cannot be PG_GLOBAL.
- The name of any added partition must differ from the names of existing partitions in the partitioned table.
- The partition key value of any added partition must match the partition key type of the partitioned table.
- When a range partition is added, its partition key value must be greater than the upper limit of the last range partition in the partitioned table.
- When a list partition is added, its partition key value must not overlap with existing partition key values.
- Hash partitions cannot be added.
- When the number of partitions in the target partitioned table reaches the maximum limit (1,048,575), no additional partitions can be added.
- If a partitioned table contains only one partition, that partition cannot be dropped.
- When you use PARTITION FOR() to select partitions, the number of values specified within the parentheses must match the number of columns specified in the partition definition, and they must correspond in a one-to-one manner.
- The **Value** partitioned table does not support the ALTER PARTITION operation.
- Hash partitioned tables do not support splitting, merging, adding, or dropping partitions.
- Deleting, splitting, merging, clearing, or exchanging partitions will make global indexes unusable. You can declare the UPDATE GLOBAL INDEX clause to update the indexes synchronously.
- If partitions are dropped, split, merged, truncated, or exchanged without declaring the UPDATE GLOBAL INDEX clause, concurrent DML statements may trigger errors due to unusable indexes.
- Deleting, splitting, merging, clearing, or exchanging partitions invalidates GSIs on partitioned tables. For exchanging partitions, all GSIs on ordinary tables become invalid. You can declare the UPDATE DISTRIBUTED GLOBAL INDEX clause to update indexes synchronously.
- If the UPDATE DISTRIBUTED GLOBAL INDEX clause is not used when you delete, split, merge, clear, or exchange partitions, concurrent DML services may report errors due to invalidated indexes.
- If **enable_gpi_auto_update** is set to **on**, the global index is automatically updated even if the UPDATE GLOBAL INDEX clause is not declared.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If the **enable_drop_truncate_partition_keep_gpi** parameter is set to **on**, the global index remains valid when partitions are dropped or truncated. In this case, the global index is always valid, regardless of whether the DROP/TRUNCATE PARTITION statement contains the UPDATE GLOBAL INDEX clause. This priority is higher than that of the **enable_gpi_auto_update** parameter and the UPDATE GLOBAL INDEX clause explicitly specified by the user.
- If you add a row-level expression when adding or changing an ILM policy for a data object, note that the row-level expression supports only the functions listed in the whitelist. For details about the whitelist function list, see **Row Expression Function Whitelist**.

  NOTE

  Among the functions supported by the row-level expression ON (EXPR) in the ILM policy, the output of some may be influenced by compatibility parameters. For instance, after you set **b_format_version** to **'5.7'** and **b_format_dev_version** to **'s2'** in MySQL-compatible mode, the upper function will be unable to convert data to uppercase as usual.
- When executing online DDL operations for splitting or merging partitions, exchanging partitions, or moving partition tablespaces, reserve sufficient disk space, at least equal to the total space of the tables or partitions (and their accessories such as indexes).
- Online DDL operations cannot be executed within a transaction or a stored procedure. If you use the JDBC driver to run multiple SQL statements, a transaction is implicitly started, and these SQL statements cannot be executed together.
- If online DDL execution is canceled, the residual data clearance process starts first. If the execution is canceled again, the clearance process ends and residual data remains.
- Resource contention exists between online DDL operations and user services. Therefore, online DDL operations are not suitable for heavy-traffic scenarios. In scenarios where resources are limited, fine-grained resource control is recommended. Parallel acceleration is not recommended.
- Online DDL operations are not supported in M-compatible databases.
- In the following cases, online DDL operations do not take effect. By default, DDL operations are performed offline. If **ONLINE** is specified, a message is displayed, indicating that it is ignored.
  - The statement for altering table partitions online contains operations that are not supported by the current online DDL feature. (Splitting/merging partitions, exchanging partitions, and moving partition tablespaces are supported.)
  - A table on which online modifications to its partitions (including splitting/merging partitions, exchanging partitions, and moving partition tablespaces) are performed contains GSIs and requires synchronous updates to the GSIs.
  - The statement for modifying table partitions online contains operations of multiple clauses (for example, the statement involves both splitting and exchanging partitions online).
- Online DDL operations (including splitting/merging partitions, exchanging partitions, and moving partition tablespaces) will not take effect on database-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

level, index-level, subpartitioned, segment-page, hash bucket, temporary, or unlogged tables. In this case, DDL operations are executed offline by default, and a message is displayed indicating that the current DDL statement cannot be executed online.

- Long transactions may block online DDL statement execution results, including splitting/merging partitions, exchanging partitions, and moving partition tablespaces. You are advised to avoid performing online DDL operations when long transactions exist.
- Performing online DDL operations to modify table partitions (including splitting/merging partitions, exchanging partitions, and moving partition tablespaces) will block DQL and DML operations at a later stage. Therefore, requests may fail due to blocking in service models with a timeout error reporting mechanism. Before using this feature, carefully evaluate the service model's tolerance for waiting locks to confirm whether this feature can be used.
- During the execution of online DDL statements for modifying table partitions, avoid executing other DDL statements concurrently. If exceptions like no response or error reporting occur, reconnect to the client and try again.
- During the upgrade from a version that does not support the online DDL operations for altering table partitions to a version that supports it, the operations (including splitting/merging partitions, exchanging partitions, and moving partition tablespaces) are not supported because they are under observation. By default, DDL operations are performed offline, and a message is displayed indicating that online DDL operations are not supported during the upgrade.
- During the execution of an online DDL operation for altering table partitions (including splitting/merging partitions, exchanging partitions, and moving partition tablespaces), a specific schema is generated. The schema naming format is online$$ddl$$[hash value of (*Name of the schema where the table is located Table name*)].
  - By default, the owner of the schema is the initial user. You are advised not to operate the schema during online DDL operations; otherwise, the online DDL operations may be terminated and an unknown error may occur.
  - Do not create a schema with the prefix online$$. Otherwise, an error message is displayed. If a schema with the same name already exists, online DDL operations will not take effect. By default, DDL operations are executed offline, and a message is displayed indicating that the current DDL statement cannot be executed online due to a schema name conflict.
- If an online DDL operation (including splitting or merging partitions or moving partition tablespaces) fails, the table's storage space may bloat significantly. Exchanging partitions online may cause significant storage space bloat on the table regardless of its success or failure. The bloat degree is proportional to the DDL statement execution duration and the number of concurrent write operations. You are advised not to perform online DDL operations when long transactions exist.
- Online DDL operations (including splitting/merging partitions, exchanging partitions, and moving partition tablespaces) inherit the restrictions of **CREATE TABLE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- During the online DDL operation of merging partitions, if the data of the partitions to be merged has been compressed before the operation, the data may be decompressed after the operation.
- During online DDL operations (including splitting/merging partitions, exchanging partitions, and moving partition tablespaces), if global indexes are automatically updated, concurrent queries and DML services may be blocked for a long time at a later stage of the operations.

Syntax

Modifying a partition in a partitioned table includes modifying the table partition itself and the table partition name, and resetting the partition ID.

- Main syntax for modifying a partition:

```
ALTER TABLE [ OFFLINE | [ ONLINE [ WITH ( { online_parameter = value } [, ... ] ) ] ] ] [ IF EXISTS ]
{ table_name  [*] | ONLY table_name | ONLY ( table_name  )}
    action [, ... ];
```

  action includes the following partition maintenance clauses. For the partition continuity when multiple clauses are used for partition maintenance, GaussDB executes DROP PARTITION and ADD PARTITION first, and then the rest clauses in sequence.

```
{ move_clause  |
    exchange_clause  |
    row_clause  |
    merge_clause  |
    modify_clause  |
    split_clause  |
    add_clause  |
    drop_clause  |
    truncate_clause  |
    ilm_clause |
    set_partitioning_clause }
```

  - move_clause: moves the partition to a different tablespace.

```
MOVE PARTITION { partition_name | FOR ( partition_value [, ...] ) } TABLESPACE tablespacename
```

  - exchange_clause: moves data from an ordinary table to a specified partition.

```
EXCHANGE PARTITION { ( partition_name ) | partition_name | FOR ( partition_value [, ...] ) }
    WITH TABLE {[ ONLY ] ordinary_table_name | ordinary_table_name * | ONLY
( ordinary_table_name )}
    [ { WITH | WITHOUT } VALIDATION ] [ VERBOSE ] [ UPDATE GLOBAL INDEX ] [ UPDATE
DISTRIBUTED GLOBAL INDEX | NO UPDATE DISTRIBUTED GLOBAL INDEX ]
```

    The ordinary table and partition involved in the exchange must meet the following requirements:

    - The number of columns in the ordinary table must match that in the partition, with consistent column details including names, data types, constraints, collations, storage parameters, and compression information.
    - The compression information must be consistent between the ordinary table and partition.
    - The distribution key information of the ordinary table and partition should be consistent.
    - The number of indexes in the ordinary table must match the number of local indexes in the partition, with consistent index information.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The number of table constraints must be consistent between the ordinary table and partition, with consistent constraint information.
- An ordinary table and a partitioned table cannot be associated with foreign key constraints of other tables.
- An ordinary table cannot be a temporary table. A partitioned table can only be a range partitioned table, list partitioned table, hash partitioned table, or interval partitioned table.
- When the built-in security policy is enabled, ordinary and partitioned tables cannot have dynamic data masking or row-level security constraints.

NOTICE

- After the exchange is completed, the data of the ordinary table and partition are swapped, along with their tablespace information. At this point, statistics on the ordinary table and partition become unreliable, necessitating an ANALYZE operation.
- Non-partition keys cannot be used to create local unique indexes; only global unique indexes are permitted. Therefore, if an ordinary table contains a unique index, its data may fail to be exchanged.

  To exchange data, you can create an intermediate table, insert partition data into the intermediate table, truncate partitions, insert ordinary table data into the partitioned table, drop the ordinary table, and rename the intermediate table.
- In a scenario where both an ordinary table and a partitioned table are Ustore tables, if the UB-tree index type (RCR or PCR, RCR by default) of the ordinary table is different from the local UB-tree index type (RCR or PCR, RCR by default) of the partitioned table, data cannot be exchanged.
- If the DROP COLUMN operation is performed on the ordinary or partitioned table, the dropped column still exists physically. Therefore, you need to ensure that the dropped column of the ordinary table is strictly aligned with that of the partition. Otherwise, the exchange will fail.
- The EXCHANGE PARTITION { (partition_name) | partition_name | FOR (partition_value [, ...]) } operation is available in MySQL-compatible database (that is, **sql_compatibility** set to **'MYSQL'**). In other modes, only the EXCHANGE PARTITION { (partition_name) | FOR (partition_value [, ...]) } operation is available. If **partition_name** is the name of a partition, the partition and ordinary table are exchanged.
- If the ordinary table and partition to be swapped contain the rowid column and are in the compressed state, the data in the ordinary table and partition is automatically decompressed after the partition is swapped.

– The row_clause syntax is used to set row movement of a partitioned table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
{ ENABLE | DISABLE } ROW MOVEMENT
```

- merge_clause: merges multiple partitions into one. Each command can merge a maximum of 300 partitions.

```
MERGE PARTITIONS { partition_name } [, ...] INTO PARTITION partition_name
    [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n
{ day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM |
HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}]
[ TABLESPACE tablespacename ] [ UPDATE GLOBAL INDEX ] [ UPDATE DISTRIBUTED GLOBAL
INDEX | NO UPDATE DISTRIBUTED GLOBAL INDEX ]
```

NOTICE

- For range partitioning and interval partitioning, the ranges of the source partitions must increase continuously, and the partition name after MERGE can be the same as the name of the last source partition. For list partitioning, there is no such range requirement on the source partitions, and the partition name after MERGE can be the same as that of any source partition. If the partition name after MERGE is the same as that of a source partition, they are regarded as the same partition.
- If the GUC parameter **enable_ilm** is not enabled and the merge_clause syntax is used to merge multiple partitions with the ILM policy into one partition, the new partition does not inherit the ILM policy.
- Ustore tables do not support ALTER TABLE MERGE PARTITIONS within transaction blocks or stored procedures.

- The modify_clause syntax is used to set whether a partitioned index is usable.

```
MODIFY PARTITION partition_name { UNUSABLE LOCAL INDEXES | REBUILD UNUSABLE LOCAL
INDEXES | COLVIEW [ PRIORITY { HIGH | LOW | NONE } ] | NOCOLVIEW }
```

- split_clause: splits one partition into multiple.

```
SPLIT PARTITION { partition_name | FOR ( partition_value [, ...] ) } { split_point_clause |
no_split_point_clause } [ UPDATE GLOBAL INDEX ] [ UPDATE DISTRIBUTED GLOBAL INDEX |
NO UPDATE DISTRIBUTED GLOBAL INDEX ]
```

NOTICE

- The partition name after SPLIT can be the same as the source partition name, but they are regarded as different partitions.
- When the GUC parameter **enable_ilm** is disabled, the resulting partitions do not inherit the ILM policy, even if the source partition being split by split_clause has an ILM policy.

  - The split_point_clause syntax for specifying the split point for a range partitioned table is as follows:

```
AT ( partition_value ) INTO ( PARTITION partition_name [ ILM ADD POLICY ROW STORE
COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO
MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour |
day | month | year } OF NO MODIFICATION | NONE}] [ TABLESPACE tablespacename ] ,
PARTITION partition_name [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED
[ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON
( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year }
OF NO MODIFICATION | NONE}] [ TABLESPACE tablespacename ] )
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

The size of the split point must fall within the range of the partition key for the partition to be split. By specifying a split point, you can only split one partition into two new partitions.

- The no_split_point_clause syntax without specifying a split point for a range partitioned table is as follows:

```
INTO { ( partition_less_than_item [, ...] ) | ( partition_start_end_item [, ...] ) }
```

NOTICE

- When no split point is specified, the partition key of the first new partition specified in **partition_less_than_item** must be greater than that of the partition immediately preceding the one being split (if any); the partition key of the last partition specified in **partition_less_than_item** must equal that of the partition being split.
- When no split point is specified, the start point (if present) of the first new partition specified in **partition_start_end_item** must equal that of the partition immediately preceding the one being split (if any); the end point (if present) of the last partition specified in **partition_start_end_item** must equal that of the partition being split.
- **partition_less_than_item** supports a partition key with up to 16 columns, while **partition_start_end_item** supports a one-column partition key. For details about the supported data types, see **PARTITION BY RANGE(partition_key)** in **CREATE TABLE PARTITION**.
- **partition_less_than_item** and **partition_start_end_item** cannot be used in the same statement. This restriction does not apply across different SPLIT statements.

- The partition_less_than_item syntax is as follows (the range of the last partition is not defined, that is, the VALUES LESS THAN (partition_value) part is not defined; by default, the last partition inherits the upper boundary value of the range defined for the source partition):

```
PARTITION partition_name VALUES LESS THAN {( { partition_value | MAXVALUE }  [, ...] ) | MAXVALUE }
   [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [ TABLESPACE tablespacename ]
```

NOTICE

During range partitioning, the MAXVALUE keyword can be used without parentheses. It can be used only in MySQL-compatible mode. It cannot be used in subpartitions or in scenarios where multi-column partition keys are used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The syntax of **partition_start_end_item** is as follows. For details about the restrictions, see **START END syntax rules** in **CREATE TABLE PARTITION**.

```
PARTITION partition_name {
      {START(partition_value) END (partition_value) EVERY (interval_value)} |
      {START(partition_value) END ({partition_value | MAXVALUE})} |
      {START(partition_value)} |
      {END({partition_value | MAXVALUE})}
} [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [TABLESPACE tablespace_name]
```

- The split_point_clause syntax for specifying the split point for a list partitioned table is as follows:

```
VALUES ( partition_value_list ) INTO ( PARTITION partition_name [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [ TABLESPACE tablespacename ] , PARTITION partition_name [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [ TABLESPACE tablespacename ] )
```

NOTICE

The split point must be a non-empty true subset of the source partition. By specifying a split point, you can only split one partition into two new partitions.

- The no_split_point_clause syntax for not specifying the split point for a list partitioned table is as follows:

```
INTO ( PARTITION partition_name VALUES (partition_value_list) [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [ TABLESPACE tablespacename ][, ...] )
```

NOTICE

- Do not define the range of the last partition, that is, do not define the VALUES (partition_value_list) part; the partition range is equal to the remaining set of the source partition excluding other subpartitions.
- If no split point is specified, each new partition must be a non-empty true subset of the source partition and does not overlap with each other.

– add_clause: adds one or more partitions to a partitioned table.

```
ADD {{partition_less_than_item | partition_start_end_item} |
    PARTITION({partition_less_than_item | partition_start_end_item | partition_list_item})}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- The PARTITION({partition_less_than_item | partition_start_end_item | partition_list_item}) syntax can be used only in MySQL-compatible mode.
- The ALTER TABLE table_name ADD PARTITION (partition_definition1, partition_definition2,…); syntax cannot be used to add multiple partitions. Only the original syntax for adding multiple partitions is supported: ALTER TABLE table_name ADD PARTITION (partition_definition1), ADD PARTITION (partition_definition2), …

Syntax for **partition_list_item**:

```
PARTITION partition_name VALUES [IN] (list_values_clause)
   [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n
{ day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM |
HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}]
[ TABLESPACE tablespacename ]
```

NOTICE

- **partition_list_item** supports a maximum of 16 partition keys. For details about the supported data types, see **PARTITION BY LIST [COLUMNS] (partition_key)** in **CREATE TABLE PARTITION**.
- Partitions cannot be added to a hash partitioned table.
- IN needs to be used in MySQL-compatible mode and cannot be used in subpartitions.

– drop_clause: drops a partition from a partitioned table.

```
DROP PARTITION  { partition_name | FOR (  partition_value [, ...] )  } [ UPDATE GLOBAL
INDEX ] [ UPDATE DISTRIBUTED GLOBAL INDEX | NO UPDATE DISTRIBUTED GLOBAL INDEX ]
```

NOTICE

- Partitions cannot be dropped from hash partitioned tables.
- If a partitioned table contains only one partition, that partition cannot be dropped.

– truncate_clause: truncates a partition in a partitioned table.

```
TRUNCATE PARTITION  { partition_name | FOR (  partition_value [, ...] )  } [ UPDATE GLOBAL
INDEX ] [ UPDATE DISTRIBUTED GLOBAL INDEX | NO UPDATE DISTRIBUTED GLOBAL INDEX ]
```

– The ilm_clause syntax adds an ILM policy to a partition. It supports the syntax for the OLTP table compression feature of data lifecycle management.

```
MODIFY PARTITION partition_name ILM ADD POLICY ROW STORE COMPRESS ADVANCED
[ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR ) ]
```

– The set_partitioning_clause syntax is used to set storage parameters for partitions. It is called in online DDL operations. You are advised not to use it.

```
SET ({storage_parameter = value} )
```

- Alter the name of a partition.

```
ALTER TABLE [ IF EXISTS ] { table_name [*] | ONLY table_name | ONLY ( table_name  )}
   RENAME PARTITION { partition_name | FOR ( partition_value [, ...] ) } TO partition_new_name;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Reset a partition ID.

```
ALTER TABLE [ IF EXISTS ] { table_name [*] | ONLY table_name | ONLY ( table_name  )} RESET PARTITION;
```

Parameters

- **ONLINE | OFFLINE**

  Controls whether to enable online DDL operations.

  The keywords ONLINE and OFFLINE are used.

  – ONLINE: DDL operations are executed online. In this case, if the ALTER TABLE statement contains a scenario that is not supported by the current DDL feature, the DDL operations are executed offline and a notice is displayed, indicating that the ONLINE keyword is ignored.

  – OFFLINE: DDL operations are executed offline.

  NOTE

  - During the online DDL operation of exchanging partitions, the system checks if the data falls within the range of the corresponding partitions. If the check result indicates that it falls outside the specified range, an error occurs during the DDL operation. Online DDL operations and DML operations can be performed concurrently. However, concurrent DML operations may delete data outside the corresponding partition range, whereas an error may be still reported for the DDL operations.

    For example, during the online DDL transaction of scanning a table, if the online DDL transaction detects that tuple A in the table is not within the range of the corresponding partitions, an error occurs immediately. If a parallel transaction deletes tuple A after an online DDL operation scans the tuple, tuple A is not considered as deleted.
  - Online DDL error troubleshooting: The system automatically creates a temporary schema as the DDL implementation tool and sets the table to the **append_mode** state during online DDL operations (including changing the column type, precision, and length, and adding NOT NULL, CHECK, PRIMARY KEY, and UNIQUE constraints). If the online DDL execution is complete, the temporary mode is automatically cleared and restored. Automatic cleanup may fail due to reasons such as lock timeout. In this case, manual cleanup is required. If a severe fault occurs, such as FATAL, PANIC, database fault, or node disconnection, the created temporary schema and table appending status may remain. In this case, you need to manually clear and restore the temporary schema and table. The scenarios are as follows:
    - Canceling online DDL operations: If you cancel online DDL operations during the execution, the system automatically clears the data. If the execution is canceled, the residual data clearance process starts first. If the execution is canceled again, the clearance process ends and residual data remains.
    - Operation error: If an error needs to be reported during the execution of a DDL command, for example, type conversion fails or length modification damages data, the system reports an error and automatically clears the data.
    - Major fault: After the cluster becomes normal, you can manually clear the data using **online_ddl_cleanup()**, or the data will be automatically cleared when you run online DDL operations or online VACUUM FULL next time.
  - You are advised not to call **online_ddl_cleanup()** during online DDL execution. Otherwise, the online DDL operations may be terminated or an unknown error may occur.

- **online_parameter**

  Specifies the online DDL option when the **ONLINE** keyword is specified. If the **ONLINE** keyword is not specified or online DDL conditions are not met, this parameter does not take effect.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **parallel_threads**: specifies the number of threads when parallelism is enabled. Value range: an integer ranging from 1 to 32. The default value is **1**.
  - When **parallel_threads** is set to a value greater than 1, multiple threads are enabled and the number of threads is specified.
  - If **parallel_threads** is not set or is set to **1**, multi-thread acceleration is disabled.

    When a user enables resource and workload management, for example, I/O resource management (for example, **io_limits** > **0** or **io_priority** = **high**/**medium**/**low**), **parallel_threads** ignores resource and workload management under the user or session. In addition, a notice is displayed, indicating that the current resource and workload management is invalid and multi-thread acceleration is enabled. The multi-thread execution process of online DDL operations actually follows the resource and workload management of the initial user.
  - When **parallel_workers** and **parallel_threads** are used together for acceleration, **parallel_workers** controls the maximum number of parallel threads in an index, and **parallel_threads** controls the maximum number of parallel threads between indexes.
- **max_catchup_times**: specifies the maximum number of online DDL catchups to prevent long online DDL execution time. Value range: an integer ranging from 1 to 50. The default value is **50**.
  - Online DDL execution records the DML incremental data generated during data reconstruction to a temporary table and performs multiple cyclic catchups to reconstruct the incremental data. If the amount of concurrent DML data is too large, online DDL catchup fails. To prevent this problem, you can set this parameter to limit the number of catchups.

- **table_name**

  Specifies the name of a partitioned table.

  Value range: existing partitioned table names.

- **partition_name**

  Specifies the name of a partition.

  Value range: existing partition names.

- **tablespacename**

  Specifies the destination tablespace for partition movement.

  Value range: existing tablespace names.

- **partition_value**

  Specifies partition key values.

  The values specified by PARTITION FOR (partition_value [, ...]) uniquely identify a partition.

  Value range: determined by the value range of the partition key for the involved partition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **UNUSABLE LOCAL INDEXES**

  Sets all the indexes unusable in the partition.

- **REBUILD UNUSABLE LOCAL INDEXES**

  Rebuilds all the indexes in the partition.

- **{ ENABLE | DISABLE } ROW MOVEMENT**

  Specifies whether to enable row movement.

  If the tuple value is updated on the partition key during the UPDATE operation, the partition where the tuple is located is altered. Setting this parameter enables error messages to be reported or movement of the tuple between partitions.

  Value range:

  - **ENABLE**: Row movement is enabled.
  - **DISABLE**: Row movement is disabled.

  By default, this parameter is disabled.

- **ordinary_table_name**

  Specifies the name of the ordinary table involved in data migration.

  Value range: existing ordinary table names.

- **{ WITH | WITHOUT } VALIDATION**

  Checks whether data in the ordinary table falls within the partition key range of the partition involved in data migration.

  Value range:

  - **WITH**: The system checks whether data in the ordinary table falls within the partition key range of the target partition. If any data falls outside this range, an error is reported.
  - **WITHOUT**: The system does not check whether data in the ordinary table falls within the partition key range of the target partition.

  The default value is **WITH**.

  The check is time consuming, especially when the data volume is large. Therefore, if you are sure that all data in the ordinary table falls within the partition key range of the target partition, use **WITHOUT** to skip this check.

- **VERBOSE**

  When **VALIDATION** is **WITH**, if the ordinary table contains data that is out of the partition key range, insert the data to the correct partition. If there is no correct partition where the data can be inserted to, an error is reported.

  NOTICE

  Only when **VALIDATION** is **WITH**, **VERBOSE** can be specified.

- **partition_new_name**

  Specifies a new partition name.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **UPDATE GLOBAL INDEX**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Updates all global indexes in the partitioned table to ensure accurate data queries using global indexes.

If this parameter is not used, all global indexes in the partitioned table will become invalid.

- **UPDATE DISTRIBUTED GLOBAL INDEX**

  Updates all available global secondary indexes in the partitioned table to ensure that the data in the global secondary indexes is consistent with that in the base table. EXCHANGE PARTITION also updates all available global secondary indexes in the ordinary table.

  If this parameter is not used, all global secondary indexes in the partitioned table become invalid. EXCHANGE PARTITION also invalidates all global secondary indexes in the ordinary table.

- **NO UPDATE DISTRIBUTED GLOBAL INDEX**

  If this parameter is used, all GSIs in the partitioned table become invalid. EXCHANGE PARTITION also invalidates all global secondary indexes in the ordinary table.

- **storage_parameter**

  Specifies the name of a storage parameter.

  It is called in online DDL operations. You are advised not to use it.

  – **online_ddl_status** (enumerated type)

    This parameter is used to update the online DDL execution status of a target table.

    It is called in online DDL operations. You are advised not to use it.

    - **on**: enables the online DDL status.
    - **off**: disables the online DDL status.
    - **init**: initializes online DDL temporary objects.
    - **catchup**: synchronizes incremental data for concurrent online DDL operations.
    - **catchup_basedata**: synchronizes basic data of online DDL operations.
    - **scan_catchup**: performs catchup when NOT NULL or CHECK constraints are added online.
    - **scan_catchup_basedata**: checks the basic data about adding NOT NULL or CHECK constraints online.
    - **vacuum_catchup**: performs catchup when VACUUM FULL is performed online.
    - **vacuum_verbose_catchup**: performs catchup when VACUUM FULL VERBOSE is performed online.
    - **last_catchup**: reports errors for the write service in the last online DDL catchup.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - **switch**: exchanges temporary tables in the last online DDL catchup.
  - **copy_switch**: exchanges temporary tables in the last online tablespace movement catchup.
  - **record_toast_oid**: records the OID of a TOAST table attached to a target table of online DDL operations.
  - **abort**: specifies that online DDL operations enter the abort state.
  - **flush_buffer**: forcibly flushes buffers to disks when tablespaces are moved online.
  - **tmp_rel**: indicates a temporary table created during an online DDL process.
  - **validate_constr**: validates the CHECK constraints created by the **NOT VALID** option.
  - **vacuum_switch**: performs the last round of incremental temporary table exchange using online VACUUM FULL.
- **online_ddl_lock_tag** (int type)

  Unique ID of a target table of online DDL operations, which is used to generate temporary objects. The ID is the hash value of (Table schema + Table name).

  It is called in online DDL operations. You are advised not to use it.

Examples

- Change the name of a table partition.

```
-- Create a partitioned table.
gaussdb=# CREATE TABLE test_p1 (col1 INT, col2 INT) PARTITION BY RANGE (col1)
(
    PARTITION p1 VALUES LESS THAN (10),
    PARTITION p2 VALUES LESS THAN (20),
    PARTITION p3 VALUES LESS THAN (MAXVALUE)
);

-- Modify a partition name.
gaussdb=# ALTER TABLE test_p1 RENAME PARTITION p3 TO pmax;

-- Query partition information.
gaussdb=# SELECT relname, boundaries, oid FROM pg_partition WHERE parentid='test_p1'::regclass
AND parttype <> 'r';
 relname | boundaries |  oid
---------+------------+-------
 p1      | {10}       | 17066
 p2      | {20}       | 17067
 pmax    | {NULL}     | 17068
(3 rows)
```

- Move partition tablespaces.

```
-- Create a partition.
gaussdb=# CREATE TABLESPACE tbs_data1 RELATIVE LOCATION 'tablespace1/tbs_data1';

-- Move partition tablespaces.
gaussdb=# ALTER TABLE test_p1 MOVE PARTITION P1 TABLESPACE tbs_data1;

-- Check the partition tablespace.
gaussdb=# SELECT relname, spcname FROM pg_partition t1, pg_tablespace t2 WHERE
T1.reltablespace=t2.oid and t1.parentid='test_p1'::regclass;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 relname |  spcname
---------+-----------
 p1      | tbs_data1
(1 row)

-- Move a partition tablespace online.
gaussdb=# ALTER TABLE test_p1 MOVE PARTITION P1 TABLESPACE pg_default;
gaussdb=# ALTER TABLE ONLINE test_p1 MOVE PARTITION P1 TABLESPACE tbs_data1;
ALTER TABLE
gaussdb=# SELECT relname, spcname FROM pg_partition t1, pg_tablespace t2 WHERE
T1.reltablespace=t2.oid and t1.parentid='t1'::regclass;
 relname | spcname
---------+---------
 p1      | tbs_data1
(1 row)
```

- Exchange partitions.

```
-- Create an ordinary table and insert data.
gaussdb=# CREATE TABLE test_ep1(col1 INT,col2 INT);
gaussdb=# INSERT INTO test_ep1 VALUES (GENERATE_SERIES(1,30), 1000);

-- Migrate data from an ordinary table to a partition.
gaussdb=# ALTER TABLE test_p1 EXCHANGE PARTITION (p1) WITH TABLE test_ep1 VERBOSE;

-- Query.
gaussdb=# SELECT COUNT(*) FROM test_p1 PARTITION (p1);
 count
-------
     9
(1 row)

-- Truncate table data.
gaussdb=# TRUNCATE TABLE test_ep1;
TRUNCATE TABLE
-- Insert data.
gaussdb=# INSERT INTO test_ep1 VALUES (GENERATE_SERIES(1,30), 1000);
INSERT 0 30
-- Migrate ordinary table data to a specified partition online.
gaussdb=# ALTER TABLE ONLINE test_p1 EXCHANGE PARTITION (p1) WITH TABLE test_ep1 VERBOSE;
ALTER TABLE
-- Query.
gaussdb=# SELECT COUNT(*) FROM test_p1 PARTITION (p1);
 count
-------
     9
(1 row)

-- Drop the test_ep1 table.
gaussdb=# DROP TABLE test_ep1;
```

- Merge partitions.

```
-- Merge the p2 and pmax partitions in the test_p1 table into the pmax partition.
gaussdb=# ALTER TABLE test_p1 MERGE PARTITIONS p2,pmax INTO PARTITION pmax;

-- View the partitions.
gaussdb=# SELECT relname, boundaries, oid FROM pg_partition WHERE parentid='test_p1'::regclass
AND parttype <> 'r' order by 1;
 relname | boundaries |  oid
---------+------------+-------
 p1      | {10}       | 17066
 pmax    | {NULL}     | 17070
(2 rows)

-- Drop the table.
gaussdb=# DROP TABLE test_p1;
DROP TABLE
-- Create a partitioned table.
gaussdb=# CREATE TABLE test_p1 (col1 INT, col2 INT) PARTITION BY RANGE (col1)
(
    PARTITION p1 VALUES LESS THAN (10),
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    PARTITION p2 VALUES LESS THAN (20),
    PARTITION p3 VALUES LESS THAN (MAXVALUE)
);
-- Merge the p2 and pmax partitions in the test_p1 table into pmax online.
gaussdb=# ALTER TABLE ONLINE test_p1 MERGE PARTITIONS p2,p3 INTO PARTITION pmax;
ALTER TABLE
gaussdb=# SELECT relname, boundaries FROM pg_partition WHERE parentid='test_p1'::regclass AND
parttype <> 'r' order by 1;
 relname | boundaries
---------+------------
 p1      | {10}
 pmax    | {NULL}
(2 rows)

-- Drop the table and tablespace.
gaussdb=# DROP TABLE test_p1;
gaussdb=# DROP TABLESPACE tbs_data1;
```

- Split partitions.

```
-- Create a table.
gaussdb=# CREATE TABLE test_r1 (col1 INT,col2 INT) PARTITION BY RANGE (col1)(
    PARTITION p1   VALUES LESS THAN (10),
    PARTITION pmax VALUES LESS THAN (MAXVALUE)
);

-- Split a partition.
gaussdb=# ALTER TABLE test_r1 SPLIT PARTITION pmax AT (20) INTO (PARTITION p2, PARTITION
pmax);
gaussdb=# ALTER TABLE test_r1 SPLIT PARTITION pmax INTO (
    PARTITION p3   VALUES LESS THAN (30),
    PARTITION pmax VALUES LESS THAN (MAXVALUE)
);

-- Query.
gaussdb=# SELECT relname, boundaries, oid FROM pg_partition WHERE parentid='test_r1'::regclass
AND parttype <> 'r' order by 1;
 relname | boundaries |  oid
---------+------------+-------
 p1      | {10}       | 17088
 p2      | {20}       | 17090
 p3      | {30}       | 17092
 pmax    | {NULL}     | 17093
(4 rows)

-- Drop the table.
gaussdb=# DROP TABLE test_r1;
DROP TABLE
-- Create a table.
gaussdb=# CREATE TABLE test_r1 (col1 INT,col2 INT) PARTITION BY RANGE (col1)(
    PARTITION p1   VALUES LESS THAN (10),
    PARTITION pmax VALUES LESS THAN (MAXVALUE)
);
-- Splite a partition online.
gaussdb=# ALTER TABLE ONLINE test_r1 SPLIT PARTITION pmax AT (20) INTO (PARTITION p2,
PARTITION pmax);
ALTER TABLE
gaussdb=# ALTER TABLE ONLINE test_r1 SPLIT PARTITION pmax INTO (
    PARTITION p3   VALUES LESS THAN (30),
    PARTITION pmax VALUES LESS THAN (MAXVALUE)
);
ALTER TABLE
gaussdb=#  SELECT relname, boundaries FROM pg_partition WHERE parentid='test_r1'::regclass AND
parttype <> 'r' order by 1;
 relname | boundaries
---------+------------
 p1      | {10}
 p2      | {20}
 p3      | {30}
 pmax    | {NULL}
(4 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop the test_r1 table.
gaussdb=# DROP TABLE test_r1;
-- Create a table.
gaussdb=# CREATE TABLE test_r2(col1 INT, col2 INT) PARTITION BY RANGE (col1)(
    PARTITION p1   START(1) END(10),
    PARTITION p2   START(10) END(20),
    PARTITION pmax START(20) END(MAXVALUE)
);

-- Split a partition.
gaussdb=# ALTER TABLE test_r2 SPLIT PARTITION pmax INTO (
    PARTITION p3 START(20) END(30),
    PARTITION pmax START(30) END (MAXVALUE)
);

-- Query.
gaussdb=# SELECT relname, boundaries, oid FROM pg_partition WHERE parentid='test_r2'::regclass
AND parttype <> 'r' order by 1;
 relname | boundaries |  oid
---------+------------+-------
 p1_0    | {1}        | 17112
 p1_1    | {10}       | 17113
 p2      | {20}       | 17114
 p3      | {30}       | 17116
 pmax    | {NULL}     | 17117
(5 rows)

-- Drop the table.
gaussdb=# DROP TABLE test_r1;
DROP TABLE
-- Create a table.
gaussdb=# CREATE TABLE test_r2(col1 INT, col2 INT) PARTITION BY RANGE (col1)(
    PARTITION p1   START(1) END(10),
    PARTITION p2   START(10) END(20),
    PARTITION pmax START(20) END(MAXVALUE)
);
-- Splite a partition online.
gaussdb=# ALTER TABLE ONLINE test_r2 SPLIT PARTITION pmax INTO (
    PARTITION p3 START(20) END(30),
    PARTITION pmax START(30) END (MAXVALUE)
);
ALTER TABLE
-- Query.
gaussdb=# SELECT relname, boundaries FROM pg_partition WHERE parentid='test_r2'::regclass AND
parttype <> 'r' order by 1;
 relname | boundaries
---------+------------
 p1_0    | {1}
 p1_1    | {10}
 p2      | {20}
 p3      | {30}
 pmax    | {NULL}
(5 rows)

-- Drop the test_r2 table.
gaussdb=# DROP TABLE test_r2;
-- Create a table.
gaussdb=# CREATE TABLE test_l1(col1 INT, col2 INT) PARTITION BY LIST(col1)(
    PARTITION p1   VALUES (10,20),
    PARTITION p2 VALUES (30,40)
);

-- Split a partition.
gaussdb=# ALTER TABLE test_l1 SPLIT PARTITION p1 VALUES (10) INTO (PARTITION p1_1, PARTITION
p1_2);
gaussdb=# ALTER TABLE test_l1 SPLIT PARTITION p2 INTO (PARTITION p3_1 VALUES(30), PARTITION
p3_2);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Query.
gaussdb=# SELECT relname, boundaries, oid FROM pg_partition WHERE parentid='test_l1'::regclass
AND parttype <> 'r' order by 1;
 relname | boundaries |  oid
---------+------------+-------
 p1_1    | {10}       | 17132
 p1_2    | {20}       | 17133
 p3_1    | {30}       | 17134
 p3_2    | {40}       | 17135
(4 rows)

-- Drop the test_l1 table.
gaussdb=# DROP TABLE test_l1;
```

- Add partitions.

```
-- Create a table.
gaussdb=# CREATE TABLE test_p2 (col1 INT, col2 INT) PARTITION BY RANGE (col1)(
    PARTITION p1 VALUES LESS THAN (10),
    PARTITION p2 VALUES LESS THAN (20)
);

-- Add a partition.
gaussdb=# ALTER TABLE test_p2 ADD PARTITION p3 VALUES LESS THAN (30);

-- Drop the test_p2 table.
gaussdb=# DROP TABLE test_p2;
-- Create a table.
gaussdb=# CREATE TABLE test_p3 (col1 INT, col2 INT) PARTITION BY LIST(col1)(
    PARTITION p1 VALUES (1),
    PARTITION p2 VALUES (2)
);

-- Add a partition.
gaussdb=# ALTER TABLE test_p3 ADD PARTITION p3 VALUES (3);

-- Drop the test_p3 table.
gaussdb=# DROP TABLE test_p3;
```

- Drop partitions.

```
-- Create a table.
gaussdb=# CREATE TABLE test_p4 (col1 INT, col2 INT) PARTITION BY LIST(col1)(PARTITION p1
VALUES (1),PARTITION p2 VALUES (2));

-- Drop the p2 partition from the test_p4 table.
gaussdb=# ALTER TABLE test_p4 DROP PARTITION p2;

-- Query.
gaussdb=# SELECT relname, boundaries, oid FROM pg_partition WHERE parentid='test_p4'::regclass;
 relname | boundaries |  oid
---------+------------+-------
 test_p4 |            | 17187
 p1      | {1}        | 17188
(2 rows)

-- Drop the test_p4 table.
gaussdb=# DROP TABLE test_p4;

-- Specify the partition value to delete a partition.
-- Create a table.
gaussdb=# CREATE TABLE test_p4 (col1 INT, col2 INT) PARTITION BY RANGE(col1)(PARTITION p1
VALUES LESS THAN(1),PARTITION p2 VALUES LESS THAN (2));

-- Drop the partition whose partition key is 1 from the test_p4 table.
gaussdb=# ALTER TABLE test_p4 DROP PARTITION FOR (1);

-- Query.
gaussdb=# SELECT relname, boundaries FROM pg_partition WHERE parentid='test_p4'::regclass order
by 1 desc;
 relname | boundaries
---------+------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 test_p4 |
 p1      | {1}
(2 rows)

-- Drop the test_p4 table.
gaussdb=# DROP TABLE test_p4;
```

- Clear partitions.

```
-- Create a table.
gaussdb=# CREATE TABLE test_p5 (col1 INT, col2 INT) PARTITION BY RANGE (col1)(
    PARTITION p1 VALUES LESS THAN (5),
    PARTITION p2 VALUES LESS THAN (10)
);

-- Insert data.
gaussdb=# INSERT INTO test_p5 VALUES (GENERATE_SERIES(1,9), 100);

-- View data in the p2 partition.
gaussdb=# SELECT * FROM test_p5 PARTITION (p2);
 col1 | col2
------+------
    5 |  100
    6 |  100
    7 |  100
    8 |  100
    9 |  100
(5 rows)

-- Clear data in the p2 partition.
gaussdb=# ALTER TABLE test_p5 TRUNCATE PARTITION p2;

-- View data in the p2 partition.
gaussdb=# SELECT * FROM test_p5 PARTITION (p2);
 col1 | col2
------+------
(0 rows)

-- Drop the test_p5 table.
gaussdb=# DROP TABLE test_p5;
```

Helpful Links

CREATE TABLE PARTITION and **DROP TABLE**

1.13.7.36 ALTER TABLE SUBPARTITION

Description

Alters partitions of a subpartitioned table, including adding, dropping, clearing, splitting, merging, exchanging, renaming partitions, moving partition tablespaces, and modifying partition attributes.

Precautions

- The tablespace of the added partition cannot be PG_GLOBAL.
- The name of any added partition must differ from the names of existing partitions and subpartitions in the partitioned table.
- The partition key value of any added partition must match the partition key type of the partitioned table.
- If a range partition is added, the key value of the added partition must be greater than the upper limit of the last range partition in the partitioned

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

table. To add a partition to a table with the **MAXVALUE** partition, you are advised to use the SPLIT syntax.

- If a list partition is added, the key value of the added partition cannot be the same as that of an existing partition. To add a partition to a table with the **DEFAULT** partition, you are advised to use the SPLIT syntax.
- Hash partitions cannot be added. However, if the subpartition mode of a subpartitioned table is hash but the partition mode is not hash, you can add a partition and create the corresponding subpartition.
- When the number of partitions in the target partitioned table reaches the maximum limit (1,048,575), no additional partitions can be added.
- If a partitioned table contains only one partition or subpartition, that partition or subpartition cannot be dropped.
- Hash partitions cannot be deleted.
- When you use PARTITION FOR() or SUBPARTITION FOR() to select partitions, the number of values specified within the parentheses must match the number of columns specified in the partition definition, and they must correspond in a one-to-one manner.
- Only subpartitions (leaf nodes) can be split. Only range and list partitioning policies can be used and hash partitioning policies are not supported.
- Only subpartitions (leaf nodes) can be merged, and the source partitions must belong to the same partition.
- Only the owner of a partitioned table or users with the ALTER permission on the partitioned table can run the **ALTER TABLE PARTITION** command. System administrators have the permission to run the command by default.
- Deleting, splitting, clearing, and exchanging partitions will invalidate global indexes. The UPDATE GLOBAL INDEX clause can be used to update indexes synchronously.
- If partitions are dropped, split, truncated, or exchanged without declaring the UPDATE GLOBAL INDEX clause, concurrent DML statements may trigger errors due to unusable indexes.
- If **enable_gpi_auto_update** is set to **on**, the global index is automatically updated even if the UPDATE GLOBAL INDEX clause is not declared.
- If the **enable_drop_truncate_partition_keep_gpi** parameter is set to **on**, the global index remains valid when partitions are dropped or truncated. In this case, the global index is always valid, regardless of whether the DROP/TRUNCATE PARTITION statement contains the UPDATE GLOBAL INDEX clause. This priority is higher than that of the **enable_gpi_auto_update** parameter and the UPDATE GLOBAL INDEX clause explicitly specified by the user.
- If you add a row-level expression when adding or changing an ILM policy for a data object, note that the row-level expression supports only the functions listed in the whitelist. For details about the whitelist function list, see **Row Expression Function Whitelist**.

  NOTE

  Among the functions supported by the row-level expression ON (EXPR) in the ILM policy, the output of some may be influenced by compatibility parameters. For instance, after you set **b_format_version** to **'5.7'** and **b_format_dev_version** to **'s2'** in MySQL-compatible mode, the upper function will be unable to convert data to uppercase as usual.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax

Modifying a partition in a subpartitioned table includes modifying the table partition itself and the table partition name, and resetting the partition ID.

- Main syntax for modifying a partition:

```
ALTER TABLE [ IF EXISTS ] { table_name  [*] | ONLY table_name | ONLY ( table_name  )}
    action [, ... ];
```

action includes the following partition maintenance clauses. For the partition continuity when multiple clauses are used for partition maintenance, GaussDB executes DROP PARTITION and ADD PARTITION first, and then the rest clauses in sequence.

```
{ move_clause |
    exchange_clause  |
    row_clause  |
    merge_clause  |
    modify_clause  |
    add_clause    |
    drop_clause   |
    split_clause  |
    truncate_clause  |
    ilm_clause }
```

  - move_clause: moves the partition to a different tablespace.

```
MOVE SUBPARTITION { subpartition_name | FOR ( subpartition_value [, ...] ) } TABLESPACE
tablespacename
```

  - exchange_clause: moves data from an ordinary table to a specified partition.

```
EXCHANGE SUBPARTITION { ( subpartition_name ) | FOR ( subpartition_value [, ...] ) }
    WITH TABLE {[ ONLY ] ordinary_table_name | ordinary_table_name * | ONLY
( ordinary_table_name )}
    [ { WITH | WITHOUT } VALIDATION ] [ VERBOSE ] [ UPDATE GLOBAL INDEX ]
```

    The ordinary table and partition involved in the exchange must meet the following requirements:

    - The number of columns in the ordinary table must match that in the partition, with consistent column details including names, data types, constraints, collations, storage parameters, and compression information.
    - The compression information must be consistent between the ordinary table and partition.
    - The number of indexes in the ordinary table must match the number of local indexes in the partition, with consistent index information.
    - The number of table constraints must be consistent between the ordinary table and partition, with consistent constraint information.
    - The ordinary table cannot be a temporary table, and the partitioned table should be a subpartitioned table.
    - Ordinary tables and partitioned tables do not support dynamic data masking and row-level security constraints.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- After the exchange is completed, the data of the ordinary table and partition are swapped, along with their tablespace information. At this point, statistics on the ordinary table and partition become unreliable, necessitating an ANALYZE operation.
- Non-partition keys cannot be used to create local unique indexes; only global unique indexes are permitted. Therefore, if an ordinary table contains a unique index, its data may fail to be exchanged.

  To exchange data, you can create an intermediate table, insert the partition's data into the intermediate table, truncate the partition, insert the ordinary table's data into the partitioned table, drop the ordinary table, and rename the intermediate table.
- If the DROP COLUMN operation is performed on an ordinary or partitioned table, the dropped column still exists physically. Therefore, you need to ensure that the dropped column of the ordinary table is strictly aligned with that of the partition. Otherwise, the exchange will fail.
- If the ordinary table and partition to be swapped contain the rowid column and are in the compressed state, the data in the ordinary table and partition is automatically decompressed after the partition is swapped.

– The row_clause syntax is used to set row movement of a partitioned table.

```
{ ENABLE | DISABLE } ROW MOVEMENT
```

– The merge_clause syntax is used to merge partitions into one. The maximum number of source partitions that can be merged in a command is 300.

```
MERGE SUBPARTITIONS { subpartition_name } [, ...] INTO SUBPARTITION subpartition_name
    [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n
{ day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM |
HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}]
[ TABLESPACE tablespacename ] [ UPDATE GLOBAL INDEX ]
```

NOTICE

- For range partitioning, the ranges of the source partitions must increase continuously, and the partition name after MERGE can be the same as the name of the last source partition. For list partitioning, there is no such range requirement on the source partitions, and the partition name after MERGE can be the same as that of any source partition. If the partition name after MERGE is the same as that of a source partition, they are considered as the same partition.
- If the GUC parameter **enable_ilm** is not enabled and the merge_clause syntax is used to merge multiple partitions with the ILM policy into one partition, the new partition does not inherit the ILM policy.
- Ustore tables do not support ALTER TABLE MERGE SUBPARTITIONS in transaction blocks and stored procedures.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The modify_clause syntax is used to set whether a partitioned index is usable. Here is an example of applying the syntax to partitions:

```
MODIFY PARTITION partition_name { UNUSABLE LOCAL INDEXES | REBUILD UNUSABLE LOCAL INDEXES }
```

  It can also be applied to subpartitions:

```
MODIFY SUBPARTITION subpartition_name { UNUSABLE LOCAL INDEXES | REBUILD UNUSABLE LOCAL INDEXES }
```

- The add_clause syntax is used to add one or more partitions to a specified partitioned table. Here is an example of applying the syntax to partitions:

```
ADD {partition_less_than_item | partition_list_item } [ ( subpartition_definition_list ) ]
```

  It can also be applied to subpartitions:

```
MODIFY PARTITION partition_name ADD subpartition_definition
```

  partition_less_than_item defines a range partition. The syntax is as follows:

```
PARTITION partition_name VALUES LESS THAN ( partition_value | MAXVALUE )[ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [ TABLESPACE tablespacename ]
```

  partition_list_item defines a list partition. The syntax is as follows:

```
PARTITION partition_name VALUES ( partition_value [, ...] | DEFAULT ) [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [ TABLESPACE tablespacename ]
```

  subpartition_definition_list contains the **subpartition_definition** object of one or more subpartitions. The syntax is as follows:

```
SUBPARTITION subpartition_name [ VALUES LESS THAN ( partition_value | MAXVALUE ) | VALUES ( partition_value [, ...] | DEFAULT )] [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [ TABLESPACE tablespace ]
```

  NOTICE

  If the partition is a hash partition, you cannot use ADD to add a partition. If the subpartition is a hash partition, you cannot use MODIFY to add a subpartition.

- The drop_clause syntax is used to remove a partition from a specified partitioned table. Here is an example of applying the syntax to partitions:

```
DROP PARTITION { partition_name | FOR ( partition_value ) } [ UPDATE GLOBAL INDEX ]
```

  It can also be applied to subpartitions:

```
DROP SUBPARTITION { subpartition_name | FOR ( partition_value, subpartition_value ) } [ UPDATE GLOBAL INDEX ]
```

  NOTICE

  - If the partition is a hash partition, the partition cannot be dropped. If the subpartition is a hash partition, the subpartition cannot be dropped.
  - At least one sub-partition must be retained.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

– The split_clause syntax is used to split one partition into multiple ones.

```
SPLIT SUBPARTITION { subpartition_name| FOR ( subpartition_value [, ...] ) } { split_point_clause | no_split_point_clause } [ UPDATE GLOBAL INDEX ]
```

NOTICE

- The partition name after SPLIT can be the same as the source partition name, but they are regarded as different partitions.
- When the GUC parameter **enable_ilm** is disabled, the resulting partitions do not inherit the ILM policy, even if the source partition being split by split_clause has an ILM policy.

▪ The split_point_clause syntax for specifying the split point for range partitioning is as follows:

```
AT ( subpartition_value ) INTO ( SUBPARTITION subpartition_name [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [ TABLESPACE tablespacename ] , SUBPARTITION subpartition_name [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [ TABLESPACE tablespacename ] )
```

NOTICE

The split point must be in the range of partition keys of the partition to be split. Specifying a split point can only split one partition into two partitions.

▪ The no_split_point_clause syntax for not specifying the split point for a list partitioned table is as follows:

```
INTO ( SUBPARTITION subpartition_name VALUES LESS THAN (subpartition_value) [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [ TABLESPACE tablespacename ][, ...] )
```

NOTICE

- The range defined for the first new partition must be greater than that of the previous partition (if any) of the partition being split.
- The range of the last partition is not defined, that is, the VALUES LESS THAN (subpartition_value) part is not defined; by default, the last partition inherits the upper boundary value of the range defined for the source partition.
- The new partition must meet the constraint of continuously increasing range.

▪ The split_point_clause syntax for specifying the split point for list-range partitioning is as follows:

```
VALUES ( subpartition_value_list ) INTO ( SUBPARTITION subpartition_name [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ]
UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}]
[ TABLESPACE tablespacename ] , SUBPARTITION subpartition_name [ ILM ADD POLICY
ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month |
year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT
AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [ TABLESPACE
tablespacename ] )
```

NOTICE

The split point must be a non-empty true subset of the source partition. By specifying a split point, you can only split one partition into two new partitions.

- The no_split_point_clause syntax for not specifying the split point for a list partitioned table is as follows:

```
INTO ( SUBPARTITION subpartition_name VALUES (subpartition_value_list) [ ILM ADD
POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day |
month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ]
UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}]
[ TABLESPACE tablespacename ][, ...] )
```

NOTICE

- Do not define the range for the last partition, that is, VALUES (subpartition_value_list), as it automatically covers the remaining portion of the source partition, excluding other subpartitions.
- If no split point is specified, each new partition must be a non-empty true subset of the source partition and does not overlap with each other.

– truncate_clause: truncates a partition in a partitioned table. Here is an example of applying the syntax to partitions:

```
TRUNCATE PARTITION  { partition_name | FOR (  partition_value [, ...] )  } [ UPDATE GLOBAL
INDEX ]
```

It can also be applied to subpartitions:

```
TRUNCATE SUBPARTITION  { subpartition_name | FOR (  subpartition_value [, ...] )  } [ UPDATE
GLOBAL INDEX ]
```

– The ilm_clause syntax adds an ILM policy to a partition. It supports the syntax for the OLTP table compression feature of data lifecycle management. Here is an example of applying the syntax to partitions:

```
MODIFY PARTITION partition_name ILM ADD POLICY ROW STORE COMPRESS ADVANCED
[ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR ) ]
```

It can also be used in subpartitions.

```
MODIFY SUBPARTITION subpartition_name ILM ADD POLICY ROW STORE COMPRESS
ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION
[ ON ( EXPR ) ]
```

- Alter the name of a partition. This syntax can be used to change partitions of a partitioned table.

```
ALTER TABLE [ IF EXISTS ] { table_name [*] | ONLY table_name | ONLY ( table_name  )}
    RENAME PARTITION { partition_name | FOR ( partition_value [, ...] ) } TO partition_new_name;
```

It can also be used to modify subpartitions in a partitioned table:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ALTER TABLE [ IF EXISTS ] { table_name [*] | ONLY table_name | ONLY ( table_name  )}
   RENAME SUBPARTITION { subpartition_name | FOR ( subpartition_value [, ...] ) } TO
subpartition_new_name;
```

- Reset a partition ID.

```
ALTER TABLE [ IF EXISTS ] { table_name [*] | ONLY table_name | ONLY ( table_name  )} RESET
PARTITION;
```

Parameters

- **table_name**

  Specifies the name of a partitioned table.

  Value range: existing partitioned table names.

- **subpartition_name**

  Specifies the name of a subpartition.

  Value range: existing subpartition names.

- **tablespacename**

  Specifies the destination tablespace for partition movement.

  Value range: existing tablespace names.

- **partition_value**

  Specifies partition key values.

  The values specified by PARTITION FOR (partition_value [, ...]) uniquely identify a partition.

  Value range: determined by the value range of the partition key for the involved partition.

- **subpartition_value**

  Specifies the values of the partition and subpartition keys.

  The values specified by SUBPARTITION FOR (subpartition_value [, ...]) uniquely identify a subpartition.

  Value range: determined by the value ranges of the partition keys for the involved partition and subpartition.

- **UNUSABLE LOCAL INDEXES**

  Sets all indexes in the partition as unusable.

- **REBUILD UNUSABLE LOCAL INDEXES**

  Rebuilds all indexes in the partition.

- **{ ENABLE | DISABLE } ROW MOVEMENT**

  Specifies whether to enable row movement.

  If an UPDATE operation changes the value of a tuple on the partition key, causing the tuple to move to a different partition, an error will occur based on the setting you have chosen, or the tuple will be transferred between partitions.

  Value range:

  - **ENABLE**: Row movement is enabled.
  - **DISABLE**: Row movement is disabled.

  By default, it is disabled.

- **ordinary_table_name**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the name of the ordinary table involved in data migration.

Value range: existing ordinary table names.

- **{ WITH | WITHOUT } VALIDATION**

  Checks whether data in the ordinary table falls within the partition key range of the partition involved in data migration.

  Value range:

  - **WITH**: The system checks whether data in the ordinary table falls within the partition key range of the target partition. If any data falls outside this range, an error is reported.
  - **WITHOUT**: The system does not check whether data in the ordinary table falls within the partition key range of the target partition.

  The default value is **WITH**.

  The check is time consuming, especially when the data volume is large. Therefore, if you are sure that all data in the ordinary table falls within the partition key range of the target partition, use **WITHOUT** to skip this check.

- **VERBOSE**

  When **VALIDATION** is **WITH**, if the ordinary table contains data that is out of the partition key range, insert the data to the correct partition. If there is no correct partition where the data can be inserted to, an error is reported.

  NOTICE

  Only when **VALIDATION** is **WITH**, **VERBOSE** can be specified.

- **partition_new_name**

  Specifies a new partition name.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **subpartition_new_name**

  Specifies a new subpartition name.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **UPDATE GLOBAL INDEX**

  Updates all global indexes in the partitioned table to ensure accurate data queries using global indexes. If this parameter is not used, all global indexes in the partitioned table will become invalid.

Examples

- Rename a partition.

```
-- Create a subpartitioned table tbl_rge_lst_test.
gaussdb=# CREATE TABLE tbl_lst_reg_test(
    area_id char(5),
    sdate char(8),
    eid char(5),
    sales_amt int
) PARTITION BY LIST(area_id) SUBPARTITION BY RANGE(sdate)(
    PARTITION p_1001 VALUES ('1001')(
        SUBPARTITION p_1001_201901 VALUES LESS THAN ('20190201'),
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    SUBPARTITION p_1001_201902 VALUES LESS THAN ('20190301'),
    SUBPARTITION p_1001_201903 VALUES LESS THAN ('20190401')
  ),
  PARTITION p_1002 VALUES ('1002')(
    SUBPARTITION p_1002_201901 VALUES LESS THAN ('20190201'),
    SUBPARTITION p_1002_201902 VALUES LESS THAN ('20190301'),
    SUBPARTITION p_100w VALUES LESS THAN ('20190401')
  )
);

-- Change the name of the subpartition p_100w to p_1002_201903.
gaussdb=# ALTER TABLE tbl_lst_reg_test RENAME SUBPARTITION p_100w TO p_1002_201903;

-- Query.
gaussdb=# SELECT table_name,partition_name,subpartition_name FROM db_tab_subpartitions
WHERE table_name = 'tbl_lst_reg_test' AND partition_name = 'p_1002';
   table_name    | partition_name | subpartition_name
-----------------+----------------+-------------------
 tbl_lst_reg_test | p_1002         | p_1002_201901
 tbl_lst_reg_test | p_1002         | p_1002_201902
 tbl_lst_reg_test | p_1002         | p_1002_201903
(3 rows)
```

- Move partition tablespaces.

```
-- Create the tbs_data1 tablespace.
gaussdb=# CREATE TABLESPACE tbs_data1 RELATIVE LOCATION 'tablespace/tbs_data1';

-- Move the subpartition p_1002_201901 to the tablespace tbs_data1.
gaussdb=# ALTER TABLE tbl_lst_reg_test MOVE SUBPARTITION p_1002_201901 TABLESPACE
tbs_data1;

-- Move the subpartition p_1002_201902 to the tablespace tbs_data1.
gaussdb=# ALTER TABLE tbl_lst_reg_test MOVE SUBPARTITION FOR ('1002','20190325') TABLESPACE
tbs_data1;

-- Query the tablespace of the modified subpartition.
gaussdb=# SELECT subpartition_name,tablespace_name FROM db_tab_subpartitions WHERE
subpartition_name IN ('p_1002_201901','p_1002_201903');
 subpartition_name | tablespace_name
-------------------+-----------------
 p_1002_201901     | tbs_data1
 p_1002_201903     | tbs_data1
(2 rows)
```

- Exchange partitions.

```
-- Create an ordinary table and insert data into the table.
gaussdb=# CREATE TABLE tbl_test(
  area_id char(5),
  sdate char(8),
  eid char(5),
  sales_amt int
);

gaussdb=# INSERT INTO tbl_test VALUES ('1002','20190326','00001',9000);
gaussdb=# INSERT INTO tbl_test VALUES ('1002','20190326','00002',7500);
gaussdb=# INSERT INTO tbl_test VALUES ('1002','20190326','00003',6000);

-- Insert data into the partitioned table.
gaussdb=# INSERT INTO tbl_lst_reg_test VALUES ('1002','20190301','00001',126);

-- Exchange partitions.
gaussdb=# ALTER TABLE tbl_lst_reg_test EXCHANGE SUBPARTITION (p_1002_201903) WITH TABLE
tbl_test;

-- Query data. Data in the table and partition is exchanged.
gaussdb=# SELECT * FROM tbl_lst_reg_test;
 area_id |  sdate   |  eid  | sales_amt
---------+----------+-------+-----------
 1002    | 20190326 | 00001 |      9000
 1002    | 20190326 | 00002 |      7500
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 1002    | 20190326 | 00003 |      6000
(3 rows)

gaussdb=# SELECT * FROM tbl_test;
 area_id |  sdate   |  eid  | sales_amt
---------+----------+-------+-----------
 1002    | 20190301 | 00001 |       126
(1 row)

-- Check that the partitioned table tablespace is exchanged.
gaussdb=# SELECT subpartition_name,tablespace_name FROM db_tab_subpartitions WHERE
subpartition_name = 'p_1002_201903';
 subpartition_name |  tablespace_name
-------------------+--------------------
 p_1002_201903     | DEFAULT TABLESPACE
(1 row)
```

- Merge partitions.

```
-- Merge partitions.
gaussdb=# ALTER TABLE tbl_lst_reg_test MERGE SUBPARTITIONS
p_1002_201901,p_1002_201902,p_1002_201903 INTO SUBPARTITION p_1002_20191;

-- Query subpartition information.
gaussdb=# SELECT table_name,partition_name,subpartition_name FROM db_tab_subpartitions
WHERE table_name = 'tbl_lst_reg_test';
    table_name    | partition_name | subpartition_name
------------------+----------------+-------------------
 tbl_lst_reg_test | p_1001         | p_1001_201901
 tbl_lst_reg_test | p_1001         | p_1001_201902
 tbl_lst_reg_test | p_1001         | p_1001_201903
 tbl_lst_reg_test | p_1002         | p_1002_20191
(4 rows)
```

- Add a partition.

```
-- Add a partition.
gaussdb=# ALTER TABLE tbl_lst_reg_test ADD PARTITION p_1003 VALUES('1003') (SUBPARTITION
p_1003_201901 VALUES LESS THAN ('20190201'));

-- Add a subpartition to a specific partition.
gaussdb=# ALTER TABLE tbl_lst_reg_test MODIFY PARTITION p_1003 ADD SUBPARTITION
p_1003_201902 VALUES LESS THAN ('20190301');

-- Query partition information.
gaussdb=# SELECT table_name,partition_name,subpartition_name FROM db_tab_subpartitions
WHERE table_name = 'tbl_lst_reg_test';
    table_name    | partition_name | subpartition_name
------------------+----------------+-------------------
 tbl_lst_reg_test | p_1001         | p_1001_201901
 tbl_lst_reg_test | p_1001         | p_1001_201902
 tbl_lst_reg_test | p_1001         | p_1001_201903
 tbl_lst_reg_test | p_1002         | p_1002_20191
 tbl_lst_reg_test | p_1003         | p_1003_201901
 tbl_lst_reg_test | p_1003         | p_1003_201902
(6 rows)
```

- Drop a partition.

```
-- Drop the subpartition p_1003_201902.
gaussdb=# ALTER TABLE tbl_lst_reg_test DROP SUBPARTITION p_1003_201902;

-- Query.
gaussdb=# SELECT table_name,partition_name,subpartition_name FROM db_tab_subpartitions
WHERE table_name = 'tbl_lst_reg_test' AND partition_name = 'p_1003';
    table_name    | partition_name | subpartition_name
------------------+----------------+-------------------
 tbl_lst_reg_test | p_1003         | p_1003_201901
(1 row)

-- Drop the partition p_1003.
gaussdb=# ALTER TABLE tbl_lst_reg_test DROP PARTITION p_1003;

-- Query.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT table_name,partition_name,subpartition_name FROM db_tab_subpartitions
WHERE table_name = 'tbl_lst_reg_test';
    table_name    | partition_name | subpartition_name
------------------+----------------+-------------------
 tbl_lst_reg_test | p_1001         | p_1001_201901
 tbl_lst_reg_test | p_1001         | p_1001_201902
 tbl_lst_reg_test | p_1001         | p_1001_201903
 tbl_lst_reg_test | p_1002         | p_1002_20191
(4 rows)
```

- Split partitions.

```
-- Specify the split point to split partitions.
gaussdb=# ALTER TABLE tbl_lst_reg_test SPLIT SUBPARTITION p_1002_20191
    AT ('20190201') INTO (SUBPARTITION p_1002_201901,SUBPARTITION p_1002_20191) UPDATE
GLOBAL INDEX;

-- Do not specify a point for splitting a partition.
gaussdb=# ALTER TABLE tbl_lst_reg_test SPLIT SUBPARTITION p_1002_20191 INTO (
    SUBPARTITION p_1002_201902 VALUES LESS THAN ('20190301'),
    SUBPARTITION p_1002_201903
) UPDATE GLOBAL INDEX;

-- Query partition information.
gaussdb=# SELECT table_name,partition_name,subpartition_name,high_value FROM
db_tab_subpartitions;
    table_name    | partition_name | subpartition_name | high_value
------------------+----------------+-------------------+------------
 tbl_lst_reg_test | p_1001         | p_1001_201901     | 20190201
 tbl_lst_reg_test | p_1001         | p_1001_201902     | 20190301
 tbl_lst_reg_test | p_1001         | p_1001_201903     | 20190401
 tbl_lst_reg_test | p_1002         | p_1002_201901     | 20190201
 tbl_lst_reg_test | p_1002         | p_1002_201902     | 20190301
 tbl_lst_reg_test | p_1002         | p_1002_201903     | 20190401
(6 rows)
```

- Clear partition data.

```
-- Clear the partition.
gaussdb=# ALTER TABLE tbl_lst_reg_test TRUNCATE PARTITION p_1001 UPDATE GLOBAL INDEX;

-- Clear the subpartition.
gaussdb=# ALTER TABLE tbl_lst_reg_test TRUNCATE SUBPARTITION p_1002_201903 UPDATE GLOBAL
INDEX;

-- Drop tables.
gaussdb=# DROP TABLE tbl_lst_reg_test;
gaussdb=# DROP TABLE tbl_test;

-- Drop the tablespace.
gaussdb=# DROP TABLESPACE tbs_data1;
```

Helpful Links

CREATE TABLE SUBPARTITION

1.13.7.37 ALTER TABLESPACE

Description

Modifies the attributes of a tablespace.

Precautions

- Only the tablespace owner or a user with the ALTER permission can run the **ALTER TABLESPACE** command. By default, system administrators have the permission. To modify a tablespace owner, you must be the tablespace owner or system administrator and a member of the new_owner role.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The ALTER TABLESPACE operation on a row-store table cannot be performed in a transaction block.
- Only the initial user can change the owner of a system tablespace to itself. Other changes are not allowed.
- To change the owner, you must also be a direct or indirect member of the new owning role.

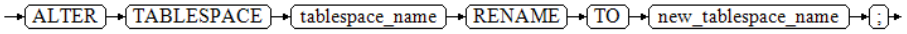
  NOTE

  If new_owner is the same as old_owner, the current user will not be verified. A message indicating successful ALTER execution is displayed.

Syntax

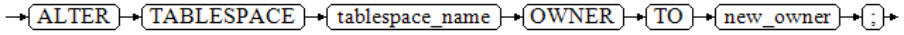
- The syntax of renaming a tablespace is as follows:

```
ALTER TABLESPACE tablespace_name
    RENAME TO new_tablespace_name;
```



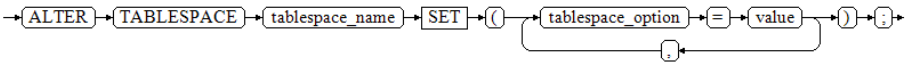
- The syntax of setting the owner of a tablespace is as follows:

```
ALTER TABLESPACE tablespace_name
    OWNER TO new_owner;
```



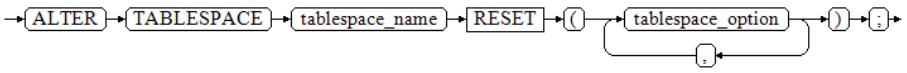
- The syntax of setting the attributes of a tablespace is as follows:

```
ALTER TABLESPACE tablespace_name
    SET ( { tablespace_option = value } [, ... ] );
```



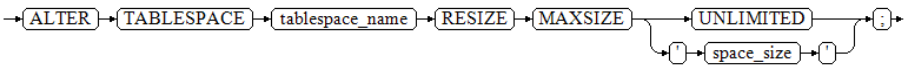
- The syntax of resetting the attributes of a tablespace is as follows:

```
ALTER TABLESPACE tablespace_name
    RESET ( { tablespace_option } [, ... ] );
```



- The syntax of setting the quota of a tablespace is as follows:

```
ALTER TABLESPACE tablespace_name
    RESIZE MAXSIZE { UNLIMITED | 'space_size'};
```



Parameters

- **tablespace_name**

  Specifies the tablespace to be modified.

  Value range: an existing tablespace name.

- **new_tablespace_name**

  Specifies the new name of the tablespace. The new name cannot start with **PG_**.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **new_owner**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the new owner of the tablespace.

Value range: an existing username.

- **tablespace_option**

  Sets or resets the parameters of a tablespace.

  Value range:

  - seq_page_cost: sets the optimizer to calculate the cost of obtaining disk pages in sequence. The default value is **1.0**.
  - **random_page_cost**: sets the optimizer to calculate the cost of obtaining disk pages in a non-sequential manner. The default value is **4.0**.

    NOTE

    - The value of **random_page_cost** is relative to that of **seq_page_cost**. It is meaningless when the value is equal to or less than the value of **seq_page_cost**.
    - The prerequisite for the default value **4.0** is that the optimizer uses indexes to scan table data and the hit ratio of table data in the cache is about 90%.
    - If the size of the table data space is smaller than that of the physical memory, decrease the value to a proper level. On the contrary, if the hit ratio of table data in the cache is lower than 90%, increase the value.
    - If random-access memory like SSD is adopted, the value can be decreased to a certain degree to reflect the cost of true random scan.

  Value range: Positive number of the floating-point type.

- **RESIZE MAXSIZE**

  Resets the maximum size of tablespace.

  Value range:

  - **UNLIMITED**: No limit is set for the tablespace.
  - The value is determined by space_size. For details about the format, see **CREATE TABLESPACE**.

    NOTE

    - If the adjusted quota is smaller than the current tablespace usage, the adjustment is successful. You need to decrease the tablespace usage to a value less than the new quota before writing data to the tablespace.

Examples

- Rename a tablespace.

```
-- Create a tablespace.
gaussdb=# CREATE TABLESPACE tbs_data1 RELATIVE LOCATION 'tablespace1/tbs_data1';

-- Rename a tablespace.
gaussdb=# ALTER TABLESPACE tbs_data1 RENAME TO tbs_data2;

-- Query.
gaussdb=# \db tbs_data2
            List of tablespaces
   Name    | Owner |       Location
-----------+-------+-----------------------
 tbs_data2 | omm   | tablespace1/tbs_data1
```

- Set the tablespace owner.

  Example:

```
-- Create a user.
gaussdb=# CREATE USER test PASSWORD '********';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Change the owner of the tablespace.
gaussdb=# ALTER TABLESPACE tbs_data2 OWNER TO test;

-- Query.
gaussdb=# \db tbs_data2
            List of tablespaces
   Name    | Owner |       Location
-----------+-------+----------------------
 tbs_data2 | test  | tablespace1/tbs_data1
(1 row)
```

- Set tablespace attributes.

  Example:

```
-- Change the value of seq_page_cost.
gaussdb=# ALTER TABLESPACE tbs_data2 SET (seq_page_cost = 10);

-- Query.
gaussdb=# SELECT * FROM pg_tablespace WHERE spcname = 'tbs_data2';
  spcname  | spcowner | spcacl |     spcoptions     | spcmaxsize | relative
-----------+----------+--------+--------------------+------------+----------
 tbs_data2 |    16778 |        | {seq_page_cost=10} |            | t
(1 row)
```

- Reset tablespace attributes.

```
-- Reset the value of seq_page_cost to the default value.
gaussdb=# ALTER TABLESPACE tbs_data2 RESET (seq_page_cost);

-- Query.
gaussdb=# SELECT * FROM pg_tablespace WHERE spcname = 'tbs_data2';
  spcname  | spcowner | spcacl | spcoptions | spcmaxsize | relative
-----------+----------+--------+------------+------------+----------
 tbs_data2 |    16778 |        |            |            | t
(1 row)
```

- Set the tablespace limit.

  Example:

```
-- Set the maximum size of a tablespace.
gaussdb=# ALTER TABLESPACE tbs_data2 RESIZE MAXSIZE '10G';

-- Query.
gaussdb=# SELECT * FROM pg_tablespace WHERE spcname = 'tbs_data2';
  spcname  | spcowner | spcacl | spcoptions | spcmaxsize | relative
-----------+----------+--------+------------+------------+----------
 tbs_data2 |    16778 |        |            | 10485760 K | t
(1 row)

-- Drop the tablespace.
gaussdb=# DROP TABLESPACE tbs_data2;

-- Drop the user.
gaussdb=# DROP USER test;
```

Helpful Links

CREATE TABLESPACE and **DROP TABLESPACE**

1.13.7.38 ALTER TEXT SEARCH CONFIGURATION

NOTICE

This function is for internal use only. You are advised not to use it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description

Alters the definition of a text search configuration. You can modify its mappings from strings to dictionaries, change the configuration's name or owner, or modify the parameters.

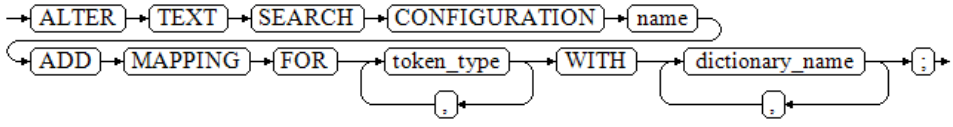
Precautions

- If a search configuration is referenced (to create indexes), users are not allowed to modify the text search configuration.
- To use ALTER TEXT SEARCH CONFIGURATION, you must be the owner of the configuration.
- The **ADD MAPPING FOR** option installs a list of string type mappings to be consulted for the specified token types; an error will be generated if there is already a mapping for any of the token types.
- The **ALTER MAPPING FOR** option removes the existing string type mappings for those token types and then adds specified string type mappings.
- The **ALTER MAPPING REPLACE ... WITH ...** and **ALTER MAPPING FOR ... REPLACE ... WITH ...** options replace **old_dictionary** with **new_dictionary**. Note that only when pg_ts_config_map has tuples corresponding to **maptokentype** and **old_dictionary**, the update will succeed. If the update fails, no messages are returned.
- The **DROP MAPPING FOR** option drops all string type mappings for the specified token types in the text search configuration. If **IF EXISTS** is not specified and the string type mappings specified by **DROP MAPPING FOR** do not exist in the text search configuration, an error will occur in the database.
- When the full-text search function is used, the return result of the text retrieval function is directly affected by the text search configuration. After the text search configuration is modified, the cached query plan is not updated immediately. As a result, when the function is called again for query, the return result of the function may not be updated synchronously. In this case, you need to clear the plan cache. For example, the related function exists in the PBE scenario or the scenario where automatic parameterization is enabled.

Syntax

- Add string type mappings to a text search configuration.

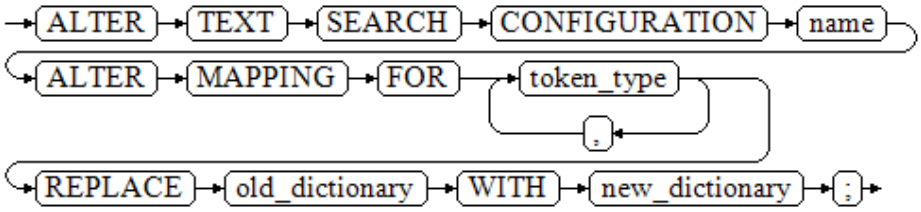
```
ALTER TEXT SEARCH CONFIGURATION name
    ADD MAPPING FOR { token_type } [, ... ] WITH { dictionary_name } [, ... ];
```



- Alter dictionaries in a text search configuration.

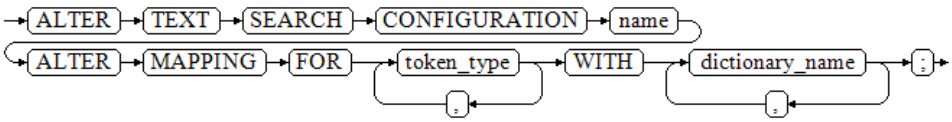
```
ALTER TEXT SEARCH CONFIGURATION name
    ALTER MAPPING FOR { token_type } [, ... ] REPLACE old_dictionary WITH new_dictionary;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



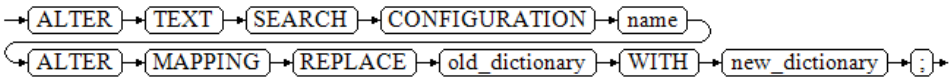
- Alter string type mappings in a text search configuration.

```
ALTER TEXT SEARCH CONFIGURATION name
    ALTER MAPPING FOR { token_type } [, ... ] WITH { dictionary_name } [, ... ];
```



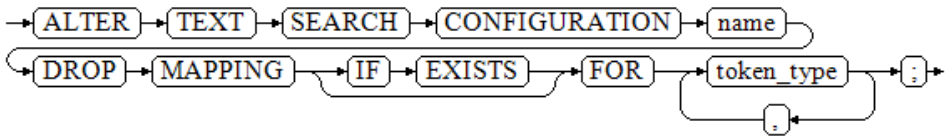
- Alter dictionaries in a text search configuration.

```
ALTER TEXT SEARCH CONFIGURATION name
    ALTER MAPPING REPLACE old_dictionary WITH new_dictionary;
```



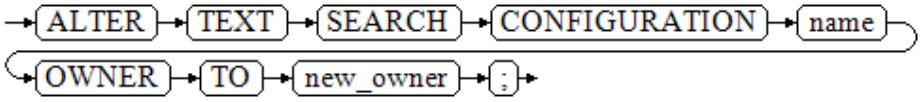
- Drop string type mappings from a text search configuration.

```
ALTER TEXT SEARCH CONFIGURATION name
    DROP MAPPING [ IF EXISTS ] FOR { token_type } [, ... ];
```



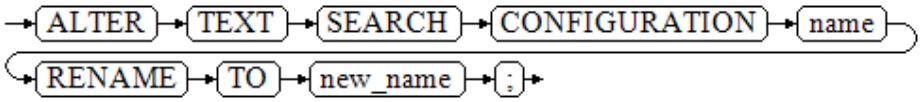
- Rename the owner of a text search configuration.

```
ALTER TEXT SEARCH CONFIGURATION name OWNER TO new_owner;
```



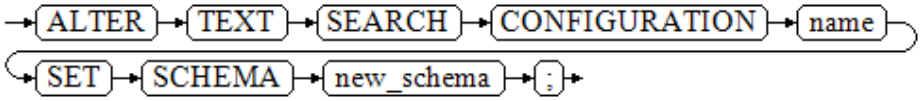
- Rename a text search configuration.

```
ALTER TEXT SEARCH CONFIGURATION name RENAME TO new_name;
```



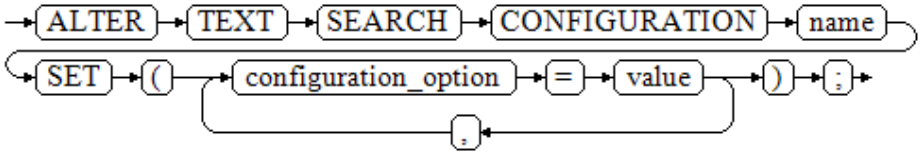
- Rename the namespace of a text search configuration.

```
ALTER TEXT SEARCH CONFIGURATION name SET SCHEMA new_schema;
```



- Alter the attributes of a text search configuration.

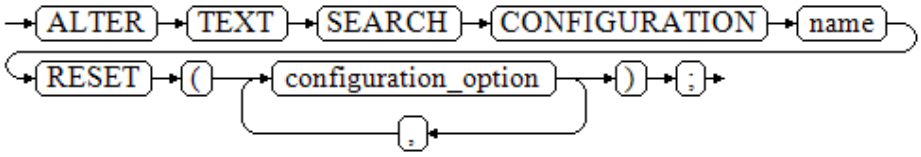
```
ALTER TEXT SEARCH CONFIGURATION name SET ( { configuration_option = value } [, ...] );
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Reset the attributes of a text search configuration.

```
ALTER TEXT SEARCH CONFIGURATION name RESET ( {configuration_option} [, ...] );
```



Parameters

- **name**

  Specifies the name (optionally schema-qualified) of an existing text search configuration.

- **token_type**

  Specifies the name of a token type that is emitted by the configuration's parser.

- **dictionary_name**

  Specifies the name of a text search dictionary. If multiple dictionaries are listed, they are searched in the specified order.

- **old_dictionary**

  Specifies the name of a text search dictionary to be replaced in the mapping.

- **new_dictionary**

  Specifies the name of a text search dictionary to be substituted for **old_dictionary**.

- **new_owner**

  Specifies the new owner of the text search configuration.

- **new_name**

  Specifies the new name of the text search configuration.

- **new_schema**

  Specifies the new schema for the text search configuration.

- **configuration_option**

  Specifies the text search configuration option.

- **value**

  Specifies the value of text search configuration option.

Examples

```
-- Create text search configuration english_1.
gaussdb=# CREATE TEXT SEARCH CONFIGURATION english_1 (parser=default);
CREATE TEXT SEARCH CONFIGURATION

-- Create text search configuration english_2.
gaussdb=# CREATE TEXT SEARCH CONFIGURATION english_2 (parser=default);
CREATE TEXT SEARCH CONFIGURATION

-- Add string type mappings to a text search configuration.
gaussdb=# ALTER TEXT SEARCH CONFIGURATION english_1 ADD MAPPING FOR word WITH
simple,english_stem;
ALTER TEXT SEARCH CONFIGURATION

gaussdb=# ALTER TEXT SEARCH CONFIGURATION english_1 ADD MAPPING FOR email WITH english_stem,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
french_stem;
ALTER TEXT SEARCH CONFIGURATION

-- Query information about the text search configuration.
gaussdb=# SELECT b.cfgname,a.maptokentype,a.mapseqno,a.mapdict,c.dictname FROM pg_ts_config_map
a,pg_ts_config b, pg_ts_dict c WHERE a.mapcfg=b.oid AND a.mapdict=c.oid AND b.cfgname='english_1'
ORDER BY 1,2,3,4,5;
  cfgname  | maptokentype | mapseqno | mapdict |    dictname
-----------+--------------+----------+---------+--------------
 english_1 |            2 |        1 |    3765 | simple
 english_1 |            2 |        2 |   12960 | english_stem
 english_1 |            4 |        1 |   12960 | english_stem
 english_1 |            4 |        2 |   12964 | french_stem
(4 rows)

-- Alter string type mappings in a text search configuration.
gaussdb=# ALTER TEXT SEARCH CONFIGURATION english_1 ALTER MAPPING REPLACE french_stem with
german_stem;
ALTER TEXT SEARCH CONFIGURATION

-- Query information about the text search configuration.
gaussdb=# SELECT b.cfgname,a.maptokentype,a.mapseqno,a.mapdict,c.dictname FROM pg_ts_config_map
a,pg_ts_config b, pg_ts_dict c WHERE a.mapcfg=b.oid AND a.mapdict=c.oid AND b.cfgname='english_1'
ORDER BY 1,2,3,4,5;
  cfgname  | maptokentype | mapseqno | mapdict |    dictname
-----------+--------------+----------+---------+--------------
 english_1 |            2 |        1 |    3765 | simple
 english_1 |            2 |        2 |   12960 | english_stem
 english_1 |            4 |        1 |   12960 | english_stem
 english_1 |            4 |        2 |   12966 | german_stem
(4 rows)

-- Rename the text search configuration english_2 to english_3.
gaussdb=# ALTER TEXT SEARCH CONFIGURATION english_2 RENAME TO english_3;

-- Create a user joe.
gaussdb=# CREATE USER joe IDENTIFIED BY '********';

-- Rename the owner of a text search configuration to joe.
gaussdb=# ALTER TEXT SEARCH CONFIGURATION english_3 OWNER TO joe;

-- Rename the namespace of a text search configuration.
gaussdb=# ALTER TEXT SEARCH CONFIGURATION english_3 set schema joe;

-- Drop a schema.
gaussdb=# DROP SCHEMA IF EXISTS joe CASCADE;

-- Drop user joe.
gaussdb=# DROP ROLE IF EXISTS joe;

-- Drop a text search configuration.
gaussdb=# DROP TEXT SEARCH CONFIGURATION english_1;
```

1.13.7.39 ALTER TEXT SEARCH DICTIONARY

NOTICE

This function is for internal use only. You are advised not to use it.

Description

Alters the definition of a full-text search dictionary, including the parameters, name, owner, and schema.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

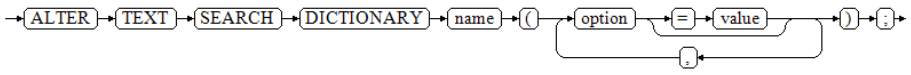
Precautions

- Predefined dictionaries do not support the ALTER operations.
- Only the owner of a dictionary or a system administrator can perform the ALTER operations. When separation of duties is disabled, system administrators have this permission by default.
- After a dictionary is created or modified, any modification to the customized dictionary definition file in the **filepath** directory does not affect the dictionary in the database. To use these modifications in the database, use the ALTER TEXT SEARCH DICTIONARY statement to update the definition file of the corresponding dictionary.
- When the full-text search function is used, the return result of the text search function is directly affected by the full-text search dictionary configuration. After the full-text search dictionary configuration is modified, the cached query plan is not updated immediately. As a result, when the function is called again for query, the return result of the function may not be updated synchronously. In this case, you need to clear the plan cache. For example, the related function exists in the PBE scenario or the scenario where automatic parameterization is enabled.

Syntax

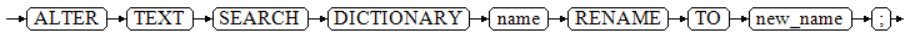
- Modify the dictionary definition.

```
ALTER TEXT SEARCH DICTIONARY name (
    { option [ = value ] } [, ... ]
);
```



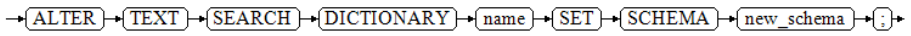
- Rename a dictionary.

```
ALTER TEXT SEARCH DICTIONARY name RENAME TO new_name;
```



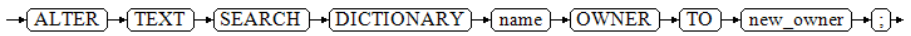
- Set the schema of the dictionary.

```
ALTER TEXT SEARCH DICTIONARY name SET SCHEMA new_schema;
```



- Change the owner of a dictionary.

```
ALTER TEXT SEARCH DICTIONARY name OWNER TO new_owner;
```



Parameters

- **name**

  Specifies the name of an existing dictionary. (If you do not specify a schema name, the dictionary in the current schema will be used.)

  Value range: an existing dictionary name

- **option**

  Specifies the name of a parameter to be modified. Each type of dictionaries has a template containing their custom parameters. Parameters function in a way irrelevant to their setting sequence. For details, see **option** in **CREATE TEXT SEARCH DICTIONARY**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The value of **TEMPLATE** in the dictionary cannot be changed.
- To specify a dictionary, specify both the dictionary definition file path (**FILEPATH**) and the file name.
- The name of a dictionary definition file can contain only lowercase letters, digits, and underscores (_).

- **value**

  Specifies the new value of a parameter. If the equal sign (=) and **value** are omitted, the previous settings of the option are deleted and the default value is used.

  Value range: valid values defined by **option**.

- **new_name**

  Specifies the new name of a dictionary.

  Value range: a string, which complies with the identifier naming convention. A value can contain a maximum of 63 characters.

- **new_owner**

  Specifies the new owner of a dictionary.

  Value range: an existing username.

- **new_schema**

  Specifies the new schema of a dictionary.

  Value range: an existing schema name.

Examples

```
-- Create the dictionary my_dict.
gaussdb=# CREATE TEXT SEARCH DICTIONARY my_dict ( TEMPLATE = Simple );

-- Create the dictionary my_dict1.
gaussdb=# CREATE TEXT SEARCH DICTIONARY my_dict1 ( TEMPLATE = Simple );

-- Rename the dictionary my_dict1 to my_dict2.
gaussdb=# ALTER TEXT SEARCH DICTIONARY my_dict1 RENAME TO my_dict2;

-- Modify the dictionary of the simple type, set non-stop words as recognized, and retain the values of other parameters.
gaussdb=# ALTER TEXT SEARCH DICTIONARY my_dict ( Accept = true );

-- Modify the dictionary of the simple type and reset the Accept parameter.
gaussdb=# ALTER TEXT SEARCH DICTIONARY my_dict ( Accept );

-- Update the dictionary definition and do not change any other content.
gaussdb=# ALTER TEXT SEARCH DICTIONARY my_dict ( dummy );

-- Drop dictionaries.
gaussdb=# DROP TEXT SEARCH DICTIONARY my_dict;
gaussdb=# DROP TEXT SEARCH DICTIONARY my_dict2;
```

1.13.7.40 ALTER TRIGGER

Description

ALTER TRIGGER is used to change the name and owner of a trigger.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

Currently, only the name and owner of a trigger can be modified.

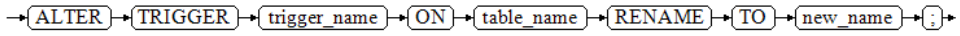
Precautions

- The owner of the table where a trigger resides or a user with the ALTER ANY TRIGGER permission can perform the ALTER TRIGGER operation. A system administrator has this permission by default.
- To modify the owner of a trigger, you must be the owner of the trigger or a system administrator, and you must be a direct or indirect member of the role to which the new owner belongs. The new owner must have the CREATE permission on the schema where the trigger is. When separation of duties is disabled, system administrators have this authorization by default. When separation of duties is enabled, system administrators does not this permission by default.
- Only the initial user can change the owner of a trigger to the initial user.

Syntax

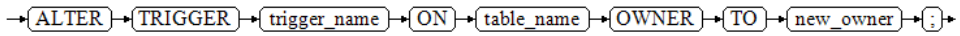
- Rename a trigger.

```
ALTER TRIGGER trigger_name ON table_name RENAME TO new_name;
```



- Change the owner of a trigger.

```
ALTER TRIGGER trigger_name ON table_name OWNER TO new_owner;
```



Parameters

- **trigger_name**

  Specifies the name of the trigger to be modified.

  Value range: an existing trigger

- **table_name**

  Specifies the name of the table where the trigger to be modified is located.

  Value range: an existing table having a trigger

- **new_name**

  New name of a trigger. The name cannot be the same as that of another trigger in the same table.

  Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

- **new_owner**

  Name of the new owner of a trigger.

  Value range: an existing username in the database.

Examples

```
-- Create a source table and a target table.
gaussdb=# CREATE TABLE test_trigger_src_tbl(id1 INT, id2 INT, id3 INT);
gaussdb=# CREATE TABLE test_trigger_des_tbl(id1 INT, id2 INT, id3 INT);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a DELETE trigger function.
gaussdb=# CREATE OR REPLACE FUNCTION TRI_DELETE_FUNC() RETURNS TRIGGER AS
        $$
        DECLARE
        BEGIN
                DELETE FROM test_trigger_des_tbl WHERE id1=OLD.id1;
                RETURN OLD;
        END
        $$ LANGUAGE plpgsql;

-- Create a DELETE trigger.
gaussdb=# CREATE TRIGGER delete_trigger BEFORE DELETE ON test_trigger_src_tbl FOR EACH ROW
EXECUTE PROCEDURE tri_delete_func();

-- Rename a trigger.
gaussdb=# ALTER TRIGGER delete_trigger ON test_trigger_src_tbl RENAME TO delete_trigger_renamed;

-- Change the owner of the trigger to jack. (Ensure that jack is a user created in the database.)
gaussdb=# ALTER TRIGGER delete_trigger ON test_trigger_src_tbl OWNER TO jack;

-- Delete the trigger.
gaussdb=# DROP TRIGGER delete_trigger_renamed ON test_trigger_src_tbl;

-- Delete the function.
gaussdb=# DROP FUNCTION tri_delete_func;

-- Delete the source table and target table.
gaussdb=# DROP TABLE test_trigger_src_tbl;
gaussdb=# DROP TABLE test_trigger_des_tbl;
```

Helpful Links

CREATE TRIGGER, **DROP TRIGGER**, and **ALTER TABLE**

1.13.7.41 ALTER TYPE

Description

Changes the definition of a type.

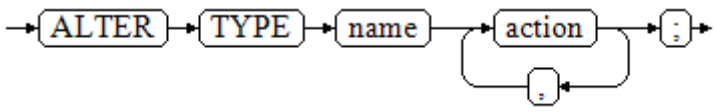
Precautions

Only the owner of a type, a user granted the ALTER permission on a type, or a user granted the ALTER ANY TYPE permission on a sequence can run the **ALTER TYPE** command. When separation of duties is disabled, system administrators have this permission by default. To modify the owner or schema of a type, you must be a type owner or system administrator and a member of the new owner role.

Syntax

- Change a type.

  ```
  ALTER TYPE name action [, ... ];
  ```

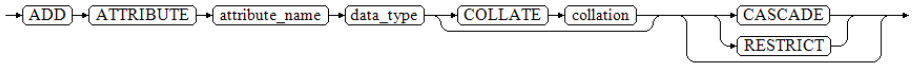
  

  The clauses corresponding to **action** are as follows:

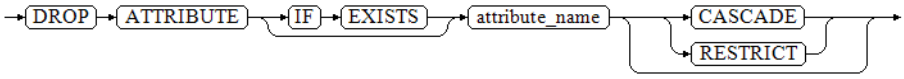
  – Add a new attribute to a composite type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

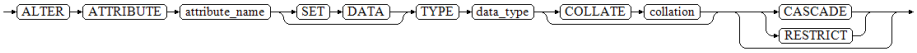
ADD ATTRIBUTE attribute_name data_type [COLLATE collation] [CASCADE | RESTRICT]



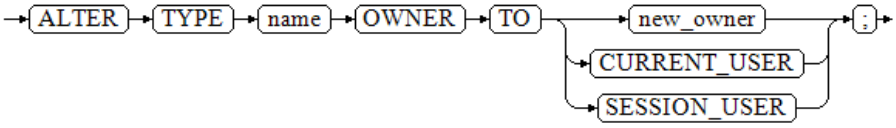
  - Delete an attribute from a composite type.
    DROP ATTRIBUTE [IF EXISTS] attribute_name [CASCADE | RESTRICT]



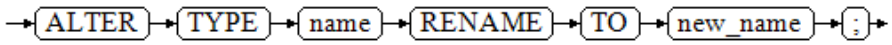
  - Change the type of an attribute in a composite type.
    ALTER ATTRIBUTE attribute_name [SET DATA] TYPE data_type [COLLATE collation] [CASCADE | RESTRICT]



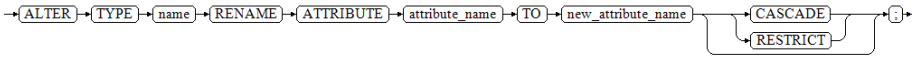
- Change the owner of a type.
  ALTER TYPE name OWNER TO { new_owner | CURRENT_USER | SESSION_USER };



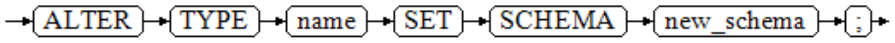
- Change the name of a type.
  ALTER TYPE name RENAME TO new_name;



- Change the name of an attribute in a composite type.
  ALTER TYPE name RENAME ATTRIBUTE attribute_name TO new_attribute_name [CASCADE | RESTRICT];



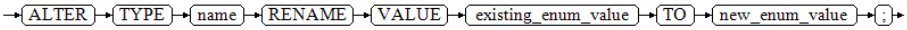
- Move a type to a new schema.
  ALTER TYPE name SET SCHEMA new_schema;



- Add a new value to an enumerated type.
  ALTER TYPE name ADD VALUE [IF NOT EXISTS] new_enum_value [{ BEFORE | AFTER } neighbor_enum_value];



- Change an enumerated value in the value list.
  ALTER TYPE name RENAME VALUE existing_enum_value TO new_enum_value;



Parameters

- **name**

  Specifies the name of an existing type that needs to be modified (optionally schema-qualified).

- **new_name**

  Specifies the new name of the type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **new_owner**

  Specifies the new owner of the type.
- **new_schema**

  Specifies the new schema of the type.
- **attribute_name**

  Specifies the name of the attribute to be added, modified, or deleted.
- **new_attribute_name**

  Specifies the new name of the attribute to be renamed.
- **data_type**

  Specifies the data type of the attribute to be added, or the new type of the attribute to be modified.
- **new_enum_value**

  Specifies a new enumerated value. It is a non-null string with a maximum length of 63 bytes.
- **neighbor_enum_value**

  Specifies an existing enumerated value before or after which a new enumerated value will be added.
- **existing_enum_value**

  Specifies an enumerated value to be changed. It is a non-null string with a maximum length of 63 bytes.
- **CASCADE**

  Determines that the type to be modified, its associated records, and subtables that inherit the type will all be updated.
- **RESTRICT**

  (Default) Refuses to update the associated records of the modified type.

NOTICE

- **ADD ATTRIBUTE**, **DROP ATTRIBUTE**, and **ALTER ATTRIBUTE** can be combined for modifying multiple attributes. For example, it is possible to add several attributes or change the types of several attributes at the same time in one command.
- To modify a schema of a type, you must have the CREATE permission on the new schema. To alter the owner, you must be a direct or indirect member of the new owner role, and that member must have CREATE permission on the schema of this type (these restrictions enforce that the operation of altering the owner cannot be done by other means other than deleting and rebuilding the type). However, the system administrators can change ownership of any type in any way when separation of duties is disabled. To add an attribute or modify the type of an attribute, you must also have the USAGE permission of this type.

- **CURRENT_USER**

  Specifies the current user.
- **SESSION_USER**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the current system user.

- **COLLATE collation**

  Assigns a collation to the column, which must be a sortable data type. If the collation is not specified, the default collation for the column's data type is used.

Examples

- Modify a composite type:

```
-- Create a composite type.
gaussdb=# CREATE TYPE typ_stu AS (name varchar(10),age int);

-- Create a table and insert data into the table. The data type of the info column is typ_stu.
gaussdb=# CREATE TABLE tbl_test (id int PRIMARY KEY,info typ_stu);
gaussdb=# INSERT INTO tbl_test VALUES (1,('Jim',16));

-- View data in the tbl_test table.
gaussdb=# SELECT * FROM tbl_test;
 id |   info
----+----------
  1 | (Jim,16)
(1 row)

-- Add an attribute to a composite type.
gaussdb=# ALTER TYPE typ_stu ADD ATTRIBUTE year int CASCADE;

-- Check data in the tbl_test table again. The info column contains an extra attribute.
gaussdb=# SELECT * FROM tbl_test;
 id |   info
----+-----------
  1 | (Jim,16,)
(1 row)

-- Delete an attribute from a composite type.
gaussdb=# ALTER TYPE typ_stu DROP ATTRIBUTE year;
```

- Change the type owner:

```
-- Create a user test.
gaussdb=# CREATE ROLE test PASSWORD '********';

-- Change the owner of typ_stu to test.
gaussdb=# ALTER TYPE typ_stu OWNER TO test;

-- Query the owner of typ_stu.
gaussdb=# SELECT t1.typname, t2.rolname AS owner
FROM pg_type t1, gs_roles t2
WHERE t1.typname = 'typ_stu' AND
      t1.typowner = t2.oid;
 typname | owner
---------+--------
 typ_stu | test
(1 row)
```

- Change a type or the name of an attribute in a composite type.

```
-- Rename one of the attributes age of typ_stu to age1.
gaussdb=# ALTER TYPE typ_stu RENAME ATTRIBUTE age to age1;

-- Run the following SQL statement to query the data whose age is 16 in the tbl_test table:
gaussdb=# SELECT id,(info).name,(info).age1 FROM tbl_test WHERE (info).age1 = 16;
 id | name | age1
----+------+------
  1 | Jim  |   16
(1 row)
```

- Move a type to a new schema.

```
-- Create a schema sctest.
gaussdb=# CREATE SCHEMA sctest;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Move typ_stu to the sc_test schema.
gaussdb=# ALTER TYPE typ_stu SET SCHEMA sctest;

-- Query the schema of typ_stu.
gaussdb=# \dT sctest.*
         List of data types
 Schema |      Name      | Description
--------+----------------+-------------
 sctest | sctest.typ_stu |
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE tbl_test;
gaussdb=# DROP TYPE sctest.typ_stu;
gaussdb=# DROP ROLE test;
gaussdb=# DROP SCHEMA sctest;
```

- Add a new value to an enumerated type.

```
-- Create an enumerated type typ_bugstatus.
gaussdb=# CREATE TYPE typ_bugstatus AS ENUM ('create', 'modify', 'closed');

-- Create a table tbl_test1.
gaussdb=# CREATE TABLE tbl_test1 (id serial, bugstat typ_bugstatus);

-- Insert data and ensure that the value of the bugstat column must be of the enumerated type.
Otherwise, an error is reported.
gaussdb=# INSERT INTO tbl_test1 (bugstat) VALUES ('closed');
gaussdb=# INSERT INTO tbl_test1 (bugstat) VALUES ('deleted');
ERROR:  invalid input value for enum typ_bugstatus: "deleted"
LINE 1: INSERT INTO tbl_test1 (bugstat) VALUES ('deleted');
                                                ^
CONTEXT:  referenced column: bugstat

-- View data in the tbl_test1 table.
gaussdb=# SELECT * FROM tbl_test1;
 id | bugstat
----+---------
  1 | closed
(1 row)

-- Add a tag value to an enumerated type.
gaussdb=# ALTER TYPE typ_bugstatus ADD VALUE IF NOT EXISTS 'regress' BEFORE 'closed';

-- Query.
gaussdb=# \dT+ typ_bugstatus
                                  List of data types
 Schema |     Name      | Internal name | Size | Elements | Access privileges | Description
--------+---------------+---------------+------+----------+-------------------+-------------
 public | typ_bugstatus | typ_bugstatus | 4    | create  +|                   |
        |               |               |      | modify  +|                   |
        |               |               |      | regress +|                   |
        |               |               |      | closed   |                   |
(1 row)
```

- Change an enumerated value in the value list.

```
-- Change closed in typ_bugstatus to close.
gaussdb=# ALTER TYPE typ_bugstatus RENAME VALUE 'closed' TO 'close';

-- Check the data in the tbl_test1 table. closed is changed to close.
gaussdb=# SELECT * FROM tbl_test1;
 id | bugstat
----+---------
  1 | close
(1 row)

-- Query.
gaussdb=# \dT+ typ_bugstatus;
                                  List of data types
 Schema |     Name      | Internal name | Size | Elements | Access privileges | Description
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------+---------------+---------------+------+----------+--------------------+-------------
 public | typ_bugstatus | typ_bugstatus | 4    | create  +|                    |
        |               |               |      | modify  +|                    |
        |               |               |      | regress +|                    |
        |               |               |      | close    |                    |
(1 row)

-- Delete.
gaussdb=# DROP TABLE tbl_test1;
gaussdb=# DROP TYPE typ_bugstatus;
```

Helpful Links

CREATE TYPE and **DROP TYPE**

1.13.7.42 ALTER USER

Description

Alters database user attributes, including the username and permission.

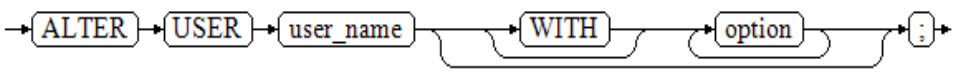
Precautions

When you use ALTER USER to modify session parameters, the changes will only apply to the specified user and will take effect in the next session.

Syntax

- Modify user permissions or other information.

```
ALTER USER user_name [ [ WITH ] option [ ... ] ];
```

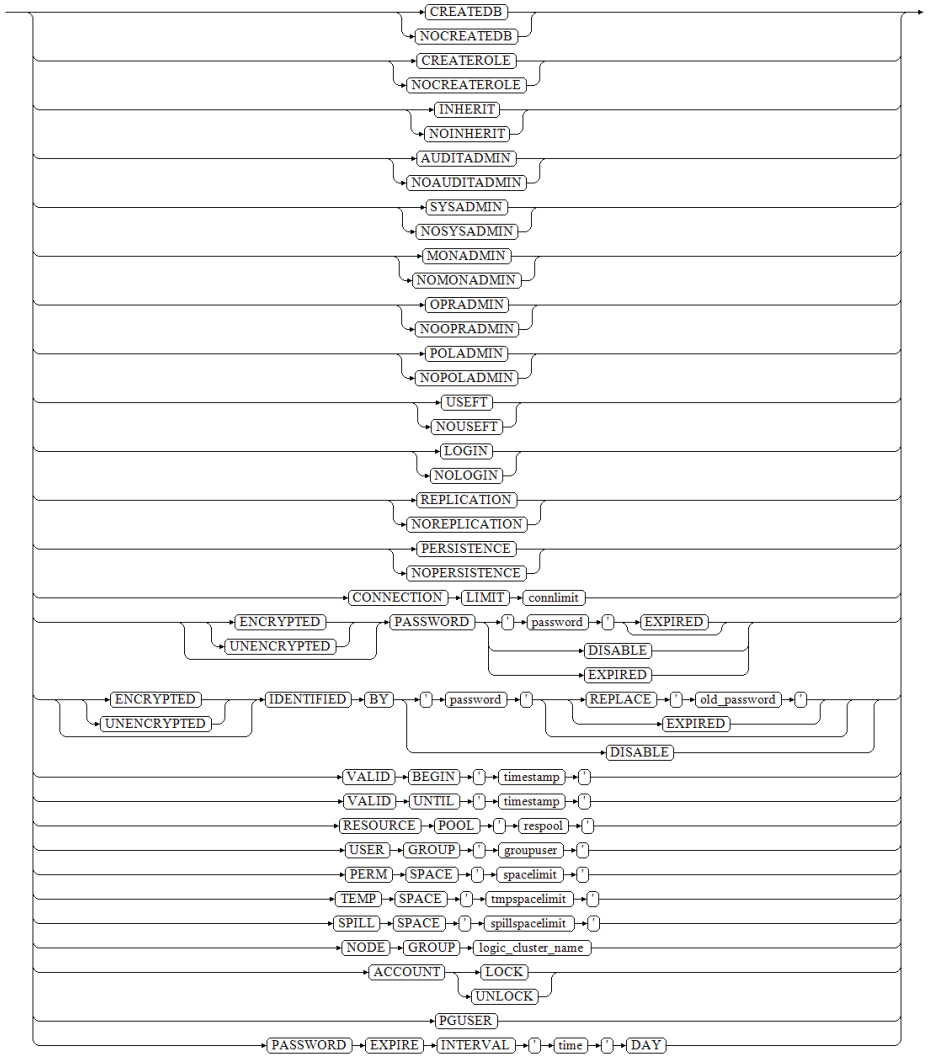


The option clause is as follows:

```
{
   { CREATEDB | NOCREATEDB }
   | { CREATEROLE | NOCREATEROLE }
   | { INHERIT | NOINHERIT }
   | { AUDITADMIN | NOAUDITADMIN }
   | { SYSADMIN | NOSYSADMIN }
   | { MONADMIN | NOMONADMIN }
   | { OPRADMIN | NOOPRADMIN }
   | { POLADMIN | NOPOLADMIN }
   | { USEFT | NOUSEFT }
   | { LOGIN | NOLOGIN }
   | { REPLICATION | NOREPLICATION }
   | { PERSISTENCE | NOPERSISTENCE }
   | CONNECTION LIMIT connlimit
   | [ ENCRYPTED | UNENCRYPTED ] PASSWORD { 'password' [EXPIRED] | DISABLE | EXPIRED }
   | [ ENCRYPTED | UNENCRYPTED ] IDENTIFIED BY { 'password' [ REPLACE 'old_password' |
EXPIRED ] | DISABLE }
   | VALID BEGIN 'timestamp'
   | VALID UNTIL 'timestamp'
   | RESOURCE POOL 'respool'
   | USER GROUP 'groupuser'
   | PERM SPACE 'spacelimit'
   | TEMP SPACE 'tmpspacelimit'
   | SPILL SPACE 'spillspacelimit'
   | NODE GROUP logic_cluster_name
   | ACCOUNT { LOCK | UNLOCK }
   | PGUSER
```

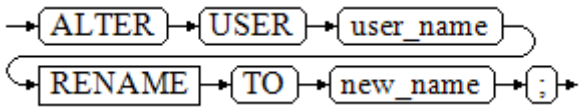
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  | PASSWORD EXPIRE INTERVAL 'time' DAY
}
```



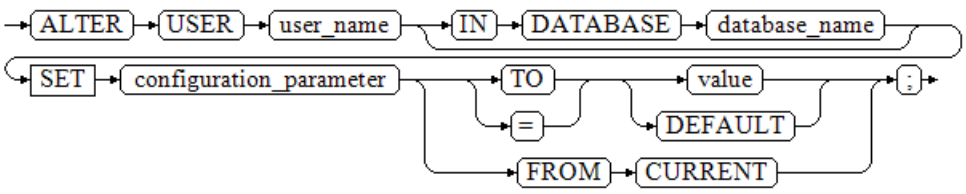
- Change the username.

```
ALTER USER user_name
  RENAME TO new_name;
```



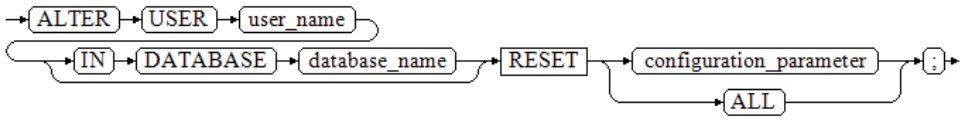
- Change the value of a session parameter associated with a user. The current version does not support the setting of user-level parameters.

```
ALTER USER user_name [ IN DATABASE database_name ]
  SET configuration_parameter {{ TO | = } { value | DEFAULT }|FROM CURRENT};
```



- Reset the value of a specified parameter associated with the user. The current version does not support the setting of user-level parameters.

```
ALTER USER user_name
    [ IN DATABASE database_name ] RESET {configuration_parameter|ALL};
```



Parameters

- **user_name**

  Specifies the current username.

  Value range: an existing username. If a username contains uppercase letters, declare the name with double quotation marks ("").

  NOTICE

  Currently, the RENAME TO operation cannot be performed on redisuser, an object used for resource control during redistribution. Performing the RENAME TO redisuser operation on other usernames is not supported.

- **new_password**

  Specifies a new password.

  Value range: character strings.

  Password rules:

  – Differ from the old password.
  – Contain at least eight characters. This is the default length.
  – Not be the same as or the reverse of the username.
  – Contain at least three types of the following: uppercase letters (A–Z), lowercase letters (a–z), digits (0–9), and special characters (limited to ~ ! @ # $ % ^ & * ( ) - _ = + \ | [{ }] ; : , < . > / ?). If the password contains characters other than the preceding characters, an error will be reported during statement execution.
  – The user password must be declared using single or double quotation marks.

- **old_password**

  Specifies the old password.

- **ACCOUNT { LOCK | UNLOCK }**

  – **ACCOUNT LOCK**: locks an account and prevents the account from logging in to the database using a password.
  – **ACCOUNT UNLOCK**: unlocks the account to allow database login.

- **PGUSER**

  In the current version, the **PGUSER** attribute of a user cannot be modified.

- For details about other parameters, see **Parameters** in **ALTER ROLE** and **Parameters** in **ALTER ROLE**.

Examples

```
-- Create user jim whose login password is ********.
gaussdb=# CREATE USER jim PASSWORD '********';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a database.
gaussdb=# CREATE DATABASE testdb1;

-- Change the login password of user jim.
gaussdb=# ALTER USER jim IDENTIFIED BY '**********' REPLACE '********';

-- Lock jim.
gaussdb=# ALTER USER jim ACCOUNT LOCK;

-- Unlock jim.
gaussdb=# ALTER USER jim ACCOUNT UNLOCK;

-- Change the username.
gaussdb=# ALTER USER jim RENAME TO lisa;

-- Drop the user.
gaussdb=# DROP USER lisa CASCADE;
```

Helpful Links

CREATE ROLE, **CREATE USER**, and **DROP USER**

1.13.7.43 ALTER VIEW

Description

ALTER VIEW changes the auxiliary properties of a view or recompiles a view (use CREATE OR REPLACE VIEW to change the query definition of a view) or recompiles a view.

Precautions

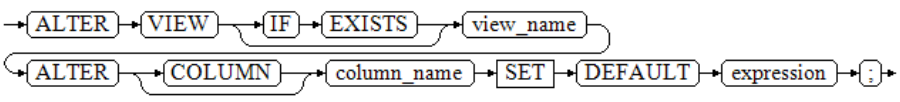
Only the owner of a view and users with the ALTER permission on the view are authorized to run the **ALTER VIEW** command. When separation of duties is disabled, system administrators have this authorization by default. Different permission restrictions may apply, depending on the specific attributes to modify.

- To modify the schema of a view, you must be either the view owner or a system administrator and have the CREATE permission on the new schema. However, when separation of duties is enabled, system administrators are restricted from making this modification.
- To modify the owner of a view, you must be either the view owner or a system administrator. Additionally, that user must be a member of the new owning role, and that role must have the CREATE permission on the view's schema. However, when separation of duties is enabled, system administrators are restricted from making this modification.

Syntax

- Set the default value of a column in a view.

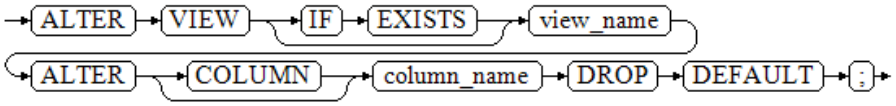
```
ALTER VIEW [ IF EXISTS ] view_name
    ALTER [ COLUMN ] column_name SET DEFAULT expression;
```



- Drop the default value of a column in a view.

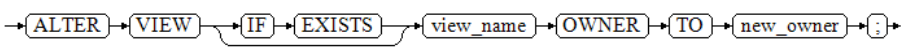
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ALTER VIEW [ IF EXISTS ] view_name
    ALTER [ COLUMN ] column_name DROP DEFAULT;
```



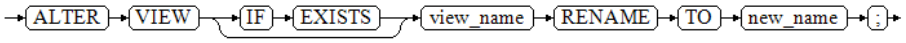
- Modify the owner of a view.

```
ALTER VIEW [ IF EXISTS ] view_name
    OWNER TO new_owner;
```



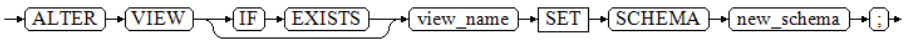
- Rename a view.

```
ALTER VIEW [ IF EXISTS ] view_name
    RENAME TO new_name;
```



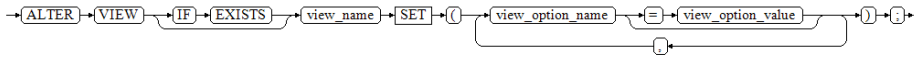
- Set the schema of a view.

```
ALTER VIEW [ IF EXISTS ] view_name
    SET SCHEMA new_schema;
```



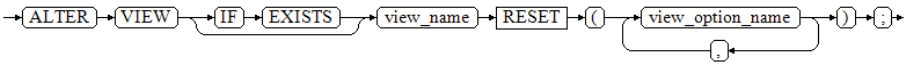
- Set the options of a view.

```
ALTER VIEW [ IF EXISTS ] view_name
    SET ( { view_option_name [ = view_option_value ] } [, ... ] );
```



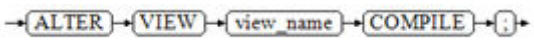
- Reset the options of a view.

```
ALTER VIEW [ IF EXISTS ] view_name
    RESET ( view_option_name [, ... ] );
```



- Recompile a view.

```
ALTER VIEW view_name COMPILE;
```



Parameters

- **IF EXISTS**

  If this option is used, no error is generated when the view does not exist, and only a message is displayed.

- **view_name**

  Specifies a view name (optionally schema-qualified).

  Value range: existing view names (character string).

- **column_name**

  Specifies the column name.

  Value range: a string, which indicates the name of a column in an existing view.

- **SET/DROP DEFAULT**

  Sets or drops the default value of a column. This parameter currently has no practical effect.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **new_owner**

  Specifies the new owner of a view.
- **new_name**

  Specifies a new name for the view.
- **new_schema**

  Specifies a new schema for the view.
- **view_option_name [= view_option_value]**

  Specifies an optional parameter for a view.
  - **security_barrier**: specifies whether the view provides row-level security. The value is of the Boolean type. The default value is **true**.
  - **check_option**: controls the behavior of updating a view. This parameter can be set to **CASCADED** or **LOCAL**. This parameter cannot be left blank.
- **expression**

  Specifies constants, functions, or SQL expressions.
- COMPILE

  Recompiles the view manually.

  NOTE

  For details about how to recompile an invalid view, see "Feature Usage Guidelines > Recompilation upon Invalidation" in *Developer Guide*.

Examples

- Rename a view.

```
-- Create the test_tbl table.
gaussdb=# CREATE TABLE test_tb1(col1 INT,col2 INT);

-- Create a view.
gaussdb=# CREATE VIEW abc AS SELECT * FROM test_tb1;

-- Set the default value of the view column to 100.
gaussdb=# ALTER VIEW IF EXISTS  abc ALTER  COLUMN  col1 SET DEFAULT 100;

-- Cancel the default value of the view column.
gaussdb=# ALTER VIEW IF EXISTS  abc ALTER  COLUMN  col1 DROP DEFAULT;

-- Rename the view.
gaussdb=# ALTER VIEW IF EXISTS abc RENAME TO test_v1;

-- Query the view.
gaussdb=# \dv
            List of relations
 Schema |  Name   | Type | Owner | Storage
--------+---------+------+-------+---------
 public | test_v1 | view | omm   |
(1 row)
```

- Modify the owner of a view.

```
-- Create a user.
gaussdb=# CREATE ROLE role_test PASSWORD '********';

-- Modify the owner of a view.
gaussdb=# ALTER VIEW IF EXISTS test_v1 OWNER TO role_test;

-- Query the view information.
gaussdb=# \dv
              List of relations
 Schema |  Name   | Type |   Owner   | Storage
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------+---------+------+-----------+---------
 public | test_v1 | view | role_test |
(1 row)
```

- Set the schema of a view.

```
-- Create a schema.
gaussdb=# CREATE SCHEMA tcpds;

-- Modify the schema of a view.
gaussdb=# ALTER VIEW test_v1 SET SCHEMA tcpds;

-- Query the view information.
gaussdb=# \dv tcpds.test_v1;
              List of relations
 Schema |  Name   | Type |   Owner   | Storage
--------+---------+------+-----------+---------
 tcpds  | test_v1 | view | role_test |
(1 row)
```

- Set and reset view options.

```
-- Change view options.
gaussdb=# ALTER VIEW tcpds.test_v1 SET (security_barrier = TRUE);
ALTER VIEW

-- Query.
gaussdb=# \dv tcpds.test_v1
                     List of relations
 Schema |  Name   | Type |   Owner   |         Storage
--------+---------+------+-----------+-------------------------
 tcpds  | test_v1 | view | role_test | {security_barrier=true}
(1 row)

-- Change the check_option option.
gaussdb=# ALTER VIEW tcpds.test_v1 SET (check_option = 'LOCAL');

-- Query.
gaussdb=# \dv tcpds.test_v1
                              List of relations
 Schema |  Name   | Type |   Owner   |                 Storage
--------+---------+------+-----------+-------------------------------------------
 tcpds  | test_v1 | view | role_test | {security_barrier=true,check_option=LOCAL}
(1 row)


-- Reset view options.
gaussdb=# ALTER VIEW tcpds.test_v1 RESET (security_barrier);
ALTER VIEW
gaussdb=# ALTER VIEW tcpds.test_v1 RESET (check_option);
ALTER VIEW

-- Query.
gaussdb=# \dv tcpds.test_v1
              List of relations
 Schema |  Name   | Type |   Owner   | Storage
--------+---------+------+-----------+---------
 tcpds  | test_v1 | view | role_test |
(1 row)

-- Drop the test_v1 view.
gaussdb=# DROP VIEW tcpds.test_v1;
DROP VIEW

-- Drop the test_tb1 table.
gaussdb=# DROP TABLE test_tb1;
DROP TABLE

-- Delete the user.
gaussdb=# DROP ROLE role_test;
DROP ROLE
-- Delete the schema.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# DROP SCHEMA tcpds;
DROP SCHEMA
```

- Recompile a view.

```
-- Enable view recompilation upon invalidation.
gaussdb=# SET enable_view_invalidation=on;

-- Create a table.
gaussdb=# CREATE TABLE test_tb1(col1 INT, col2 INT);

-- Create a view.
gaussdb=# CREATE VIEW test_v1 AS SELECT col1, col2 FROM test_tb1;

-- Drop the test_tb1 table.
gaussdb=# DROP TABLE test_tb1;
DROP TABLE

-- Check the view's validity.
gaussdb=# SELECT c.relname, o.valid FROM pg_catalog.pg_object o JOIN pg_catalog.pg_class c ON
c.oid = o.object_oid WHERE c.relname = 'test_v1';
 relname | valid
---------+-------
 test_v1 | f
(1 row)

-- Rebuild the test_tb1 table.
gaussdb=# CREATE TABLE test_tb1(col1 INT, col2 INT);

-- Recompile the view.
gaussdb=# ALTER VIEW test_v1 COMPILE;
ALTER VIEW

-- Check the view's validity again.
gaussdb=# SELECT c.relname, o.valid FROM pg_catalog.pg_object o JOIN pg_catalog.pg_class c ON
c.oid = o.object_oid WHERE c.relname = 'test_v1';
 relname | valid
---------+-------
 test_v1 | t
(1 row)

-- Drop the test_v1 view.
gaussdb=# DROP VIEW test_v1;
DROP VIEW

-- Drop the test_tb1 table.
gaussdb=# DROP TABLE test_tb1;
DROP TABLE
```

Helpful Links

CREATE VIEW and **DROP VIEW**

1.13.7.44 ANALYZE | ANALYSE

Description

- ANALYZE collects statistics about ordinary tables in a database and stores the results in the PG_STATISTIC and PG_STATISTIC_EXT system catalogs. After you run the **ANALYZE** command, you can query the collected statistics in the system catalogs or in the system views PG_STATS and PG_EXT_STATS. The execution plan generator uses these statistics to generate the most efficient execution plan.
- If no parameters are specified, ANALYZE analyzes each table and partitioned table in the database. You can narrow the analysis to a specific table, column,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

or partitioned table by specifying **table_name**, **column_name**, or **partition_name**.

- ANALYZE|ANALYSE VERIFY is used to verify files related to the target table in the database. The following table lists the supported verification types.

Page Type/ Verification Type	Physical Page Damage	Page Logic Verification	Tuple Verification
UHEAP	Supported	Supported	Supported in COMPLETE mode
HEAP	Supported	-	Supported in COMPLETE mode
FSM	Supported	-	-
VM	-	-	-
UBTREE	Supported	Supported	-
UBTREE.FSM(URQ)	Supported	Supported	-
BTREE	Supported	-	-
BTREE.FSM	Supported	-	-

- The time required for executing ANALYZE|ANALYSE VERIFY depends on the size of the current operation object (target table or database) and the performance of the current machine.
- Statistics collected each time are stored in historical statistics tables (**GS_STATISTIC_HISTORY**, **GS_STATISTIC_EXT_HISTORY**, and **GS_TABLESTATS_HISTORY**). The number of historical tables and the retention period of statistics are specified by the **stats_history_record_limit** and **stats_history_retention_time** parameters.

Precautions

- Non-temporary tables cannot be analyzed in an anonymous block, transaction block, function, or stored procedure. Temporary tables can be analyzed during execution of stored procedures, but statistics cannot be rolled back.
- If remote read is not involved, the remote read parameter does not take effect. If a key system catalog page is damaged, an error is reported and the detection stops.
- If no parameter to be analyzed is specified, tables on which the current user has the corresponding permission are analyzed by default. With a table specified, ANALYZE processes the table only.
- To perform ANALYZE operation to a table, you must be a table owner or a user granted the VACUUM permission on the table. By default, a system administrator has this permission. However, database owners are allowed to

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

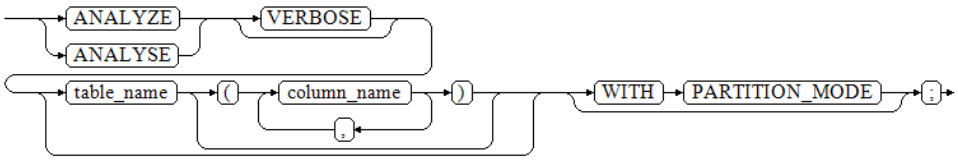
analyze all tables in their databases, except shared catalogs. (The restriction for shared catalogs implies that a true database-wide ANALYZE operation can only be performed by a system administrator). ANALYZE skips tables on which users do not have permissions.

- ANALYZE does not collect columns for which comparison or equivalent operations cannot be performed, for example, columns of the cursor type.
- If the table to be analyzed is empty, ANALYZE does not record the statistics of the table. The existing statistics are retained.

Syntax

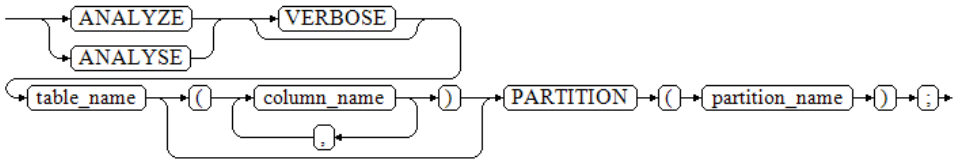
- Collect table statistics.

```
{ ANALYZE | ANALYSE } [ VERBOSE ]
    [ table_name [ ( column_name [, ...] ) ] ] [ WITH PARTITION_MODE ];
```



- Collect the partition statistics of a partitioned table.

```
{ ANALYZE | ANALYSE } [ VERBOSE ]
    table_name [ ( column_name [, ...] ) ] PARTITION ( partition_name ) ;
```

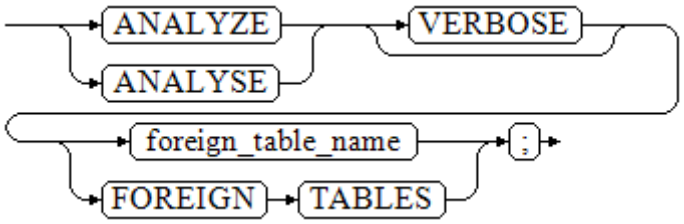


NOTE

- If the keyword PARTITION is included, **partition_name** must be the name of a partition.
- The distributed system does not support subpartitions.

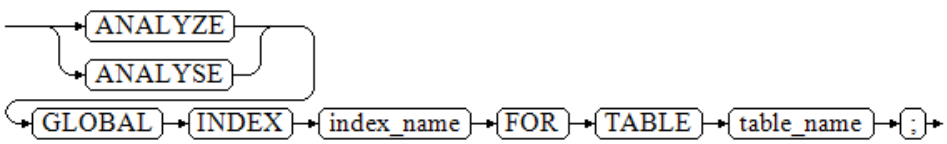
- Collect statistics about a foreign table.

```
{ ANALYZE | ANALYSE } [ VERBOSE ]
    { foreign_table_name | FOREIGN TABLES };
```



- Collect statistics about global secondary indexes.

```
{ ANALYZE | ANALYSE } GLOBAL INDEX index_name FOR TABLE table_name;
```



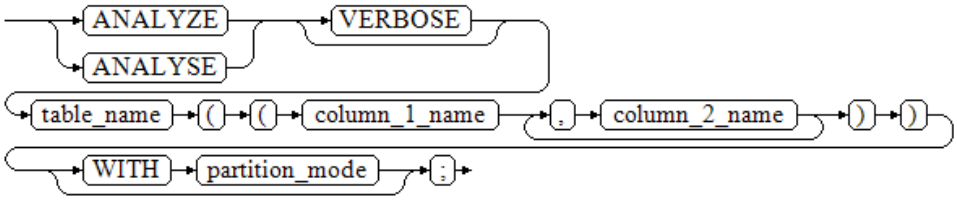
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- Before performing ANALYZE on the global secondary index, you need to perform ANALYZE on the base table to ensure the accuracy of global secondary indexes.
- After performing the REINDEX/REBUILD operation on the global secondary indexes, you need to perform ANALYZE on the base table first before the global secondary indexes to ensure the accuracy of global secondary indexes.
- To collect statistics from global secondary indexes, you must specify **index_name** and **table_name**, and **index_name** must be the name of a global secondary index.
- The global secondary indexes include only the statistics (reltuple and relpages) in the pg_class system catalog.
- If a concurrent autoanalyze occurs within the execution time window of ANALYZE, there is a low probability that the following error is reported: Duplicate key value violates unique constraint "gs_tablestats_history_relid_kind_time_index". In this case, you can manually execute ANALYZE again.

- Manually collect multi-column statistics.

```
{ ANALYZE | ANALYSE } [ VERBOSE ]
    table_name (( column_1_name, column_2_name [, ...] )) [ WITH partition_mode ];
```



NOTE

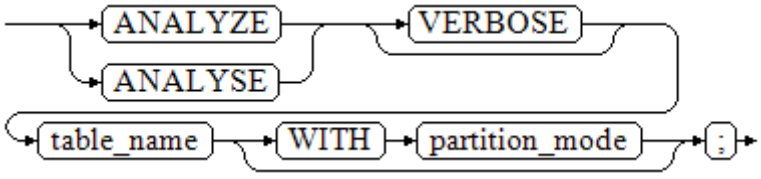
- If the GUC parameter **enable_functional_dependency** is disabled, each group of multi-column statistics supports a maximum of 32 columns.
- You are not allowed to collect statistics about multiple columns in system catalogs.

- Automatically collect multi-column statistics.

  If the ANALYZE statement is executed on a table after the **auto_statistic_ext_columns** parameter is enabled, multi-column statistics are automatically generated based on the index prefix of the table. The number of columns included in these statistics will not exceed the value of **auto_statistic_ext_columns**.

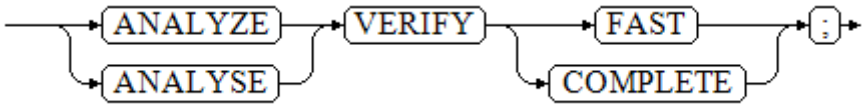
  For example, if table **t** has index (a,b,c,d) and **auto_statistic_ext_columns** is set to **4**, executing ANALYZE on table **t** will generate multi-column statistics for (a,b), (a,b,c), and (a,b,c,d).

```
{ ANALYZE | ANALYSE } [ VERBOSE ] table_name [ WITH partition_mode ];
```



- Verify data files in the current database.

```
{ ANALYZE | ANALYSE } VERIFY { FAST | COMPLETE };
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

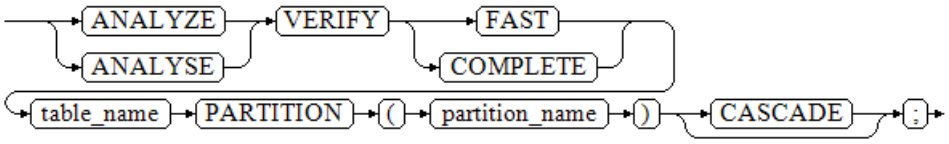
NOTE

- This operation reads data directly from disk. If concurrent DML operations are performed on the table being verified—that is, if disk files are modified concurrently—the retrieved data may be inaccurate, leading to false positives during verification. Therefore, you are advised to perform this operation when no database services are running or during off-peak hours.
- To ensure efficiency, it is advisable to run a checkpoint before performing this operation.
- You can perform operations on the entire database. Because a large number of tables are involved, you are advised to save the results in redirection mode.
  ```
  gsql -d database -p port -f sqlfile> sqllog.txt  2>&1
  ```
- NOTICE is displayed only for externally visible tables. The detection of internal tables is included in the foreign tables on which the internal tables depend and is not displayed externally.
- If any damage occurs during a database-wide operation, an error will be reported and the operation will halt.
- Only ordinary tables and ordinary indexes in the current database can be verified. Local temporary tables and unlogged tables are not supported. Any unsupported table types will be automatically skipped during verification.
- This operation can be performed on a standby node. When the database contains a large number of tables or a significant volume of data, the operation may take a long time. Due to the RECOVERY operation on the standby node, the ANALYZE VERIFY operation may be automatically canceled, with related error information displayed.

- Verify the data files of a table and an index.
  ```
  { ANALYZE | ANALYSE } VERIFY { FAST  | COMPLETE } { table_name  [ CASCADE ] | index_name };
  ```
  

  NOTE

  - This operation can be performed on ordinary tables and index tables. However, CASCADE is not permitted on the indexes of index tables, because it is intended to process all index tables associated with a main table. When verifying an index table only, the CASCADE mode is unnecessary.
  - When a main table is verified, its internal tables (such as TOAST tables) are also verified.
  - If the system indicates that index tables are damaged, it is advisable to run the **reindex** command to rebuild the indexes.
  - Local temporary tables and unlogged tables cannot be verified.
  - This operation can be performed on a standby node.

- Verify the data files of a table partition.
  ```
  { ANALYZE | ANALYSE } VERIFY { FAST | COMPLETE } table_name PARTITION (partition_name)
  [ CASCADE ];
  ```
  

NOTE

- You can verify a single partition within a table. However, CASCADE is not permitted on the indexes of index tables.
- Subpartitions cannot be specified.
- This operation can be performed on a standby node.

Parameters

- **VERBOSE**

  Enables the display of progress messages.

  NOTE

  When **VERBOSE** is specified, ANALYZE emits progress messages to indicate which table is currently being processed. Various statistics on the tables are printed as well.

- **table_name**

  Specifies a table name (optionally schema-qualified). If omitted, all non-foreign tables in the current database will be analyzed.

  Value range: existing table names.

- **column_name**, **column_1_name**, **column_2_name**

  Specify column names. All columns are analyzed by default.

  Value range: existing column names.

- **partition_name**

  Specifies a partition name after the keyword PARTITION to collect statistics on the corresponding partitioned table.

  Value range: names of partitions in the table.

- **foreign_table_name**

  Specifies the name of the specific table to be analyzed (possibly schema-qualified).

  Value range: existing table names.

- **FOREIGN TABLES**

  Analyzes all foreign tables accessible to the current user.

- **index_name**

  Specifies an index table name (optionally schema-qualified).

  Value range: existing table names.

- **FAST|COMPLETE**

  For a row-store table, the FAST mode verifies the CRC and page header of the row-store table. If the verification fails, an alarm is generated. In COMPLETE mode, the pointer and tuple of the row-store table are parsed and verified.

- **CASCADE**

  Verifies all indexes of the current table.

- **PARTITION_MODE**

  Cascadingly collects statistics about partitioned tables. **Table 1-392** describes the options and their meanings. This is not applicable to non-partitioned tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-392 PARTITION_MODE options

PARTITION_MODE Option	Description
ALL	Statistics about the entire table and partition are collected.
GLOBAL	Statistics about the entire table are collected.
PARTITION	Statistics about the partition are collected.
GLOBAL AND PARTITION	Statistics about the entire table and partition are collected.
ALL COMPLETE	Statistics about the entire table and partition are collected.
AUTO	Default value. The parameter configured in **statistic_granularity** or **default_statistic_granularity** is used. The priority of the table-level parameter **statistic_granularity** is higher than that of the global parameter **default_statistic_granularity**. For details about statistic_granularity, see **statistic_granularity**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The difference between **ALL** and **ALL COMPLETE** is as follows: In **ALL** mode, the entire table is sampled once. In **ALL COMPLETE** mode, each partition is sampled once. For partitions, the statistics calculated in **ALL COMPLETE** mode are more accurate than those in **ALL** mode, but the former mode consumes more resources.
- Partition statistics are used when the Cplan is pruned to a single partition. You are advised to select a proper policy for collecting statistics about partitioned tables based on the statistics collection time and actual service requirements.
- After large-scale data import or export, you are advised to manually collect partition-level statistics in ALL COMPLETE mode.
- When a version earlier than 505.0.0 is upgraded to 505.0.0 or later, **PARTITION_MODE** does not take effect during the upgrade observation period. The behavior of **PARTITION_MODE** is the same as that of the source version.
- When the ANALYZE operation is performed on the entire database, **PARTITION_MODE** cannot be specified. That is, syntax such as ANALYZE WITH GLOBAL is not supported.
- When the entire database is analyzed, **default_statistic_granularity** can be set to **GLOBAL** or **ALL**. In this case, the ANALYZE behavior is the same as that defined in **default_statistic_granularity**. When **default_statistic_granularity** is set to other values, the ANALYZE behavior is degraded to the GLOBAL behavior.
- Subpartition statistics cannot be collected.
- In **ALL** mode, if the value of **partition_statistics_min_rows** in the partitioned table is not 0 and the GUC parameter **enable_partition_statistics_min_rows** is enabled, the minimum number of sampled rows in the partition is the value of **partition_statistics_min_rows**. However, the maximum number of sampled rows in the entire table is controlled by the GUC parameter **maintenance_work_mem**. In addition, if data skew occurs in different partitions, you are advised to run **ANALYZE** in **GLOBAL** mode for the entire table after running **ANALYZE** in **ALL** mode. This ensures that the statistics of the entire table can accurately reflect the data skew characteristics.

Examples

- Collect table statistics.

```
-- Create the customer_info table.
gaussdb=# CREATE TABLE customer_info(
wr_returned_date_sk INTEGER ,
wr_returned_time_sk INTEGER ,
wr_item_sk INTEGER NOT NULL
,wr_refunded_customer_sk INTEGER ) DISTRIBUTE BY HASH (wr_item_sk);

-- Run ANALYZE to update statistics.
gaussdb=# ANALYZE customer_info;

-- Run ANALYZE VERBOSE to update statistics and display information about the customer_info table.
gaussdb=# ANALYZE VERBOSE customer_info;
INFO: analyzing "public.customer_info"(cn_5002 pid=53078)

-- Run ANALYZE VERBOSE to export the wr_returned_time_sk column information in the customer_info table.
gaussdb=# ANALYZE VERBOSE customer_info(wr_returned_time_sk);
INFO:  analyzing "public.customer_info"(cn_5002 pid=53078)
```

- Collect statistics about a partitioned table.

```
-- Create a partitioned table.
gaussdb=# CREATE TABLE customer_par(
wr_returned_date_sk INTEGER ,
wr_returned_time_sk INTEGER ,
wr_item_sk INTEGER NOT NULL,
wr_returned_customer_sk INTEGER)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DISTRIBUTE BY HASH (wr_item_sk)PARTITION BY RANGE(wr_returned_date_sk)(PARTITION P1
VALUES LESS THAN(2452275),PARTITION P2 VALUES LESS THAN(2452640),PARTITION P3 VALUES
LESS THAN(2453000),PARTITION P4 VALUES LESS THAN(MAXVALUE))ENABLE ROW MOVEMENT;

-- Run ANALYZE to update statistics.
gaussdb=# ANALYZE customer_par;

-- Run ANALYZE VERBOSE to export the information in the customer_par table.
gaussdb=# ANALYZE VERBOSE customer_par;
INFO:  analyzing "public.customer_par"(cn_5001 pid=446617)

-- Run ANALYZE VERBOSE to output information about the P1 partitioned table.
gaussdb=# ANALYZE VERBOSE customer_par PARTITION(P1);
INFO:  analyzing "public.customer_par"(cn_5001 pid=446617)
```

- Manually collect multi-column statistics.

```
-- Manually collect statistics in the wr_returned_date_sk and wr_returned_time_sk columns.
gaussdb=# ANALYZE VERBOSE customer_info (wr_returned_date_sk,wr_returned_time_sk);
INFO:  analyzing "public.customer_par"(cn_5001 pid=446617)
```

- Automatically collect multi-column statistics.

```
-- Create an index for the customer_info table.
gaussdb=# CREATE INDEX customer_index ON customer_info USING
btree(wr_returned_date_sk,wr_returned_time_sk,wr_item_sk,wr_refunded_customer_sk);

-- Set auto_statistic_ext_columns to 4.
gaussdb=# set auto_statistic_ext_columns=4;

-- Collect statistics about multiple columns automatically.
gaussdb=# ANALYZE VERBOSE customer_info;
INFO:  analyzing "public.customer_info"(cn_5001 pid=446617)
```

- Verify data files in the current database.

```
gaussdb=# ANALYZE VERIFY FAST;
```

- Verify the data files of a table and an index.

```
-- Check the customer_info table.
gaussdb=# ANALYZE VERIFY FAST customer_info;

-- Check the customer_index index.
gaussdb=# ANALYZE VERIFY FAST customer_index;
```

- Verify the data files of a table partition.

```
-- Check the P1 partition in the partitioned table customer_par.
gaussdb=# ANALYZE VERIFY FAST customer_par PARTITION (P1);
```

- Drop data.

```
-- Drop the index customer_index.
gaussdb=# DROP INDEX customer_index;

-- Drop the customer_info table.
gaussdb=# DROP TABLE customer_info;

-- Drop the customer_par partitioned table.
gaussdb=# DROP TABLE customer_par;
```

1.13.7.45 AUTOHINT

Description

AUTOHINT is used to explore hint combinations and outputs recommended hint combination that can accelerate the execution of the current SQL statement.

Precautions

- Only SELECT statements are supported. The user must have the super or administrator permission.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- When using any of the supported options, except for **ANALYZE**, remember to enclose them in parentheses, like in **AUTOHINT (VERBOSE) statement**.

Syntax

- Use AUTOHINT to recommend hints for an executing SQL statement. Multiple options are supported, regardless of their sequence.

```
AUTOHINT [ (  option  [, ...] )  ] statement;
```

The syntax of the option clause is as follows:

```
{
   ANALYZE [ boolean ] |
   VERBOSE [ boolean ] |
   SQLPATCH [ boolean ] |
   TEST [ boolean ] |
   DEBUG [ boolean ]
}
```

Parameters

- **statement**

  Specifies the SQL statement (limited to SELECT statements) that requires intelligent hint recommendation.

  NOTICE

  If the SQL statement already includes hints, AUTOHINT ignores the existing hints and recommends hints. For example, when being applied to the **SELECT /*+IndexScan(t1)*/* FROM t1, t2 where t1.id = t2.id and t1.a > 20;** statement, AUTOHINT will recommend hints based on the **SELECT * FROM t1, t2 where t1.id = t2.id and t1.a > 20;** statement rather than offering further recommendations based on the existing IndexScan(t1).

- **ANALYZE boolean**

  Before hint recommendation, hint exploration is performed. If hint exploration is disabled, hints are recommended for the current SQL statement based on historical hint exploration records.

  Value range:

  – **TRUE** (default value): Hint exploration is performed before recommendation.

  – **FALSE**: Hint exploration is not performed.

- **VERBOSE boolean**

  Displays additional information about hint exploration and recommendation.

  Value range:

  – **TRUE** (default): The additional information is displayed.

  – **FALSE**: It is not displayed.

- **SQLPATCH boolean**

  Outputs the corresponding SQLPATCH command after hint recommendation.

  Value range:

  – **TRUE** (default value): The SQLPATCH command is output.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - **FALSE**: It is not output.

- **TEST boolean**

  After hint recommendation, the initial query and the query with hints applied are executed to verify the recommendation result, and the actual plan execution time is output.

  Value range:

  - **TRUE** (default value): The recommendation result is verified and the verification result is output.
  - **FALSE**: Recommendations are not verified.

- **DEBUG boolean**

  Outputs more details than the **VERBOSE** option, including the initial plan tree and execution time of each round of exploration, the exploration subgraph obtained after each round of exploration ends, and the final merged exploration graph.

  Value range:

  - **TRUE** (default value): The details are output.
  - **FALSE**: It is not output.

Example 1

Use AUTOHINT to recommend hints for a slow SQL statement. The statement in the JOB data set is used as an example.

```
-- Import the JOB data set and use the ANALYZE option to explore and recommend hints for the 02a query.
gaussdb=# AUTOHINT ANALYZE SELECT MIN(t.title) AS movie_title
FROM company_name AS cn,
    keyword AS k,
    movie_companies AS mc,
    movie_keyword AS mk,
    title AS t
WHERE cn.country_code ='[de]'
  AND k.keyword ='character-name-in-title'
  AND cn.id = mc.company_id
  AND mc.movie_id = t.id
  AND t.id = mk.movie_id
  AND mk.keyword_id = k.id
  AND mc.movie_id = mk.movie_id;
-- Obtain the HINT recommendation result with expected improvement.
gaussdb=#
                      AUTOHINT
-------------------------------------------------------------
 Recommended hintset:
   MergeJoin(@sel$1 k@sel$1 mk@sel$1 mc@sel$1 cn@sel$1)

 Expected boost:
   planning:     0%
   execution:    50%
   overall:      50%
(7 rows)
```

1.13.7.46 AUTOHINT DROP MODEL

Description

Drop the historical hint exploration results of a specified SQL statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

Only SELECT statements are supported. The user must have the super or administrator permission.

Syntax

Drop the historical hint exploration results of a specified SQL statement.

```
AUTOHINT DROP MODEL statement;
```

Parameters

statement

Specifies the SQL statement (only SELECT statements are supported) whose exploration record is to be dropped.

Examples

Use AUTOHINT DROP MODEL to drop the historical HINT exploration results of a specified SQL statement.

```
gaussdb=# AUTOHINT DROP MODEL SELECT MIN(t.title) AS movie_title
FROM company_name AS cn,
    keyword AS k,
    movie_companies AS mc,
    movie_keyword AS mk,
    title AS t
WHERE cn.country_code ='[de]'
  AND k.keyword ='character-name-in-title'
  AND cn.id = mc.company_id
  AND mc.movie_id = t.id
  AND t.id = mk.movie_id
  AND mk.keyword_id = k.id
  AND mc.movie_id = mk.movie_id;
-- The deletion is successful.
AUTOHINT: 1 MODEL REMOVED
-- If no historical exploration result of the SQL statement exists, the deletion fails.
AUTOHINT: 0 MODELS REMOVE
```

1.13.7.47 AUTOHINT PURGE

Description

Purges the historical hint exploration results of AUTOHINT.

Precautions

The user must have the super permission or administrator permission.

Syntax

Purge the historical hint exploration results of AUTOHINT.

```
AUTOHINT PURGE;
```

Examples

Use AUTOHINT PURGE to purge all historical exploration results of AUTOHINT.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# AUTOHINT PURGE;
-- The PURGE operation is successful.
AUTOHINT: 12 MODELS REMOVED
-- The PURGE operation fails if no historical exploration result is available.
AUTOHINT: 0 MODELS REMOVE
```

1.13.8 B

1.13.8.1 BEGIN

Description

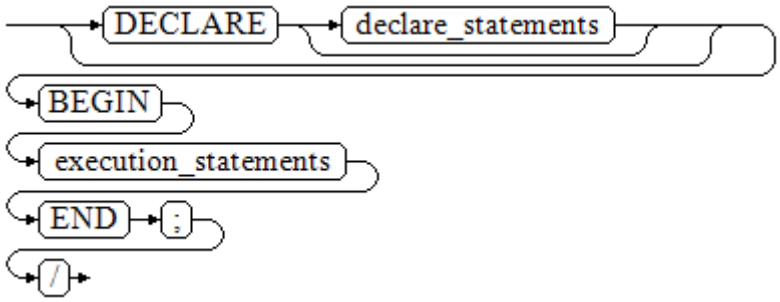
BEGIN may be used to initiate an anonymous block or a single transaction.

An anonymous block is a structure that can dynamically create and execute stored procedure code instead of permanently storing code as a database object in the database.

Syntax

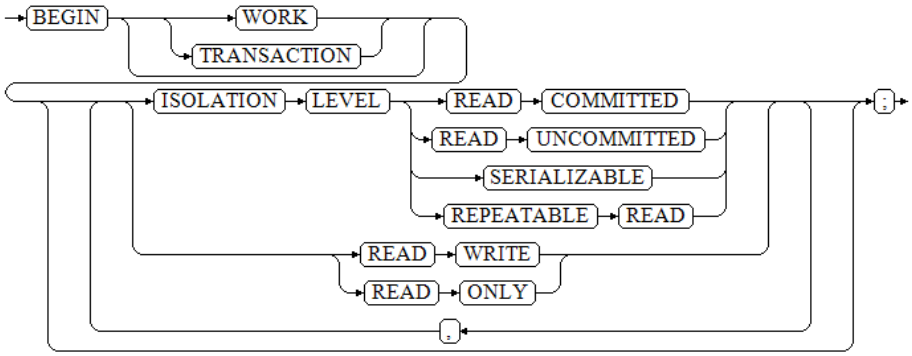
- Enable an anonymous block.

```
[DECLARE [declare_statements]]
BEGIN
execution_statements
END;
/
```



- Start a transaction.

```
BEGIN [ WORK | TRANSACTION ]
 [
   {
     ISOLATION LEVEL { READ COMMITTED | READ UNCOMMITTED | SERIALIZABLE | REPEATABLE
READ }
     | { READ WRITE | READ ONLY }
    } [, ...]
 ];
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **declare_statements**

  Declares a variable, including its name and type, for example, **sales_cnt int**.

- **execution_statements**

  Specifies the statement to be executed in an anonymous block.

  Value range: DML operations (SELECT, INSERT, DELETE, and UPDATE) or registered functions in the system catalog.

- **WORK | TRANSACTION**

  Specifies the optional keyword in the BEGIN syntax, which is meaningless.

- **ISOLATION LEVEL**

  Specifies the transaction isolation level, which is used to control data visibility between concurrent transactions.

  NOTE

  The isolation level of a transaction cannot be reset after the first clause (INSERT, DELETE, UPDATE, FETCH, COPY) for modifying data is executed in the transaction.

  Value range:

  - **READ COMMITTED**: Only committed data can be read; uncommitted data remains invisible. If no isolation level is specified, **READ COMMITTED** is used by default.
  - **READ UNCOMMITTED**: This isolation level behaves identically to **READ COMMITTED**.
  - **REPEATABLE READ**: Only the data committed before transaction start is read. Uncommitted data or data committed in other concurrent transactions cannot be read.
  - **SERIALIZABLE**: Currently, this isolation level is not supported. Setting this isolation level is equivalent to **REPEATABLE READ**.

- **READ WRITE | READ ONLY**

  Specifies the transaction access mode (**READ WRITE** or **READ ONLY**).

Examples

- Start a transaction.

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE tbl_test1(col1 int, col2 int);
gaussdb=# INSERT INTO tbl_test1 VALUES (1,1), (2,2), (3,3);

-- Start a transaction in default mode.
gaussdb=# BEGIN;
INSERT INTO tbl_test1 VALUES (4,4);
END;

-- Start a transaction with the separation level being REPEATABLE READ.
gaussdb=# BEGIN ISOLATION LEVEL REPEATABLE READ;
SELECT * FROM tbl_test1;
END;

-- Drop the table.
gaussdb=# DROP TABLE tbl_test1;
```

- Use anonymous blocks.

```
-- Generate a string using anonymous blocks.
gaussdb=# BEGIN
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
dbe_output.print_line('Hello');
END;
/
```

Helpful Links

START TRANSACTION

1.13.9 C

1.13.9.1 CALL

Description

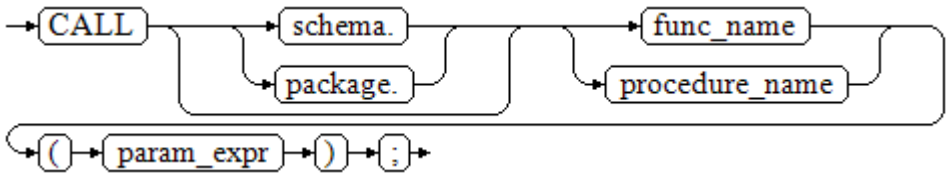
Calls defined functions and stored procedures.

Precautions

- Function or stored procedure owners, users with the EXECUTE permission on a function or stored procedure, or users with the EXECUTE ANY FUNCTION permission can call the function or stored procedure. When separation of duties is disabled, a system administrator has the permission to execute the **CALL** command by default.
- If the current database or pg_catalog is used, you can use database.schema.name to call functions or stored procedures. In this case, the objects called are those in the current database.

Syntax

```
CALL [ schema.|package.] { func_name | procedure_name } ( param_expr );
```



Parameters

- **schema.**

  Specifies the name of the schema where a function or stored procedure is located.

- **package.**

  Specifies the name of the package where a function or stored procedure is located.

- **func_name**

  Specifies the name of the function to be called.

  Value range: an existing function name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- You can use database links to perform operations on remote functions or stored procedures. For details about the usage and limitations, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.
- The function to be called can be an independent function or a member of a specified package or schema.
- When creating a function with the same name as a system function, you need to specify the schema of the function.

- **procedure_name**

  Specifies the name of the stored procedure to be called.

  Value range: character strings complying with **Identifier Naming Conventions**.

  NOTE

  - You can use database links to perform operations on remote functions or stored procedures. For details about the usage and limitations, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.
  - The stored procedure to be called can be an independent procedure or a member of a specified package or schema.

- **param_expr**

  List of parameters.

  You can use the naming tag method to pass parameters. That is, use **:=** or **=>** to separate a parameter name and its value. This method allows parameters to be placed in any order.

  If parameters are passed by values, that is, only parameter values are in the list, the value order must be the same as that defined in the function or stored procedure.

  Value range: an existing function parameter name or stored procedure parameter name

  NOTE

  - The parameters include input parameters (whose name and type are separated by **IN**) and output parameters (whose name and type are separated by **OUT**). When you run the **CALL** command to call a function or stored procedure, the parameter list must contain an output parameter for non-overloaded functions. You can set the output parameter to a variable or any constant. For an overloaded package function, the parameter list can have no output parameter, but the function may not be matched. If an output parameter is contained, it must be a constant.
  - Packages can be used only in ORA-compatible databases.

Examples

```
-- Create the func_add_sql function, calculate the sum of two integers, and return the result.
gaussdb=# CREATE FUNCTION func_add_sql(num1 integer, num2 integer) RETURN integer
AS
BEGIN
RETURN num1 + num2;
END;
/

-- Pass by parameter value.
gaussdb=# CALL func_add_sql(1, 3);

-- Pass by naming tag method.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CALL func_add_sql(num1 => 1,num2 => 3);
gaussdb=# CALL func_add_sql(num2 := 2, num1 := 3);

-- Drop the function.
gaussdb=# DROP FUNCTION func_add_sql;

-- Create a function with output parameters.
gaussdb=# CREATE FUNCTION func_increment_sql(num1 IN integer, num2 IN integer, res OUT integer)
RETURN integer
AS
BEGIN
res := num1 + num2;
END;
/

-- Specify a constant as an output parameter.
gaussdb=# CALL func_increment_sql(1,2,1);

-- Pass a variable as an output parameter.
gaussdb=# DECLARE
res int;
BEGIN
func_increment_sql(1, 2, res);
dbe_output.print_line(res);
END;
/

-- Drop the function.
gaussdb=# DROP FUNCTION func_increment_sql;

-- Create overloaded functions.
gaussdb=# CREATE OR REPLACE PROCEDURE package_func_overload(col int, col2 out int) PACKAGE
AS
DECLARE
    col_type text;
BEGIN
    col := 122;
    dbe_output.print_line('two out parameters ' || col2);
END;
/

gaussdb=# CREATE OR REPLACE PROCEDURE package_func_overload(col int, col2 out varchar) PACKAGE
AS
DECLARE
    col_type text;
BEGIN
    col2 := '122';
    dbe_output.print_line('two varchar parameters ' || col2);
END;
/

-- Call the function.
gaussdb=# CALL package_func_overload(1, 'test');
gaussdb=# CALL package_func_overload(1, 1);

-- Drop the function.
gaussdb=# DROP FUNCTION package_func_overload;
```

Helpful Links

CREATE FUNCTION and **CREATE PROCEDURE**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.9.2 CHECKPOINT

Description

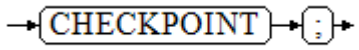
Sets WAL checkpoints. A checkpoint is a point in the transaction log sequence at which all data files have been updated to reflect the information in the log. All data files will be flushed to a disk. By default, WAL checkpoints are placed in the Xlog at intervals. You can use the **gs_guc** command to set related GUC parameters (**checkpoint_segments**, **checkpoint_timeout**, and **incremental_checkpoint_timeout**) to adjust the atomic checkpoint interval.

Precautions

- Only SYSADMIN and OPRADMIN can call CHECKPOINT.
- CHECKPOINT performs an immediate checkpoint when the related command is issued, without waiting for a regular checkpoint scheduled by the system.

Syntax

```
CHECKPOINT;
```



Examples

```
-- Set a checkpoint.
gaussdb=# CHECKPOINT;
```

1.13.9.3 CLEAN CONNECTION

Description

Clears network connections between the CN and other specified CNs or DNs. This function can clear idle or invalid connections of specified databases or users on CNs, and forcibly clear active connections.

Precautions

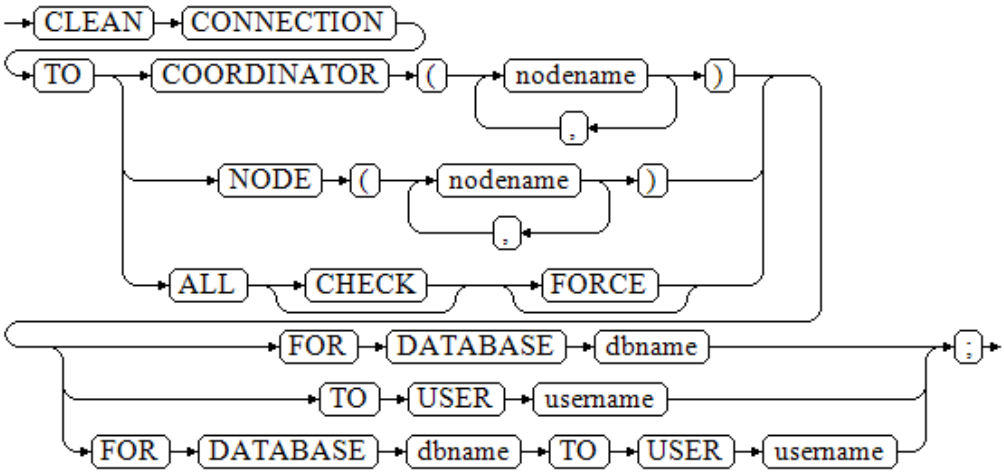
- You are advised to run the **CLEAN CONNECTION** command only when the database network connection is abnormal. You can query the pg_stat_get_pooler_status() function to view cached connections and check the cleanup effect.
- If the **FORCE** mode is not specified, the following results are obtained:
  - The CLEAN CONNECTION operation cleans up only the connections between database nodes (CNs or DNs) and does not affect the client connections.
  - In non-**FORCE** mode, the CLEAN CONNECTION operation cleans up only idle and invalid connections cached on CNs. Normal connections that are being used are not cleaned up.
- If the **FORCE** mode is specified, the following results are obtained:
  - The CLEAN CONNECTION operation forcibly cleans up normal connections and client connections that are being used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - The CLEAN CONNECTION operation cleans up the connections of a specified database or user on the DN. If residual connections cannot be cleared on DNs, O&M personnel are advised to use the pg_terminate_session function to clear invalid sessions.

Syntax

```
CLEAN CONNECTION
   TO { COORDINATOR ( nodename [, ... ] ) | NODE ( nodename [, ... ] )| ALL [ CHECK ] [ FORCE ] }
   { FOR DATABASE dbname | TO USER username | FOR DATABASE dbname TO USER username };
```



Parameters

- **CHECK**

  Checks connections before **DROP DATABASE** is executed. If any sessions are detected, an error will be reported and the database will not be dropped. This parameter can be used only when the node list is specified as **TO ALL**. If this parameter is used, the system checks whether the database is accessed by other sessions before cleaning up connections.

- **FORCE**

  Specifies the cleanup mode. If this parameter is specified, all threads related to the specified databases and users on the current CN will receive the SIGTERM signal, and the corresponding sessions will be forcibly closed, transactions will be aborted, and network connections will be cleaned up. All threads related to the specified databases and users on the DN will receive the SIGTERM signal and be forcibly closed.

  This parameter can be specified only when the node list is specified as **TO ALL**.

- **COORDINATOR (nodename [, ...] }) | NODE (nodename [, ...]) | ALL**

  Specifies a list of nodes. Idle and invalid connections between the current CN and the specified nodes will be cleaned up. The following scenarios exist:

  - **COORDINATOR**: deletes the idle or invalid connections from the current CN to a specified CN.
  - **NODE**: deletes the idle or invalid connections from the current CN to a specified DN.
  - **ALL**: deletes the idle or invalid connections from the current CN to all nodes, including CNs and DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: names of existing nodes.

- **dbname**

  Deletes connections to a specified database from the current CN. If this parameter is not specified, connections to all databases will be deleted by default.

  Value range: existing database names.

- **username**

  Deletes connections to a specified user from the current CN. If this parameter is not specified, connections of all users will be deleted by default.

  Value range: existing usernames.

Examples

```
-- Create the test_clean_connection database.
gaussdb=# CREATE DATABASE test_clean_connection;

-- Create user jack.
gaussdb=# CREATE USER jack PASSWORD '********';

-- Log in to the database as the user jack in another session, and query the connection information in the view.
gaussdb=# SELECT datname,usename,application_name,waiting,state
          FROM pg_stat_activity
          WHERE datname = 'test_clean_connection';
        datname         | usename | application_name | waiting | state
------------------------+---------+------------------+---------+-------
 test_clean_connection | jack    | gsql             | f       | idle
(1 row)

-- If you directly delete the test_clean_connection database, the following error information is displayed:
gaussdb=# DROP DATABASE test_clean_connection;
ERROR:  Database "test_clean_connection" is being accessed by other users. You can stop all connections by command: "clean connection to all force for database XXXX;" or wait for the sessions to end by querying view: "pg_stat_activity".
DETAIL:  There is 1 other session using the database.

-- Delete the connections between the current CN and all nodes of the test_clean_connection database.
-- If the FORCE parameter is not used, connections whose stat is not in other states cannot be deleted.
gaussdb=# CLEAN CONNECTION TO ALL FORCE FOR DATABASE test_clean_connection;

-- Query the connection for logging in to the test_clean_connection database.
gaussdb=# SELECT datname,usename,application_name,waiting,state
          FROM pg_stat_activity
          WHERE datname = 'test_clean_connection';
 datname | usename | application_name | waiting | state
---------+---------+------------------+---------+-------
(0 rows)

-- Drop the test_clean_connection database.
gaussdb=# DROP DATABASE test_clean_connection;

-- Drop user jack.
gaussdb=# DROP USER jack;
```

1.13.9.4 CLOSE

Description

Frees all resources associated with a specified cursor.

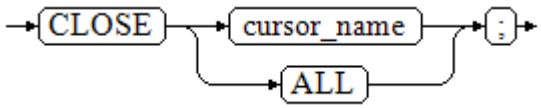
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

- A cursor that is no longer used should be closed in time to free the database resources it occupies and avoid resource leakage.
- After a cursor is closed, no subsequent operations are allowed on it.
- The behavior of closing a cursor is closely related to its holdability. The details are as follows:
  - **WITHOUT HOLD**: a cursor that is not holdable. When the transaction that creates the cursor ends with COMMIT or ROLLBACK, the cursor is automatically closed. This is an implicit behavior.
  - **WITH HOLD**: a cursor that is holdable.
    - If the transaction ends with ROLLBACK, the cursor is implicitly closed.
    - If the transaction is successfully committed, the cursor remains open and valid until an explicit CLOSE operation is performed or the client connection is terminated.
- GaussDB does not specify the OPEN statement for opening a cursor because a cursor is opened when it is defined using the **CURSOR** command. You can query the PG_CURSORS system view to obtain all available cursors.

Syntax

```
CLOSE { cursor_name | ALL } ;
```



Parameters

- **cursor_name**

  Specifies the name of a cursor to be closed or released.
- **ALL**

  Closes or releases all open cursors.

Examples

For details, see **Examples**.

Helpful Links

FETCH and **MOVE**

1.13.9.5 CLUSTER

Description

Clusters a table based on the specified index. The index must already exist in the target table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- In GaussDB, after a table is clustered, the table is physically stored based on the index information. Clustering is a one-time operation. After a table is updated, the updated content is not clustered again. Therefore, the system does not attempt to re-cluster the new content and updated records in index order.
- In GaussDB, after a table is clustered, the index used for clustering the table is recorded. When **CLUSTER** *table_name***;** is executed subsequently, the database automatically performs re-clustering based on the previously recorded index. You can run **ALTER TABLE** *table_name* **CLUSTER ON** *index_name***;** to specify the index used for subsequent clustering operations, or run **ALTER TABLE** *table_name* **SET WITHOUT CLUSTER;** to clear the index set for table clustering.
- Running **CLUSTER** without any parameters will re-cluster all previously clustered tables in the database owned by the current user. If this command is called by system administrators, all previously clustered tables will be re-clustered.
- In GaussDB, when a table is clustered, the database requests an ACCESS EXCLUSIVE lock on the table, preventing any other operations (including reads and writes) on the table until the CLUSTER operation is complete.

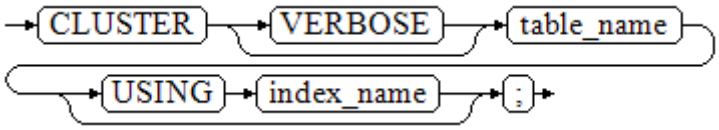
Precautions

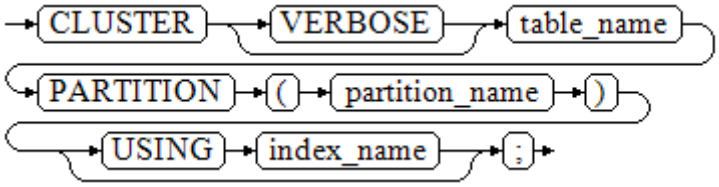
- Only row-store B-tree indexes can be used for clustering.
- When the table is randomly read, the data storage sequence has no significant impact. If it is accessed frequently based on a specific index and the index groups data, the CLUSTER operation can improve the query performance.
- When the query condition involves index-based range scan or the index key maps to multiple rows of data, the CLUSTER operation can effectively improve the query performance. If the index has identified the page storing the first matching row, other matching rows are easily located because they are usually stored on the same page, reducing disk I/O and accelerating query.
- When performing the CLUSTER operation on a table, GaussDB creates a temporary backup of the table created in the index sequence and a temporary backup of each index on the table. Therefore, it is crucial to ensure that the disk has sufficient free space (no less than the total size of the table and all indexes) during clustering.
- Optimizers rely on the sorting statistics of a table for query planning. After clustering a table, you must run **ANALYZE** to update the sorting statistics for optimizers. If the statistics are not updated in a timely manner, the optimizers may generate a non-optimal query execution plan based on outdated information.
- **CLUSTER** cannot be executed within a transaction.
- If the GUC parameter **xc_maintenance_mode** is not set to **on**, the CLUSTER operation will ignore all system catalogs.
- For a global secondary index (GSI):
  - After the **CLUSTER** command is executed on a single partition or the **CLUSTER [VERBOSE];** command is executed to recluster a table, an error will be reported if a query statement accesses the GSI. In this case, you need to run the **REINDEX INDEX** statement to rebuild the GSI to restore the query.

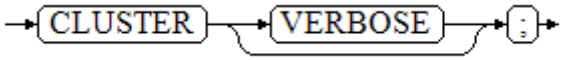
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - After **CLUSTER** is executed on a table with a GSI, you need to collect GSI statistics again by calling the **ANALYZE GLOBAL INDEX** *gsi_name* **FOR TABLE** *table_name* statement. For details, see **ANALYZE | ANALYSE**.
- To use an expression index for clustering, you must have the REINDEX permission on that index.

Syntax

- Cluster a table.
  ```
  CLUSTER [ VERBOSE ] table_name [ USING index_name ];
  ```
  

- Cluster a partition.
  ```
  CLUSTER [ VERBOSE ] table_name PARTITION ( partition_name ) [ USING index_name ];
  ```
  

- Recluster a table.
  ```
  CLUSTER [ VERBOSE ];
  ```
  

Parameters

- **VERBOSE**

  (Optional) Specifies whether to output the detailed progress information of the operation.
- **table_name**

  Specifies the name of the table to be clustered.

  Value range: existing table names.
- **USING index_name**

  Specifies the name of the index used for clustering a table or partition.

  Value range: existing index names.

  > NOTE
  >
  > This parameter must be specified for the initial CLUSTER operation on a table. When clustering a table again, you can omit **index_name**, and the database will automatically recluster the table based on the index recorded for previous clustering.
- **partition_name**

  Specifies the name of the partition to be clustered.

  Value range: existing partition names.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

- Cluster a table.

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE test_c1(id int, name varchar(20));
gaussdb=# CREATE INDEX idx_test_c1_id ON test_c1(id);
gaussdb=# INSERT INTO test_c1 VALUES (3,'Joe'),(1,'Jack'),(2,'Scott');

-- Query.
gaussdb=# SELECT * FROM test_c1;
 id | name
----+-------
  3 | Joe
  1 | Jack
  2 | Scott
(3 rows)

-- Perform clustering.
gaussdb=# CLUSTER test_c1 USING idx_test_c1_id;

-- Query.
gaussdb=# SELECT * FROM test_c1;
 id | name
----+-------
  1 | Jack
  2 | Scott
  3 | Joe
(3 rows)

-- Drop.
gaussdb=# DROP TABLE test_c1;
```

- Recluster a table.

```
-- Create a table.
gaussdb=# CREATE TABLE test(col1 int,CONSTRAINT pk_test PRIMARY KEY (col1));

-- An error is reported when the keyword USING is not contained in the first clustering.
gaussdb=# CLUSTER test;
ERROR:  there is no previously clustered index for table "test"

-- Perform clustering.
gaussdb=# CLUSTER test USING pk_test;

-- Insert data.
gaussdb=# INSERT INTO test VALUES (1),(99),(10),(8);

-- Recluster a table.
gaussdb=# CLUSTER VERBOSE test;
INFO:  clustering "public.test" using index scan on "pk_test"(dn_6001 pid=3672)
INFO:  "test": found 0 removable, 4 nonremovable row versions in 1 pages(dn_6001 pid=3672)
DETAIL:  0 dead row versions cannot be removed yet.
CPU 0.00s/0.00u sec elapsed 0.01 sec.
CLUSTER

-- Drop.
gaussdb=# DROP TABLE test;
```

- Cluster a partition.

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE test_c2(id int, info varchar(4)) PARTITION BY RANGE (id)(
    PARTITION p1 VALUES LESS THAN (11),
    PARTITION p2 VALUES LESS THAN (21)
);
gaussdb=# CREATE INDEX idx_test_c2_id1 ON test_c2(id);
gaussdb=# INSERT INTO test_c2 VALUES (6,'ABBB'),(2,'ABAB'),(9,'AAAA');
gaussdb=# INSERT INTO test_c2 VALUES (11,'AAAB'),(19,'BBBA'),(16,'BABA');

-- Query.
gaussdb=# SELECT * FROM test_c2;
 id | info
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
----+------
 6 | ABBB
 2 | ABAB
 9 | AAAA
11 | AAAB
19 | BBBA
16 | BABA
(6 rows)

-- Perform clustering on partition p2.
gaussdb=# CLUSTER test_c2 PARTITION (p2) USING idx_test_c2_id1;

-- Query.
gaussdb=# SELECT * FROM test_c2;
 id | info
----+------
 6 | ABBB
 2 | ABAB
 9 | AAAA
11 | AAAB
16 | BABA
19 | BBBA
(6 rows)

-- Drop.
gaussdb=# DROP TABLE test_c2;
```

1.13.9.6 COMMENT

Description

Creates or modifies the comment of an object.

Precautions

- Only one comment is stored for each object. To modify a comment, run the **COMMENT** command again for the same object. To drop a comment, set the comment text to **NULL**. When an object is dropped, its associated comment is automatically cleared.
- There is no security mechanism for viewing comments. Users connected to a database can view comments of all objects in the database. Comments of shared objects (such as databases, roles, and tablespaces) are stored globally, and users connected to any database can query the comments. Therefore, do not put security-critical information in comments.
- For most objects, only the object owner and users with the COMMENT permission are authorized to set comments. System administrators have this authorization by default.
- Roles do not have owners, so the rule for **COMMENT ON ROLE** is that you must be an administrator to comment on an administrator role, or have the CREATE ROLE permission to comment on non-administrator roles. System administrators can comment on all objects.

Syntax

```
COMMENT ON
{
  AGGREGATE agg_name (agg_type [, ...] ) |
  CAST (source_type AS target_type) |
  COLLATION object_name |
  COLUMN { table_name.column_name | view_name.column_name } |
```

```
  CONSTRAINT constraint_name ON table_name |
  CONVERSION object_name |
  DATABASE object_name |
  EXTENSION object_name |
  FOREIGN DATA WRAPPER object_name |
  FOREIGN TABLE object_name |
  FUNCTION function_name ( [ {[ argname ] [ argmode ] argtype} [, ...] ] ) |
  INDEX object_name |
   OPERATOR operator_name (left_type, right_type) |
  OPERATOR CLASS object_name USING index_method |
  OPERATOR FAMILY object_name USING index_method |
  [ PROCEDURAL ] LANGUAGE object_name |
  ROLE object_name |
  SCHEMA object_name |
  SERVER object_name |
  TABLE object_name |
  TABLESPACE object_name |
  TEXT SEARCH CONFIGURATION object_name |
  TEXT SEARCH DICTIONARY object_name |
  TEXT SEARCH PARSER object_name |
  TEXT SEARCH TEMPLATE object_name |
  TYPE object_name |
  VIEW object_name |
  TRIGGER trigger_name ON table_name
}
   IS 'text';
```

Parameters

- **agg_name**

  Specifies the new name of an aggregate function.
- **agg_type**

  Data type of the aggregate function parameters.
- **source_type**

  Specifies the source data type of the cast.
- **target_type**

  Specifies the target data type of the cast.
- **object_name**

  Specifies an object name.
- **table_name.column_name | view_name.column_name**

  Specifies a column name. You can prefix it with a table or view name.
- **constraint_name**

  Specifies the name of a table constraint.
- **table_name**

  Specifies a table name.
- **function_name**

  Function name.
- **argmode,argname,argtype**

  Specifies the schema, name, and type of the function parameters.
- **operator_name**

  Specifies the name of the operator.
- **left_type,right_type**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the data type (optionally schema-qualified) of GUC parameters. If the prefix or suffix operator does not exist, the **NONE** option can be added.

- **trigger_name**

  Specifies the trigger name.

- **text**

  Specifies the comment text.

Examples

```
-- Create a table.
gaussdb=# CREATE TABLE emp(
    empno varchar(7),
    ename varchar(50),
    job varchar(50),
    mgr varchar(7),
    deptno int
);

-- Add a comment to a table.
gaussdb=# COMMENT ON TABLE emp IS 'Department table';

-- Add comments to columns.
gaussdb=# COMMENT ON COLUMN emp.empno  IS  'Employee ID';
gaussdb=# COMMENT ON COLUMN emp.ename  IS  'Employee name';
gaussdb=# COMMENT ON COLUMN emp.job    IS  'Job';
gaussdb=# COMMENT ON COLUMN emp.mgr    IS  'Manager ID';
gaussdb=# COMMENT ON COLUMN emp.deptno IS  'Department ID';

-- View the comment of the table.
gaussdb=# \d+
 Schema |    Name     |   Type   | Owner |    Size    |              Storage               | Description
--------+-------------+----------+-------+------------+------------------------------------+-------------
 public | emp         | table    | omm   | 0 bytes    | {orientation=row,compression=no} | Department table

-- View the comments of columns.
gaussdb=# \d+ emp
                          Table "public.emp"
 Column |         Type          | Modifiers | Storage  | Stats target | Description
--------+-----------------------+-----------+----------+--------------+-------------
 empno  | character varying(7)  |           | extended |              | Employee ID
 ename  | character varying(50) |           | extended |              | Employee name
 job    | character varying(50) |           | extended |              | Job
 mgr    | character varying(7)  |           | extended |              | Manager ID
 deptno | integer               |           | plain    |              | Department ID
Has OIDs: no
Distribute By: HASH(empno)
Location Nodes: ALL DATANODES
Options: orientation=row, compression=no

-- Drop the emp table.
gaussdb=# DROP TABLE emp;
```

1.13.9.7 COMMIT | END

Description

Commits all operations of a transaction.

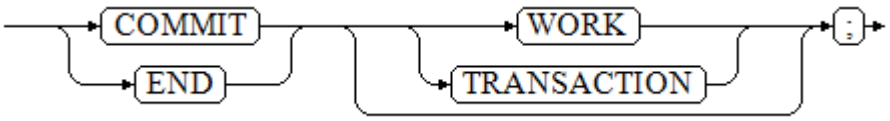
Precautions

Only the creator of a transaction or a system administrator can run the **COMMIT** command. The CREATE and COMMIT operations can be in different sessions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax

```
{ COMMIT | END } [ WORK | TRANSACTION ] ;
```



Parameters

- **COMMIT | END**

  Commits the current transaction and makes all changes made by the transaction become visible to others.

- **WORK | TRANSACTION**

  Optional keyword. It is only used for syntax compatibility and has no actual meaning.

Examples

```
-- Create a schema.
gaussdb=# CREATE SCHEMA tpcds;

-- Create a table.
gaussdb=# CREATE TABLE tpcds.customer_demographics_t2
(
    CD_DEMO_SK              INTEGER              NOT NULL,
    CD_GENDER               CHAR(1)                      ,
    CD_MARITAL_STATUS       CHAR(1)                      ,
    CD_EDUCATION_STATUS     CHAR(20)                     ,
    CD_PURCHASE_ESTIMATE    INTEGER                      ,
    CD_CREDIT_RATING        CHAR(10)                     ,
    CD_DEP_COUNT            INTEGER                      ,
    CD_DEP_EMPLOYED_COUNT   INTEGER                      ,
    CD_DEP_COLLEGE_COUNT    INTEGER
) DISTRIBUTE BY HASH (CD_DEMO_SK);

-- Start a transaction.
gaussdb=# START TRANSACTION;

-- Insert data.
gaussdb=# INSERT INTO tpcds.customer_demographics_t2 VALUES(1,'M', 'U', 'DOCTOR DEGREE', 1200,
'GOOD', 1, 0, 0);
gaussdb=# INSERT INTO tpcds.customer_demographics_t2 VALUES(2,'F', 'U', 'MASTER DEGREE', 300, 'BAD',
1, 0, 0);

-- Commit the transaction to make all changes permanent.
gaussdb=# COMMIT;

-- Query data.
gaussdb=# SELECT * FROM tpcds.customer_demographics_t2;

-- Delete the tpcds.customer_demographics_t2 table.
gaussdb=# DROP TABLE tpcds.customer_demographics_t2;

-- Drop the schema.
gaussdb=# DROP SCHEMA tpcds;
```

Helpful Links

ROLLBACK

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.9.8 COMMIT PREPARED

Description

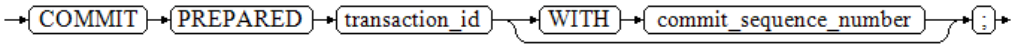
Commits a prepared two-phase transaction.

Precautions

- This functionality is for internal use only. You are advised not to use it.
- This function is available only in maintenance mode (when the GUC parameter **xc_maintenance_mode** is set to **on**). It is used only by maintenance personnel for troubleshooting. Common users should not use this mode.
- Only the creator of a transaction or a system administrator can run the command. The CREATE and COMMIT operations can be in different sessions.
- The transaction function is maintained automatically by the database, and should be not visible to users.

Syntax

```
COMMIT PREPARED transaction_id [ WITH commit_sequence_number ];
```



Parameters

- **transaction_id**

  Specifies the identifier of the transaction to be committed. The identifier cannot be the same as any one that has been used by any current prepared transaction.

- **commit_sequence_number**

  Sequence number of the transaction to be committed. The value is a 64-bit ascending unsigned number.

Examples

```
-- Start.
gaussdb=# SET XC_MAINTENANCE_MODE = ON;
SET

gaussdb=# BEGIN;

-- Create a table.
gaussdb=# CREATE TABLE item1(id int);

-- Prepare a transaction whose identifier is trans_test.
gaussdb=# PREPARE TRANSACTION 'trans_test';

-- Query the item1 table. The item1 table cannot be accessed.
gaussdb=# SELECT * FROM item1;
ERROR:  Relation "item1" does not exist on test.
LINE 1: SELECT * FROM item1;

-- Commit the transaction whose identifier is trans_test.
gaussdb=# COMMIT PREPARED 'trans_test';

-- Query the item1 table. The item1 table can be accessed.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT * FROM item1;
 a
---
(0 rows)

-- Drop the table.
gaussdb=# DROP TABLE item1;
```

Helpful Links

PREPARE TRANSACTION, **ROLLBACK PREPARED**

1.13.9.9 COPY

Description

Copies data between a table and a file.

Specifically, COPY FROM copies data from a file to a table, and COPY TO copies data from a table to a file.

Precautions

- When the **enable_copy_server_files** parameter is disabled, only the initial user can execute the **COPY FROM FILENAME** and **COPY TO FILENAME** commands. When the **enable_copy_server_files** parameter is enabled, users with the SYSADMIN permission or users inheriting permissions from the built-in role gs_role_copy_files can execute these commands. Nevertheless, to protect sensitive files from unauthorized access or modification, the use of the **COPY FROM FILENAME** and **COPY TO FILENAME** commands is restricted by default on database configuration files, key files, certificate files, environment variable files, and audit logs.
- The database administrator can set the GUC parameter **safe_data_path** to set the whitelist of paths that can be imported and exported by all users. The whitelist takes effect by default. If this parameter is left empty, the used paths are not intercepted, that is, the whitelist does not take effect. Exercise caution when setting this parameter. Note that an error will be reported when COPY uses a relative path of the designated path.
- COPY applies only to tables but not views.
- COPY TO requires the SELECT permission on the table to be read, and COPY FROM requires the INSERT permission on the table to be inserted.
- If a list of columns is specified, COPY will only copy the data in the specified columns to or from the file. If the table contains any columns that are not in the column list, COPY FROM will insert the default values for those columns.
- If a data source file is specified, the file must be accessible to the server. When **STDIN** is specified, data is transmitted via the connection between the client and the server. When entering data, use tab characters to separate the columns of the table and backslash-periods (\.) to indicate the end of the input.
- COPY FROM will raise an error if any row in the data file contains more or fewer columns than expected.
- The end of data can be represented by a single line containing just the backslash-period (\.). An end-of-data marker is not necessary when data is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

read from a file. It is needed when data is coped to or from client applications.

- In COPY FROM, **\N** is an empty string. To enter the actual value **\N**, use **\\N**.
- COPY FROM can preprocess data using column expressions, but column expressions do not support subqueries.
- If a data format error occurs during COPY FROM, the transaction will be rolled back. However, the error information provided is insufficient, making it difficult to locate the error data within a large amount of raw data.
- COPY FROM and COPY TO are suitable for low-concurrency, local export and import of small data volumes.
- When COPY is used in binary format, transcoding in distributed mode is not supported.
- **COPY** is a server command that runs within the same environment as the database server processes. **\COPY** is a client meta-command that runs within the same environment as the client tool gsql. Note that when the database and gsql are used in the sandbox environment, the **COPY** and **\COPY** commands both use the paths in the sandbox. When the database is used in the sandbox environment and gsql is used outside the sandbox, the **COPY** command uses the path inside the sandbox, and the **\COPY** command uses the path outside the sandbox.
- When executing COPY to import data to a base table with a GSI, the **enable_stream_operator** parameter must be enabled to achieve optimal data import performance.
- During the export using COPY TO, if the GUC parameter **support_zero_character** is disabled and the column data in the table contains the '\0' characters, the column data will be truncated during the export. Only the data before '\0' will be exported. When the parameter is enabled, the '\0' characters are also exported. The performance loss is positively correlated with the number of '\0' characters.
- Changing the schema of the pgxc_copy_error_log system catalog from public to pg_catalog deprecates the original catalog in the public schema. A table in the public schema may have the same name as a user table. Therefore, you need to identify the table. If the table is not a user table, migrate data to the corresponding table in the pg_catalog schema and delete the corresponding table in the original public schema. If a table with the same name exists in the pg_catalog schema, the GSQL meta-command **\d(+)** preferentially matches pg_catalog.pgxc_copy_error_log, which is not displayed because it is a system catalog. As a result, pgxc_copy_error_log cannot be found in the list, but this table exists in the public schema. You can directly reference public.pgxc_copy_error_log.
- When exporting data of the float4 or float8 type, you are advised to set **extra_float_digits** to **3** to avoid loss of significant digits and ensure consistency between the data before and after import and export.
- In certain scenarios, you can set the GUC parameter **batch_insert_index_types** to **"rcr_ubtree"** to improve the performance of inserting indexes in batches. For details, see **Batch insertion index performance optimization** in **INSERT**.
- For security purposes, do not use COPY FROM to import data to the pg_authid, pg_auth_history, gs_global_config, and gs_workload_rule system catalogs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- When the parameter **enable_security_policy** is enabled, and if a masking policy has been set, the system will perform data masking.
- When importing and exporting data, check whether the data contains escape characters. If you prepare the data file, you need to assess the import mode (import with or without escaping). If the data file is exported using the **COPY** command, the escape option set for import must be the same as that for export.
- When the **COPY** command is used to import or export data, the size of a single row of data tuples cannot exceed 1 GB – 1 byte in non-transcoding scenarios, and cannot exceed 256 MB – 1 byte in transcoding scenarios.
- When the **COPY** command is used to import data, error tables are involved. For details about the precautions and usage, see the corresponding sections. This involves the system view pg_catalog.pgxc_copy_error_log and the system catalog pg_catalog.gs_copy_error_log.
- During COPY FROM, data is inserted by DNs, and constraint exceptions are reported by DNs to the CN. Each DN can only view its own data. Therefore, the error line number reported by a DN to the CN is different from that in the data file.
- When you run the **\COPY** command with **detail_info** as a condition to query audit information, the command operates in a case-insensitive manner. Specifically, the SQL statement is converted to and displayed as the COPY keyword in audit information.
- Set-returning functions cannot be used in column expressions with the **COPY** command.
- Invalid characters cannot be imported as they were when they are exported using the COPY command and imported again.

Syntax

- Copy data from a file to a table.

```
COPY [BINARY] table_name [ ( column_name [, ...] ) ]
    [ WITH OIDS ]
    FROM { 'filename' | STDIN }
    [ LOAD_DISCARD 'discard_file_name' ]
    [ LOAD_BAD 'bad_file_name' ]
    [ USEEOF ]
    [ [ USING ] DELIMITERS 'delimiters' ]
    [ WITHOUT ESCAPING ]
    [ LOG ERRORS | LOG ERRORS DATA ]
    [ REJECT LIMIT 'limit' ]
    [ WITH ]
    [ copy_option [ ...] | ( option [, ...] ) ];
```

- Copy data from a table to a file.

```
COPY table_name [ ( column_name [, ...] ) ]
    [ WITH OIDS ]
    TO { 'filename' | STDOUT }
    [ [ USING ] DELIMITERS 'delimiters' ]
    [ WITHOUT ESCAPING ]
    [ WITH ]
    [ copy_option [ ...] | ( option [, ...] ) ];

COPY query {(SELECT) | (VALUES)}
    TO { 'filename' | STDOUT }
    [ WITHOUT ESCAPING ]
    [ WITH ]
    [ copy_option [ ...] | ( option [, ...] ) ];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The syntax constraints of COPY TO are as follows:

  (query) is incompatible with [USING] DELIMITERS. If the data comes from a query result, COPY TO cannot specify [USING] DELIMITERS.
- Use spaces to separate **copy_option** following FIXED FORMATTER.
- **copy_option** is a native parameter form in COPY, whereas **option** is a parameter form imported from a compatible foreign table.

copy_option can be as follows:

```
{ OIDS
| DELIMITER [ AS ] 'delimiter_string'
| NULL [ AS ] 'null_string'
| HEADER
| FILEHEADER 'header_file_string'
| FREEZE
| FORCE NOT NULL column_name [, ...]
| FORCE QUOTE { column_name [, ...] | * }
| BINARY
| CSV
| FIXED
| QUOTE [ AS ] 'quote_character'
| ESCAPE [ AS ] 'escape_character'
| EOL 'newline_character'
| ENCODING 'encoding_name'
| IGNORE_EXTRA_DATA
| FILL_MISSING_FIELDS [ { 'one' | boolean } ]
| COMPATIBLE_ILLEGAL_CHARS
| DATE_FORMAT 'date_format_string'
| TIME_FORMAT 'time_format_string'
| TIMESTAMP_FORMAT 'timestamp_format_string'
| DATEA_FORMAT 'datea_format_string'
| SMALLDATETIME_FORMAT 'smalldatetime_format_string'
| COPY_CUSTOM_ID 'custom_id_string'
| TABLE_COMPRESS_CLAUSE( ROW STORE COMPRESS ADVANCED [ MEDIUM | HIGH ] ROW [ON
(EXPR)])
| FORMATTER ( [ column_name( offset, length ) ] [, ...] )
| TRANSFORM ( [ column_name [ data_type ] [ AS transform_expr ] ] [, ...] )
| POSITION_KEEP_TRAILING_SPACE }
```

option can be as follows:

```
{ FORMAT 'format_name'
| OIDS [ boolean ]
| DELIMITER 'delimiter_character'
| NULL 'null_string'
| HEADER [ boolean ]
| USEEOF [ boolean ]
| FILEHEADER 'header_file_string'
| FREEZE [ boolean ]
| QUOTE 'quote_character'
| ESCAPE 'escape_character'
| EOL 'newline_character'
| NOESCAPING [ boolean ]
| FORCE_QUOTE { ( column_name [, ...] ) | * }
| FORCE_NOT_NULL ( column_name [, ...] )
| ENCODING 'encoding_name'
| IGNORE_EXTRA_DATA [ boolean ]
| FILL_MISSING_FIELDS [ { 'one' | boolean } ]
| COMPATIBLE_ILLEGAL_CHARS [ boolean ]
| DATE_FORMAT 'date_format_string'
| TIME_FORMAT 'time_format_string'
| TIMESTAMP_FORMAT 'timestamp_format_string'
| DATEA_FORMAT 'datea_format_string'
| SMALLDATETIME_FORMAT 'smalldatetime_format_string'
| COPY_CUSTOM_ID 'custom_id_string'
| POSITION_KEEP_TRAILING_SPACE }
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **query**

  Specifies that the results will be copied.

  Value range: Only one SELECT or VALUES command is supported. A semicolon (;) is not required at the end of the command.

- **table_name**

  Specifies a table name (optionally schema-qualified).

  Value range: existing table names.

- **column_name**

  Specifies an optional list of columns to be copied.

  Value range: existing columns. If no column list is specified, all columns will be used.

- **STDIN**

  Specifies that input comes from the standard input. In the input, table columns are separated by tab characters, and the end of data is represented by a new row containing only the backslash-period (\.).

  NOTE

  In parallel import mode, if data is imported through standard input (STDIN), the client sends data using multiple threads, which leads to lock contention. This may cause transaction commit failures in some threads, resulting in only partial data being imported. Additionally, the failed data cannot be re-imported subsequently. Therefore, it is not recommended to use standard input for parallel import when the table already contains data.

- **STDOUT**

  Specifies that output goes to the standard output.

- **LOAD_DISCARD 'discard_file_name'**

  This API is reserved and not recommended.

- **LOAD_BAD 'bad_file_name'**

  This API is reserved and not recommended.

- **USEEOF**

  If this parameter is set, no error is reported for a backslash followed by a period (\.). When importing a single column, if you enter "\." in **stdin** mode, it is considered as an end of line (EOL) character instead of data.

  Value range: **true/on** and **false/off**.

  Default value: **false**.

- **[USING] DELIMITERS 'delimiters'**

  String that separates columns within each row (line) of the file. It cannot be larger than 10 bytes.

  Value range:

  – The delimiter in text format cannot include any of the following characters: \.abcdefghijklmnopqrstuvwxyz0123456789, but has no restriction for the CSV format.

  – The delimiter cannot be set to **'\r'**, **'\n'**, or a user-defined newline character.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - A delimiter cannot contain the null string (refer to **NULL null_string**), nor can it be contained within the null string.
  - A delimiter cannot contain \0.
  - A delimiter cannot be an empty string.
  - When **COMPATIBLE_ILLEGAL_CHARS** is specified (see **COMPATIBLE_ILLEGAL_CHARS**), a delimiter cannot be set to a space or a question mark (?); otherwise, the delimiter may conflict with the characters after error tolerance conversion.
  - In CSV mode, a delimiter cannot contain QUOTE characters. For details, see **QUOTE [AS] 'quote_character'**.
  - The value of **delimiter** can be a multi-character delimiter whose length is less than or equal to 10 bytes.

  Value range: The default value is a tab character in text format and a comma in CSV format.

  NOTE

  - Both DELIMITER and DELIMITERS can specify delimiters. DELIMITERS can be followed by brackets, but DELIMITER cannot be directly followed by brackets. Otherwise, an error is reported.
  - Delimiters can be specified only in TEXT or CSV mode.
  - Within the data file, each row must not exceed 1 GB in length. Using long delimiters or having many columns may reduce the valid data length that can be exported.
  - You can use multi-character delimiters or invisible delimiters. For example, you can use multi-characters (such as $^&) and invisible characters (such as 0x07, 0x08, and 0x1b).
  - If a column to be exported in text format contains data that is the same as separators, a backslash (\) is added as an escape character before a conflicting data record, ensuring that the exported data can be properly identified.
  - You are advised not to import or export delimiters that use Chinese or other characters containing multiple bytes. Otherwise, if the data in a column is the same as the delimiter, an error may be reported after the data is imported or exported.
  - To use a multi-byte delimiter, you are advised to combine single-byte characters within the ASCII code range.

- **WITHOUT ESCAPING**

  Specifies, in the TEXT format, whether to escape the backslash (\) and its following characters.

  Value range: text only

  NOTE

  For options related to escape characters, pay attention to the following:

  - If you prepare a data file, you need to choose the desired import behavior (import with or without escaping).
  - If a data file is exported using the kernel COPY syntax, the escape options must be the same for both export and import.
  - If data is exported in text format, in the default scenario (WITHOUT ESCAPING is not used), the characters '\b', '\f', '\n', '\r', '\t', and '\v' in a column are exported as two bytes—a backslash (\) and the corresponding letter.

- **LOG ERRORS**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies whether to enable the error tolerance mechanism for data type errors in the COPY FROM statement.

Value range: a value set while data is imported using COPY FROM.

NOTE

When using this error tolerance mechanism, pay attention to the following limitations:

- This error tolerance mechanism captures only the data type errors (**DATA_EXCEPTION**) that the primary database node identifies from the data parsed during COPY FROM.
- If the COPY statement already has an error tolerance option enabled (for example, **IGNORE_EXTRA_DATA**), that type of error will be processed accordingly and trigger no error. Therefore, the error table does not contain related data.
- This error tolerance mechanism does not support constraint conflicts.
- During a rolling upgrade from an earlier version to this version, do not use the COPY error tolerance import capability before all nodes are upgraded.
- If separation of duties is enabled, you need to run the following statement to create an error table in a schema with the same name as the user before using the fault tolerance mechanism (for details, see **GS_COPY_ERROR_LOG**):

```
CREATE TABLE load_user.pgxc_copy_error_log (relname varchar(64), begintime timestamptz, filename varchar, lineno int8, rawrecord text, detail text);
CREATE INDEX copy_error_log_relname_idx ON load_user.pgxc_copy_error_log(relname);
```

- You are advised to set the **backtrace_min_messages** parameter to a value higher than the **error** level. Otherwise, the performance will be affected.

- **LOG ERRORS DATA**

  The differences between LOG ERRORS DATA and LOG ERRORS are as follows:

  a. **LOG ERRORS DATA** fills the **rawrecord** column in the error tolerance table.

  b. Only users with the SYSADMIN permission can use **LOG ERRORS DATA**.

  NOTE

  - If error content is too complex, it may fail to be written to the error tolerance table by using **LOG ERRORS DATA**, causing the task failure.
  - For errors that cannot be read in certain code, the error codes are ERRCODE_CHARACTER_NOT_IN_REPERTOIRE and ERRCODE_UNTRANSLATABLE_CHARACTER. The rawrecord column is not recorded.
  - In batch insertion scenarios, the **rawrecord** column is not recorded for error data of the PRIMARY KEY constraint.
  - If data is imported in BINARY format and error data exists, the corresponding **rawrecord** column is not recorded in the error tolerance table.
  - During a rolling upgrade from an earlier version to this version, do not use the COPY error tolerance import capability before all nodes are upgraded.

- **REJECT LIMIT 'limit'**

  Used with the **LOG ERRORS** option to set the upper limit of the tolerated errors in the COPY FROM statement. If the number of errors exceeds the limit, any further errors will be reported based on the original mechanism.

  Value range: a positive integer in the range [1,2147483647] or **'unlimited'**

  Default value: If **LOG ERRORS** is not specified, an error will be reported. If LOG ERRORS is specified, the default value is **0**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> In the error tolerance mechanism described in the description of LOG ERRORS, the count of **REJECT LIMIT** is calculated based on the number of data parsing errors on the primary database node where the **COPY FROM** statement is executed, not based on the number of all errors on the node.

- **copy_option**

  Specifies various native parameters for **COPY**.

  - OIDS

    Specifies whether to import and export OID hidden columns for tables (usually system catalogs) that contain OIDs.

    Value range: **true/on** and **false/off**.

    Default value: **false**.

    > **NOTE**
    >
    > - If the table to be exported does not have an OID, an error is reported.
    > - The exported OID defaults to the first column. If the **HEADER** option is specified, the exported **HEADER** row does not include the OID column description.

  - DELIMITER [AS] 'delimiter_string'

    Specifies the column delimiter of data in a data file line. The value range is the same as that of **[USING] DELIMITERS 'delimiters'**.

  - NULL [AS] 'null_string'

    Specifies the string that represents a null value.

    Value range:

    - The **NULL** value cannot contain '\r', '\n', or user-defined newline characters.
    - The **NULL** value contains a maximum of 100 characters.
    - **NULL** cannot contain delimiters (see **[USING] DELIMITERS 'delimiters'**) or be contained by delimiters.
    - In CSV mode, **NULL** cannot contain QUOTE characters. For details, see **QUOTE [AS] 'quote_character'**.
    - When **COMPATIBLE_ILLEGAL_CHARS** is specified (see **COMPATIBLE_ILLEGAL_CHARS**), **NULL** cannot be set to a space or a question mark (?); otherwise, it may conflict with the characters after error tolerance conversion.
    - The **NULL** value cannot contain the '\0' character.

    Default value:

    - Unquoted empty string in CSV format.
    - **\N** in text format.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The **NULL** parameter can be specified only in TEXT or CSV mode.
- During COPY FROM, any data item that matches this string will be stored as a null value; therefore, ensure that you use the same string as you used with COPY TO.

– HEADER

Specifies whether the exported data file contains a header row and whether the first row in the data stream is the header row during import. A header row describes information about each column.

NOTE

- This parameter can be specified only in CSV or FIXED mode. The content in the header row of the FIXED file must meet the column length definition.
- When this option is specified, the first row of the data text is identified as a header row during import and will be ignored. The first row in the data file exported is the header row.
- If this option is not specified, the first row of the data text is identified as data and imported. The exported data file does not contain a header row.

– FILEHEADER 'header_file_string'

Specifies a file that defines the content in the header for exported data. The file contains data description of each column.

NOTICE

- This option is valid only when the header functionality is enabled.
- **fileheader** specifies an absolute path.
- The file can contain only one row of header information, and ends with a newline character. Excess rows will be discarded. (Header information cannot contain newline characters.)
- The file, including newline characters, cannot exceed 1 MB.

– FREEZE

Sets the COPY loaded data row as **frozen**, like these data have executed VACUUM FREEZE.

This is a performance option of initial data loading. The data will be frozen only when the following three requirements are met:

- The table being loaded has been created or truncated in the same transaction before copying.
- There are no cursors open in the current transaction.
- There are no original snapshots in the current transaction.

NOTE

When COPY is completed, all the other sessions will see the data immediately. However, this violates the general principle of MVCC visibility, and users should understand that this may cause potential risks.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- FORCE NOT NULL column_name [, ...]

  Forbids null values in the specified columns in CSV COPY FROM mode. Empty inputs will be regarded as zero-length strings.

  Value range: existing columns.

- FORCE QUOTE { column_name [, ...] | * }

  In CSV COPY TO mode, forces all non-**NULL** values to be surrounded by the quote characters (see **QUOTE [AS] 'quote_character'**) around each specified column. The asterisk (*) indicates all columns. **NULL** values are not enclosed in quote characters.

  Value range: existing columns.

- BINARY

  Specifies that data is stored and read in binary rather than text format.

  NOTE

  - In binary mode, you cannot declare **DELIMITER**, **NULL**, **CSV**, or **EOL**.
  - When **BINARY** is specified, **CSV**, **FIXED**, and **TEXT** cannot be specified through **option** or **copy_option**.
  - If the GUC parameter **support_binary_copy_version** is set to **'header_encoding'**, the header of the binary file exported using COPY TO will include the server-side encoding of the database. When the binary file is imported using COPY FROM, verification will be conducted to ensure that the encoding in the file is consistent with the server-side encoding.
  - If the GUC parameter **copy_special_character_version** is set to **'no_error'**, invalid characters will not be checked during the import and will be displayed as garbled characters in query results. The database server-side encoding must be consistent with the client-side encoding. Enable this parameter with caution only when you fully understand its implications.
  - In binary mode, **copy_special_character_version** is set to **'no_error'**, and it takes effect only for columns of the TEXT, CHAR, VARCHAR, NVARCHAR2, and CLOB types.
  - Exporting data to STDOUT or importing from STDIN is not supported.

  The following table lists the types of data that cannot be imported or exported in binary mode.

smgr	aclitem	gtsvector	any	trigger
language_handler	internal	opaque	anyelement	anynesttable
anyindexbytable	anynonarray	anyenum	fdw_handler	anyrange
hll_hashval	hash16	hash32	anyset	-

  The following table lists some options that do not support binary import and export.

DELIMITER	NULL	EOL	CSV	FIXED

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

SMALLDATETIME_FORMAT	DATE_FORMAT	TIME_FORMAT	TIMESTAMP_FORMAT	DATEA_FORMAT

– CSV

Enables the CSV mode. When **CSV** is specified, **BINARY**, **FIXED**, and **TEXT** cannot be specified through **option** or **copy_option**.

NOTE

The most common error in the CSV format is caused by the QUOTE characters, which are generally double quotation marks. If the data column contains such characters, escape them. Otherwise, COPY cannot parse the incorrect data format.

– FIXED

Fixes column length. When the column length is fixed, **DELIMITER**, **NULL**, and **CSV** cannot be specified. When **FIXED** is specified, **BINARY**, **CSV**, and **TEXT** cannot be specified by **option** or **copy_option**. Newline characters in data columns cannot be correctly processed when the column length is fixed. **FORMATTER** must be specified.

NOTE

- The definition of fixed length is as follows:
  - The column length of each record is the same.
  - Spaces are used for column padding. Columns of the numeric type and fixed-length columns of the string type are left-aligned, and variable-length columns of the string type are right-aligned.
  - No delimiters are used between columns.
- You are advised not to perform encoding conversion during import or export of fixed-length data. This is because encoding conversion changes the byte length of characters, causing incorrect truncation and parsing of columns in fixed-length format. In this case, you need to adjust the start and end positions specified in fixed-length columns or use other file formats of COPY.
- When **FIXED** is used, all columns in the table must be used. You cannot specify only some columns.

– FORMATTER ([column_name (offset, length)] [, ...])

Defines the place of each column in the data file only in fixed length mode. Defines the place of each column in the data file in the **column(***offset*,*length***)** format.

Value range:

- The value of **offset** must be larger than 0. The unit is byte.
- The value of **length** must be larger than 0. The unit is byte.

The total length of all columns must be less than 1 GB.

All columns must not overlap with each other.

Replace columns that are not in the file with null.

– QUOTE [AS] 'quote_character'

Specifies the quote character for a CSV file.

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- A single-byte character without '\0', '\r', '\n', or end-of-line characters. You are advised to set **quote** to an invisible character, such as **0x07**, **0x08**, or **0x1b**.
- The value cannot contain delimiters (see **[USING] DELIMITERS 'delimiters'**) or be contained by delimiters.
- The value cannot contain **NULL** (see **NULL [AS] 'null_string'**) or be included in **NULL**.
- When **COMPATIBLE_ILLEGAL_CHARS** is specified (see **COMPATIBLE_ILLEGAL_CHARS**), **NULL** cannot be set to a space or a question mark (?); otherwise, it may conflict with the characters after error tolerance conversion.

Default value: double quotation marks ("").

– ESCAPE [AS] 'escape_character'

Specifies the escape character in CSV format.

Value range:

- A single-byte character without '\0'. You are advised to set **ESCAPE** to an invisible character, such as **0x07**, **0x08**, or **0x1b**.
- The value cannot contain delimiters (see **[USING] DELIMITERS 'delimiters'**) or be contained by delimiters.
- The value cannot contain **NULL** (see **NULL [AS] 'null_string'**) or be included in **NULL**.
- When **COMPATIBLE_ILLEGAL_CHARS** is specified (see **COMPATIBLE_ILLEGAL_CHARS**), **NULL** cannot be set to a space or a question mark (?); otherwise, it may conflict with the characters after error tolerance conversion.
- If the quote character is set (see **QUOTE [AS] 'quote_character'**) but ESCAPE is not set, ESCAPE uses the same character as the QUOTE character.

Default value: double quotation marks ("").

– EOL 'newline_character'

Specifies the end-of-line character (newline character) style of the imported or exported data file.

Value range: Common end-of-line characters include **\r** (0x0D), **\n** (0x0A), and **\r\n** (0x0D0A). Special end-of-line characters include **$** and **#**.

- Value range: multi-character end-of-line characters within 10 bytes.
- The end-of-line character cannot be contained in the delimiter.
- The end-of-line character cannot be contained in a **NULL** string.
- In non-CSV mode, the end-of-line character cannot contain the following characters: .abcdefghijklmnopqrstuvwxyz0123456789.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- An end-of-line character cannot contain '\0'.
- Parallel import is not supported when a custom newline character is specified.

NOTE

The **EOL** parameter supports only the TEXT format for data import and export and does not support the CSV or FIXED format for data import. For compatibility with earlier versions, the **EOL** parameter can be set to **0x0A** or **0x0D0A** for data export in the CSV or FIXED format.

– ENCODING 'encoding_name'

Specifies the name of a file encoding format.

Value range: a valid encoding format. Common encoding formats are **utf8**, **gb18030**, and **gbk**.

Default value: *current client-side encoding format*.

NOTE

- After the **encoding** parameter is specified, the data from the data file or standard input is identified based on the specified encoding, and whether encoding conversion is required is determined based on the value of the server-side encoding.
- In BINARY format, the **encoding** parameter does not take effect. The conversion between the server-side encoding and client-side encoding depends on the settings of **server_encoding** and **client_encoding**.
- When COPY FROM is executed, the encoding format specified by **ENCODING** in the control file must be the same as that of the file. Otherwise, an error is reported or garbled characters are displayed in the imported data.

– IGNORE_EXTRA_DATA

Specifies that when the number of data source files exceeds the number of foreign table columns, excess columns at the end of the row are ignored. This parameter is used only during data import.

If this parameter is not used and the number of columns in the data source file is greater than that defined in the foreign table, the following error information is displayed:

```
extra data after last expected column
```

– POSITION_KEEP_TRAILING_SPACE

Specifies whether to remove spaces at the end of a column value during fixed-length import. If **POSITION_KEEP_TRAILING_SPACE** is not configured, spaces at the end of a column value are removed by default. If **POSITION_KEEP_TRAILING_SPACE** is configured, spaces at the end of a column value are retained. This parameter takes effect only for the character type and is invalid for the numeric and BLOB types.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

During data import, the system reads raw data from the data file, and then performs operations such as list expression calculation and data type conversion.

The behavior of trimming spaces at the end of the data occurs only in the data reading phase. This parameter takes effect only in this phase. Once the data is read, subsequent operations such as list expression processing and type conversion will not be affected by this parameter if they involve trimming spaces at the end of the data.

For example, if a column is defined as the char type in the control file, but the corresponding column in the table is of the varchar type, the spaces at the end of the data are automatically trimmed when the data is converted from the char type to the varchar type. This behavior is not affected by the preceding parameter. To retain the spaces at the end of the data in such type conversion scenarios, you can set the **behavior_compat_options** parameter to include the **char_coerce_compat** option.

- COMPATIBLE_ILLEGAL_CHARS

  Specifies whether to tolerate invalid characters during data import and export.

  Value range: **true/on** and **false/off**.

  - **true**/**on**: No error message is reported and data import is not interrupted when there are invalid characters. Invalid characters are converted into valid ones, and then imported to the database. Invalid characters are converted into valid ones, and then exported from the database. No error is reported and data export is not interrupted.
  - **false**/**off**: An error occurs when there are invalid characters, and the import stops. An error is reported and data export is interrupted when there are invalid characters and characters whose encodings do not exist.

  Default value: **false**/**off**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

The rules for converting invalid characters are as follows:

1. '\0' is converted to a space. When **support_zero_character** is enabled, data is imported in the original format.
2. Other invalid characters are converted to the characters specified by the GUC parameter **convert_illegal_char_mode**. The default value is **'?'**.
3. When **compatible_illegal_chars** is set to **true** or **on**, after invalid characters such as **NULL**, **DELIMITER**, **QUOTE**, and **ESCAPE** are converted to spaces or question marks, an error message stating "illegal chars conversion may confuse COPY escape 0x20" will be displayed to remind you of possible parameter confusion caused by the conversion.
4. If **compatible_illegal_chars** is set to **true** or **on**, data in binary format cannot be imported. The error message "cannot specify bulkload compatibility options in BINARY mode" is displayed.
5. If the GUC parameter **copy_special_character_version** is set to **'no_error'**, invalid characters will not be checked during the import and will be displayed as garbled characters in query results. Enable this parameter with caution only when you fully understand its implications.
6. If **copy_special_character_version** is set to **'no_error'** and **compatible_illegal_chars** is set to **true**, the former has a higher priority. That is, invalid characters will not be checked during the import.
7. If **copy_special_character_version** is set to **'no_error'** or **compatible_illegal_chars** is set to **true**, a record is written to the database run logs each time a row of data containing invalid characters is identified. The log writing behavior occupies a small amount of disk bandwidth. To eliminate the impact, set the GUC parameter **enable_log_copy_illegal_chars** to **off**, that is, do not write records to the database run logs. You are advised not to modify this configuration unless otherwise required.

The rules for exporting invalid characters are as follows:

The export is classified into the scenario where transcoding is required (the database server code is different from the client code) and the scenario where transcoding is not required (the database server code is the same as the client code). The export performance varies according to the scenario. You can run **show server_encoding;** to query the database server-side encoding and **show client_encoding;** to query the database client-side encoding.

- In scenarios where transcoding is required:

1. '\0' is converted to a space. When **support_zero_character** is enabled, data is exported in the original format.
2. Other invalid characters are converted to the characters specified by the GUC parameter **convert_illegal_char_mode**. The default value is **'?'**.
3. When **compatible_illegal_chars** is set to **true** or **on**, after invalid characters such as **NULL**, **DELIMITER**, **QUOTE**, and **ESCAPE** are exported to spaces or question marks, an error message stating "illegal chars conversion may confuse COPY escape 0x20" will be displayed to remind you of possible parameter confusion caused by the export.
4. If **compatible_illegal_chars** is set to **true** or **on**, data in binary format cannot be exported.
5. If **compatible_illegal_chars** is set to **true**, a record is written to the database run logs each time a row of data containing invalid characters is identified. The log writing behavior occupies a small amount of disk bandwidth. To eliminate the impact, set the GUC parameter **enable_log_copy_illegal_chars** to **off**, that is, do not write records to the database run logs. You are advised not to modify this configuration unless otherwise required.

- In scenarios where transcoding is not required:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1. Data is exported as it is in the database, even if the data contains invalid characters and **compatible_illegal_chars** is set to **true**.
2. If **compatible_illegal_chars** is set to **true** or **on**, data in binary format cannot be exported.

- FILL_MISSING_FIELDS [{ 'one' | boolean }]

  Specifies how to handle the problem that a data row loses the last or multiple columns during data import. If only **FILL_MISSING_FIELDS** is specified but the option value is not as expected, or the specified option value is **one**, **true**, **on**, **null** is used to supplement the last missing column (an error is reported if multiple columns are missing).

  Value range: **one**, **true**/**on**, or **false**/**off**.

  Default value: **false**/**off**.

- DATE_FORMAT 'date_format_string'

  Specifies the DATE format for data import. This parameter is not supported for the binary format, as it would trigger the error "cannot specify bulkload compatibility options in BINARY mode." This parameter takes effect only during data import using COPY FROM.

  Value range: valid DATE formats. For details, see **Date and Time Processing Functions and Operators**.

- TIME_FORMAT 'time_format_string'

  Specifies the TIME format for data import. This parameter is not supported for the binary format, as it would trigger the error "cannot specify bulkload compatibility options in BINARY mode." This parameter takes effect only during data import using COPY FROM.

  Value range: a valid TIME value. Time zones cannot be used. For details, see **Date and Time Processing Functions and Operators**.

- TIMESTAMP_FORMAT 'timestamp_format_string'

  Specifies the TIMESTAMP format for data import. This parameter is not supported for the binary format, as it would trigger the error "cannot specify bulkload compatibility options in BINARY mode." This parameter takes effect only during data import using COPY FROM.

  Value range: a valid TIMESTAMP value. Time zones cannot be used. For details, see **Date and Time Processing Functions and Operators**.

- DATEA_FORMAT 'datea_format_string'

  Specifies the DATEA format for data import. This parameter is not supported for the binary format, as it would trigger the error "cannot specify bulkload compatibility options in BINARY mode." This parameter takes effect only during data import using COPY FROM.

  Value range: a valid DATEA value. For details, see **Date and Time Processing Functions and Operators**.

- SMALLDATETIME_FORMAT 'smalldatetime_format_string'

  Specifies the SMALLDATETIME format for data import. This parameter is not supported for the binary format, as it would trigger the error "cannot specify bulkload compatibility options in BINARY mode." This parameter takes effect only during data import using COPY FROM.

  Value range: a valid SMALLDATETIME value. For details, see **Date and Time Processing Functions and Operators**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- TABLE_COMPRESS_CLAUSE(ROW STORE COMPRESS ADVANCED [MEDIUM | HIGH] ROW [ON (EXPR)])

  Compresses the imported data. EXPR is an expression. Data that meets the expression is compressed, and data that does not meet the expression is not compressed. This parameter takes effect only during data import using COPY FROM.

  NOTE

  - This applies only to Astore tables.
  - TOAST data cannot be compressed.
  - This feature is not supported during the upgrade observation period.
  - When this feature is used, the new data page is used for storing the imported data.
  - When this feature is used together with the error tolerance mechanism (LOG ERRORS or LOG ERRORS DATA), compression may have no benefits if error data frequently occurs.
  - Compression is not supported when a single row is imported. For example, a partitioned table is imported and the function of non-rollback upon constraint violation is enabled, or the table contains triggers.
  - This feature cannot be used together with logical decoding. If logical decoding is enabled, an error will occur during compression of data imported using COPY.

- TRANSFORM ({ column_name [data_type] [AS transform_expr] } [, ...])

  Specifies the conversion expression of each column in the table. **data_type** specifies the data type of the column in the expression parameter. **transform_expr** is the target expression that returns the result value whose data type is the same as that of the target column in the table. For details about the expression, see **Expressions**.

  NOTE

  COPY FROM does not support conversion expressions specified for distribution keys.

- COPY_CUSTOM_ID 'custom_id_string'

  During data import, enter a user-defined character string, which is used as the identifier of the COPY import.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- In non-separation-of-duties scenarios, pg_catalog.pgxc_copy_error_log contains the **custom_id** column by default. If **copy_custom_id** is not used, **NULL** is inserted into the column.
- In separation-of-duties scenarios, if the schema with the same name as the current connection user contains the pgxc_copy_error_log table with the **custom_id** column, LOG ERRORS or LOG ERRORS DATA is used, and the error tolerance data volume is less than the value specified by **reject limit**, the character string input by this parameter is recorded to the **custom_id** column in the pgxc_copy_error_log table. One import may correspond to zero or multiple data records.
- This operation takes effect only for import. The error table and log table are not recorded during export.
- The length of a user-defined character string cannot exceed 64 bytes.
- If the error table and log table do not contain the **custom_id** column but the **copy_custom_id** parameter is set, an error message is displayed, prompting you to add a column to the table using a function or ALTER statement.
- The **custom_id** column in the error table and log table may not be unique. The column content can be duplicate.
- This parameter can be used only in ORA-compatible mode.

- **option**

  Specifies various parameters for a compatible foreign table.

  – FORMAT 'format_name'

    Specifies the format of the data source file.

    Value range: **CSV**, **TEXT**, and **BINARY**.

    - In CSV files, newline characters can be processed efficiently, but certain special characters may pose challenges. It is equivalent to the **CSV** parameter in **copy_option**.
    - In TEXT files, certain special characters can be processed efficiently, except newline characters.
    - In BINARY files, all data is stored/read as binary format rather than as text. It is somewhat faster than the text and CSV formats, but a binary-format file is less portable. It is equivalent to the **BINARY** parameter in **copy_option**.

    Default value: **TEXT**.

  – OIDS [boolean]

    Specifies whether to import and export OID hidden columns for tables (usually system catalogs) that contain OIDs.

    Value range: **true/on** and **false/off**.

    Default value: **false**.

  – DELIMITER 'delimiter_character'

    Specifies the column delimiter of data in a data file line. The value range is the same as that of **[USING] DELIMITERS 'delimiters'**.

  – NULL 'null_string'

    Specifies the string that represents a null value. The value is the same as that of **NULL null_string**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- HEADER [boolean]

  Specifies whether the exported data file contains a header row and whether the first row in the data stream is the header row during import. A header row describes information about each column.

  Value range: **true/on** and **false/off**.

  - If the parameter is set to **true** or **on**, the first row of the data text is identified as the header row during import and will be ignored. The first row in the data file exported is the header row.
  - If the parameter is set to **false** or **off**, the first row of the data text is identified as data and imported. The exported data file does not contain a header row.

  Default value: **false**.

  NOTE

  This parameter can be specified only in CSV or FIXED mode. The content in the header row of the FIXED file must meet the column length definition.

- FILEHEADER 'header_file_string'

  Specifies a file that defines the content in the header for exported data. The file contains data description of each column. It is equivalent to the **FILEHEADER 'header_file_string'** parameter in **copy_option**.

- FREEZE [boolean]

  Specifies a file that defines the content in the header for exported data. The file contains data description of each column.

  Value range: **true/on** and **false/off**. If the value is **true** or **on**, it is equivalent to the **FREEZE** parameter in **copy_option**.

  Default value: **false**.

- NOESCAPING [boolean]

  Specifies, in the TEXT format, whether to escape the backslash (\) and its following characters.

  Value range: **true/on** and **false/off**. If the value is **true** or **on**, it is equivalent to the **WITHOUT ESCAPING** parameter in **copy_option**.

  Default value: **false**.

- SKIP int_number

  Specifies that the first *int_number* rows of the data file are skipped during data import.

  Value range: an integer in the range [0,2147483647].

- USEEOF [boolean]

  If this parameter is set to **true** or **on**, no error is reported for a backslash followed by a period (\.). When importing a single column, if you enter "\." in **stdin** mode, it is considered as an end of line (EOL) character instead of data.

  Value range: **true/on** and **false/off**. If the value is **true** or **on**, this parameter is equivalent to the **USEEOF** parameter.

  Default value: **false**.

- QUOTE 'quote_character'

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies a quote character for a CSV file. It is equivalent to the **QUOTE [AS] 'quote_character'** parameter in **copy_option**.

- ESCAPE 'escape_character'

  Specifies an escape character for a CSV file. It is equivalent to the **ESCAPE [AS] 'escape_character'** parameter in **copy_option**.

- EOL 'newline_character'

  Specifies the newline character style of the imported or exported data file. The value is the same as that of **EOL 'newline_character'**.

- FORCE_QUOTE { (column_name [, ...]) | * }

  In CSV COPY TO mode, forces all non-**NULL** values to be surrounded by the QUOTE characters (see **QUOTE [AS] 'quote_character'**) around each specified column.

  Value range: existing columns.

  NOTE

  - The asterisk (*) indicates all columns.
  - The **NULL** output is not enclosed in quotation marks. If the original data has the content as the **NULL** option, the original data is quoted.

- FORCE_NOT_NULL (column_name [, ...])

  In CSV COPY FROM mode, if the specified column is **NULL** during data writing, the content specified by **NULL** (see **NULL [AS] 'null_string'**) is written instead. That is, this column does not allow **NULL** to be written directly.

  Value range: existing columns.

- ENCODING 'encoding_name'

  Specifies the encoding format of a data file. The default value is the current client-side encoding format. When COPY FROM is executed, the encoding format specified by **ENCODING** in the control file must be the same as that of the file. Otherwise, an error is reported or garbled characters are displayed in the imported data.

- IGNORE_EXTRA_DATA [boolean]

  Specifies whether to ignore excessive columns when the number of data source files exceeds the number of foreign table columns. This parameter is used only during data import.

  Value range: **true**/**on** or **false**/**off**

  - **true**/**on**: If the number of columns in a data source file is greater than that defined by the foreign table, the extra columns at the end of a row are ignored.
  - **false**/**off**: If the number of columns in a data source file is greater than that defined by the foreign table, the following error message is reported:

    ```
    extra data after last expected column
    ```

  Default value: **false**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

If a newline character at the end of a row is missing and the row and another row are integrated into one, data in another row is ignored after the parameter is set to **true**.

- POSITION_KEEP_TRAILING_SPACE

  Specifies whether to remove spaces at the end of a column value during fixed-length import. It is equivalent to the **POSITION_KEEP_TRAILING_SPACE** parameter in **copy_option**.

- COMPATIBLE_ILLEGAL_CHARS [boolean]

  Specifies whether to tolerate invalid characters during data import and export.

  Value range: **true/on** and **false/off**.

  - **true**/**on**: No error message is reported and data import is not interrupted when there are invalid characters. Invalid characters are converted into valid ones, and then imported to the database. Invalid characters are converted into valid ones, and then exported from the database. No error is reported and data export is not interrupted.
  - **false**/**off**: An error occurs when there are invalid characters, and the import stops. An error is reported and data export is interrupted when there are invalid characters and characters whose encodings do not exist.

  Default value: **false**/**off**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

The rules for converting invalid characters are as follows:

1. '\0' is converted to a space. When **support_zero_character** is enabled, data is imported in the original format.
2. Other invalid characters are converted to the characters specified by the GUC parameter **convert_illegal_char_mode**. The default value is **'?'**.
3. When **compatible_illegal_chars** is set to **true** or **on**, after invalid characters such as **NULL**, **DELIMITER**, **QUOTE**, and **ESCAPE** are converted to spaces or question marks, an error message stating "illegal chars conversion may confuse COPY escape 0x20" will be displayed to remind you of possible parameter confusion caused by the conversion.
4. If **compatible_illegal_chars** is set to **true** or **on**, data in binary format cannot be imported and an error message stating "cannot specify bulkload compatibility options in BINARY mode" will be displayed.
5. If the GUC parameter **copy_special_character_version** is set to **'no_error'**, invalid characters will not be checked during the import and will be displayed as garbled characters in query results. Enable this parameter with caution only when you fully understand its implications.
6. If **copy_special_character_version** is set to **'no_error'** and **compatible_illegal_chars** is set to **true**, the former has a higher priority. That is, invalid characters will not be checked during the import.
7. If **copy_special_character_version** is set to **'no_error'** or **compatible_illegal_chars** is set to **true**, a record is written to the database run logs each time a row of data containing invalid characters is identified. The log writing behavior occupies a small amount of disk bandwidth. To eliminate the impact, set the GUC parameter **enable_log_copy_illegal_chars** to **off**, that is, do not write records to the database run logs. You are advised not to modify this configuration unless otherwise required.

The rules for exporting invalid characters are as follows:

The export is classified into the scenario where transcoding is required (the database server code is different from the client code) and the scenario where transcoding is not required (the database server code is the same as the client code). The export performance varies according to the scenario. You can run the **show server_encoding;** statement to query the database server code and run the **show client_encoding;** statement to query the database client code.

- In scenarios where transcoding is required:

1. '\0' is converted to a space. When **support_zero_character** is enabled, data is exported in the original format.
2. Other invalid characters are converted to the characters specified by the GUC parameter **convert_illegal_char_mode**. The default value is **'?'**.
3. When **compatible_illegal_chars** is set to **true** or **on**, after invalid characters such as **NULL**, **DELIMITER**, **QUOTE**, and **ESCAPE** are exported to spaces or question marks, an error message stating "illegal chars conversion may confuse COPY escape 0x20" will be displayed to remind you of possible parameter confusion caused by the export.
4. If **compatible_illegal_chars** is set to **true** or **on**, data in binary format cannot be exported.
5. If **compatible_illegal_chars** is set to **true**, a record is written to the database run logs each time a row of data containing invalid characters is identified. The log writing behavior occupies a small amount of disk bandwidth. To eliminate the impact, set the GUC parameter **enable_log_copy_illegal_chars** to **off**, that is, do not write records to the database run logs. You are advised not to modify this configuration unless otherwise required.

- In scenarios where transcoding is not required:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1. Data is exported as it is in the database, even if the data contains invalid characters and **compatible_illegal_chars** is set to **true**.
2. If **compatible_illegal_chars** is set to **true** or **on**, data in binary format cannot be exported.

- FILL_MISSING_FIELDS [{ 'one' | boolean }]

  Specifies how to handle the problem that the last column of a row in a source data file is lost during data import.

  Value range: **one**, **true**/**on**, or **false**/**off**.

  Default value: **false**/**off**.

- COPY_CUSTOM_ID 'custom_id_string'

  During data import, enter a user-defined character string, which is used as the identifier of the COPY import. It is equivalent to the **COPY_CUSTOM_ID 'custom_id_string'** parameter in **copy_option**.

- DATE_FORMAT 'date_format_string'

  Specifies the DATE format for data import. This parameter is not supported for the binary format, as it would trigger the error "cannot specify bulkload compatibility options in BINARY mode." This parameter takes effect only during data import using COPY FROM.

  Value range: valid DATE formats. For details, see **Date and Time Processing Functions and Operators**.

- TIME_FORMAT 'time_format_string'

  Specifies the TIME format for data import. This parameter is not supported for the binary format, as it would trigger the error "cannot specify bulkload compatibility options in BINARY mode." This parameter takes effect only during data import using COPY FROM.

  Value range: a valid TIME value. Time zones cannot be used. For details, see **Date and Time Processing Functions and Operators**.

- TIMESTAMP_FORMAT 'timestamp_format_string'

  Specifies the TIMESTAMP format for data import. This parameter is not supported for the binary format, as it would trigger the error "cannot specify bulkload compatibility options in BINARY mode." This parameter takes effect only during data import using COPY FROM.

  Value range: a valid TIMESTAMP value. Time zones cannot be used. For details, see **Date and Time Processing Functions and Operators**.

- SMALLDATETIME_FORMAT 'smalldatetime_format_string'

  Specifies the SMALLDATETIME format for data import. This parameter is not supported for the binary format, as it would trigger the error "cannot specify bulkload compatibility options in BINARY mode." This parameter takes effect only during data import using COPY FROM.

  Value range: a valid SMALLDATETIME value. For details, see **Date and Time Processing Functions and Operators**.

- DATEA_FORMAT 'datea_format_string'

  Specifies the DATEA format for data import (supported only in ORA-compatible mode). The BINARY format is not supported. When data of such format is imported, error "cannot specify bulkload compatibility options in BINARY mode" will occur. This parameter takes effect only during data import using COPY FROM.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: a valid DATEAFORMAT value. For details, see **Date and Time Processing Functions and Operators**.

NOTE

The following special backslash sequences are recognized by COPY FROM:

- **\b**: Backslash (ASCII 8).
- **\f**: Form feed (ASCII 12).
- **\n**: Newline character (ASCII 10).
- **\r**: Carriage return character (ASCII 13).
- **\t**: Tab (ASCII 9).
- **\v**: Vertical tab (ASCII 11).
- **\digits**: Backslash followed by one to three octal digits specifies the character with that ASCII code
- **\xdigits**: Backslash *x* followed by one or two hex digits specifies the character with that numeric code

Permission Control Examples

```
gaussdb=> copy t1 from '/home/xy/t1.csv';
ERROR:  COPY to or from a file is prohibited for security concerns
HINT:  Anyone can COPY to stdout or from stdin. gsql's \copy command also works for anyone.
gaussdb=> grant gs_role_copy_files to xxx;
```

This error occurs because a non-initial user does not have the COPY permission. To use COPY, an administrator needs to enable the GUC parameter **enable_copy_server_files**, whereas a common user also needs to join the gs_role_copy_files group.

Examples

```
-- Create a schema.
gaussdb=# CREATE SCHEMA tpcds;

-- Create the tpcds.ship_mode table.
gaussdb=# CREATE TABLE tpcds.ship_mode
(
    SM_SHIP_MODE_SK          INTEGER              NOT NULL,
    SM_SHIP_MODE_ID          CHAR(16)             NOT NULL,
    SM_TYPE                  CHAR(30)                 ,
    SM_CODE                  CHAR(10)                   ,
    SM_CARRIER               CHAR(20)                   ,
    SM_CONTRACT              CHAR(20)
)
;

-- Insert a single data record into the tpcds.ship_mode table.
gaussdb=# INSERT INTO tpcds.ship_mode VALUES (1,'a','b','c','d','e');

-- Copy data from the tpcds.ship_mode table to the /home/omm/ds_ship_mode.dat file.
gaussdb=# COPY tpcds.ship_mode TO '/home/omm/ds_ship_mode.dat';

-- Output tpcds.ship_mode to STDOUT.
gaussdb=# COPY tpcds.ship_mode TO STDOUT;

-- Output the data of tpcds.ship_mode to STDOUT, with these settings: delimiter',' and encoding'utf8'.
gaussdb=# COPY tpcds.ship_mode TO STDOUT WITH (delimiter ',', encoding 'utf8');

-- Output the data of tpcds.ship_mode to STDOUT, with these settings: format 'CSV' and
force_quote(SM_SHIP_MODE_SK).
gaussdb=# COPY tpcds.ship_mode TO STDOUT WITH (format 'CSV', force_quote(SM_SHIP_MODE_SK));
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create the tpcds.ship_mode_t1 table.
gaussdb=# CREATE TABLE tpcds.ship_mode_t1
(
    SM_SHIP_MODE_SK           INTEGER               NOT NULL,
    SM_SHIP_MODE_ID           CHAR(16)              NOT NULL,
    SM_TYPE                   CHAR(30)                      ,
    SM_CODE                   CHAR(10)                      ,
    SM_CARRIER                CHAR(20)                      ,
    SM_CONTRACT               CHAR(20)
)
;

-- Copy data from STDIN to the tpcds.ship_mode_t1 table.
gaussdb=# COPY tpcds.ship_mode_t1 FROM STDIN;

-- Enter a row of data as an example.
--gaussdb=# COPY tpcds.ship_mode_t1 FROM STDIN;
--Enter data to be copied followed by a newline.
--End with a backslash and a period on a line by itself.
-->> 1    a a a a a
-->> \.
-- Note: Enter a tab between data.

-- Copy data from the /home/omm/ds_ship_mode.dat file to the tpcds.ship_mode_t1 table.
gaussdb=# COPY tpcds.ship_mode_t1 FROM '/home/omm/ds_ship_mode.dat';

-- Copy data from the /home/omm/ds_ship_mode.dat file to the tpcds.ship_mode_t1 table, convert the data using the TRANSFORM expression, and insert the 10 characters on the left of the SM_TYPE column into the table.
gaussdb=# COPY tpcds.ship_mode_t1 FROM '/home/omm/ds_ship_mode.dat' TRANSFORM (SM_TYPE AS LEFT(SM_TYPE, 10));

-- Copy data from the /home/omm/ds_ship_mode.dat file to the tpcds.ship_mode_t1 table, with these settings: format 'text', delimiter E'\t'), ignore_extra_data 'true', and noescaping 'true'.
gaussdb=# COPY tpcds.ship_mode_t1 FROM '/home/omm/ds_ship_mode.dat' WITH(format 'text', delimiter E'\t', ignore_extra_data 'true', noescaping 'true');

-- Copy data from the /home/omm/ds_ship_mode_fixed.dat file to the tpcds.ship_mode_t1 table, with the import format set to FIXED, fixed-length format specified (FORMATTER(SM_SHIP_MODE_SK(0, 2), SM_SHIP_MODE_ID(2,16), SM_TYPE(18,30), SM_CODE(50,10), SM_CARRIER(61,20), SM_CONTRACT(82,20))), excessive columns ignored (ignore_extra_data), and headers included (header).
gaussdb=# COPY tpcds.ship_mode TO '/home/omm/ds_ship_mode_fixed.dat' FIXED FORMATTER(SM_SHIP_MODE_SK(0, 2), SM_SHIP_MODE_ID(2,16), SM_TYPE(18,30), SM_CODE(50,10), SM_CARRIER(61,20), SM_CONTRACT(82,20)) header;
gaussdb=# COPY tpcds.ship_mode_t1 FROM '/home/omm/ds_ship_mode_fixed.dat' FIXED FORMATTER(SM_SHIP_MODE_SK(0, 2), SM_SHIP_MODE_ID(2,16), SM_TYPE(18,30), SM_CODE(50,10), SM_CARRIER(61,20), SM_CONTRACT(82,20)) header ignore_extra_data;

-- Drop tables and the schema.
gaussdb=# DROP TABLE tpcds.ship_mode;
gaussdb=# DROP TABLE tpcds.ship_mode_t1;
gaussdb=# DROP SCHEMA tpcds;
```

1.13.9.10 CREATE AGGREGATE

Description

Creates an aggregate function.

Syntax

```
CREATE AGGREGATE name ( input_data_type [ , ... ] ) (
    SFUNC = sfunc,
    -- SFUNC1 = sfunc,
    STYPE = state_data_type
    -- STYPE1 = state_data_type,
    [ , FINALFUNC = ffunc ]
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    [ , INITCOND = initial_condition ]
    -- [ , INITCOND1 = initial_condition ]
    [ , SORTOP = sort_operator ]
    [ , CFUNC = collection_func ]
    [ , INITCOLLECT = initial_collection_condition ]
    [ , IFUNC = init_func ]
    [ , SHIPPABLE = is_shippable ]
);
```

Original syntax/earlier syntax format:

```
CREATE AGGREGATE name (
    BASETYPE = base_type,
    SFUNC = sfunc,
    -- SFUNC1 = sfunc,
    STYPE = state_data_type
    -- STYPE1 = state_data_type,
    [ , FINALFUNC = ffunc ]
    [ , INITCOND = initial_condition ]
    -- [ , INITCOND1 = initial_condition ]
    [ , SORTOP = sort_operator ]
    [ , CFUNC = collection_func ]
    [ , INITCOLLECT = initial_collection_condition ]
    [ , IFUNC = init_func ]
);
```

Parameters

- **name**

  Name of the aggregate function to be created, which can be specified using a schema.

- **input_data_type**

  Data type of the input to be processed by the aggregate function.

  To create an aggregate function without input parameters, you can use an asterisk (*) instead of a list of input data types. For example, count (*) is a zero-parameter aggregate function.

- **base_type**

  You can define the data type of the input to be processed by the aggregate function by specifying the **BASETYPE** keyword, instead of directly writing the input type after the aggregate function name (that is, by specifying **input_data_type**).

  When you define the input data type of an aggregate function in this way, the aggregate function must have one input parameter. If you create a zero-parameter aggregate function, set this keyword to **ANY** (instead of ***).

- **sfunc**

  Name of the state conversion function that will be called on each input line. For an aggregate function with *N* parameters, **sfunc** must have *N* + 1 parameters. The first parameter is of the type specified by **state_data_type**, and the other parameters match the declared input data types. The function must return a value of the **state_data_type** type. This function accepts the current state value and the current input data, and returns the next state value.

  The default behavior of the conversion function of an Oracle Database is **STRICT**, that is, the **NULL** input value is skipped. For GaussDB, you need to define the **STRICT** attribute of the conversion function.

- **state_data_type**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Data type of the aggregation status value.

- **ffunc**

  Final processing function called after all the input lines have been converted, which calculates the result of aggregation. This function must accept a parameter of **state_data_type**. The output data type of the aggregation is defined as the return type of this function. If **ffunc** is not specified, the state value of the aggregation result is used as the aggregation result, and the output type is **state_data_type**.

- **initial_condition**

  Initial setting (value) of a state value. It must be a text constant value acceptable to **state_data_type**. If not specified, the initial state value is **NULL**.

- **sort_operator**

  Sort operator (optionally schema-qualified) used for MIN or MAX aggregation.

- **collection_func**

  collection_func is essentially the status conversion result generated by combining DNs. If there is no final function, the result generated by **collection_func** is the aggregation result.

  If you want **collection_func** to skip null values, specify **collection_func** as **STRICT**.

  collection_func must have two input parameters of the **state_data_type** type. The first input parameter **state_data_type** will be passed to the first parameter of the next **collection_func** execution. If **is_shippable** is **false** (that is, the aggregate function does not support pushdown), **collection_func** will not be executed.

- **initial_collection_condition**

  Initial setting (value) of the state value of **collection_func**. It must be a text constant value acceptable to **state_data_type**. If not specified, the initial state value is **NULL**.

- **init_func**

  Initial setting function of the **sfunc** status value. It cannot return **NULL**. It must return the state_data_type type and have no input parameter. If both **init_func** and **initial_condition** are set, the return value of **init_func** is preferentially used as the initial value.

- **is_shippable**

  Specifies whether the aggregate function can be pushed down. The value can only be **true** or **false**. Currently, this parameter does not affect the promotion of built-in aggregate functions. It affects only the behavior of user-defined aggregate functions. The default value is **false**. User-defined aggregate functions can be pushed down only when **collection_func** is executed and **is_shippable** is set to **true**.

  NOTE

  If an aggregate function can be pushed down, it cannot reference package content, including variables and functions. Otherwise, an error may be reported during function execution. For example, sfunc, cfunc, finalfunc, and ifunc cannot be package functions, and sfunc, finalfunc, and ifunc cannot reference package variables or call package functions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

```
-- Create a user-defined function.
gaussdb=# CREATE OR REPLACE FUNCTION int_add(int,int)
 returns int as $BODY$
declare
begin
 return $1 + $2;
end;
$BODY$ language plpgsql;

-- Create an aggregate function.
gaussdb=# CREATE AGGREGATE sum_add(int)
(
    sfunc = int_add,

    stype = int,

    initcond = '0'

);

-- Create a test table and add data.
gaussdb=# CREATE TABLE test_sum(a int,b int,c int);
gaussdb=# INSERT INTO test_sum VALUES(1,2),(2,3),(3,4),(4,5);

-- Execute the aggregate function.
gaussdb=# SELECT sum_add(a) FROM test_sum;
  sum_add
-----------
   10

-- Drop the aggregate function.
gaussdb=# DROP AGGREGATE sum_add(int);

-- Drop the user-defined function.
gaussdb=# DROP FUNCTION int_add(int,int);

-- Drop the test table.
gaussdb=# DROP TABLE test_sum;
```

Helpful Links

ALTER AGGREGATE and **DROP AGGREGATE**

1.13.9.11 CREATE AUDIT POLICY

Description

Creates an audit policy.

Precautions

- Only users with the POLADMIN or SYSADMIN permission, or the initial user can create and maintain audit policies.
- Before creating an audit policy, ensure that the security policy parameter has been enabled. That is, the audit policy takes effect only after the GUC parameter **enable_security_policy** is set to **on**.
- A system administrator or security policy administrator can access the GS_AUDITING_POLICY, GS_AUDITING_POLICY_ACCESS, GS_AUDITING_POLICY_PRIVILEGES, and GS_AUDITING_POLICY_FILTERS system catalogs to query the created audit policies.

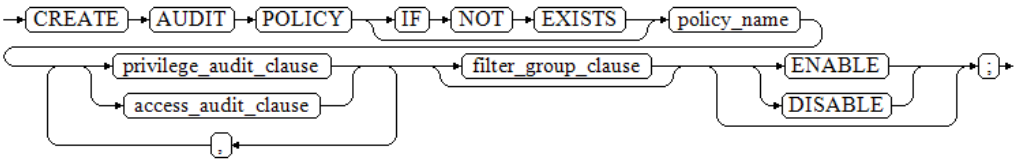
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

When you use database links to perform operations on remote objects, the client initiates a database link request. The actual sender is the server, and the attributes such as the IP address of the sender are the values of the server. For details, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.

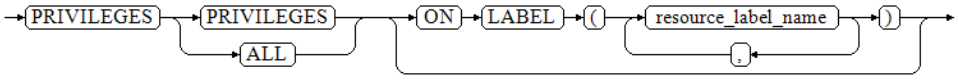
Syntax

```
CREATE AUDIT POLICY [ IF NOT EXISTS ] policy_name { { privilege_audit_clause | access_audit_clause }
[, ... ] [ filter_group_clause ] [ ENABLE | DISABLE ] };
```



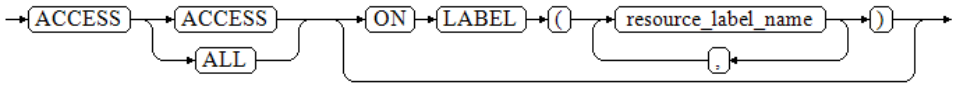
- privilege_audit_clause

  ```
  PRIVILEGES { PRIVILEGES | ALL } [ ON LABEL ( resource_label_name [, ... ] ) ]
  ```

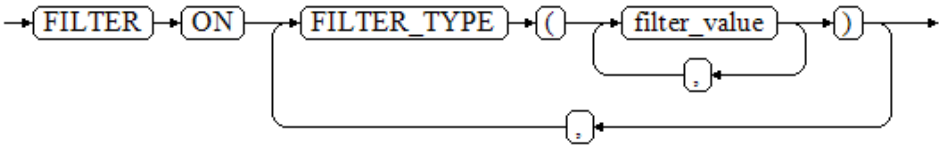
- access_audit_clause

  ```
  ACCESS { ACCESS | ALL } [ ON LABEL ( resource_label_name [, ... ] ) ]
  ```

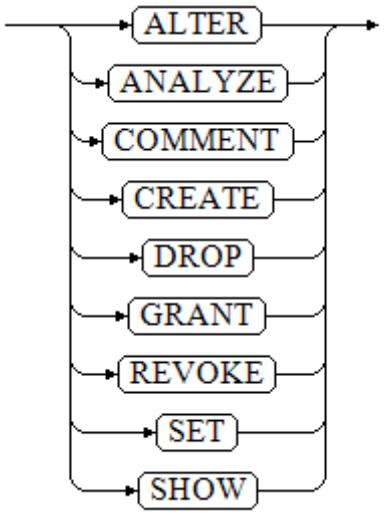
- filter_group_clause

  ```
  FILTER ON { FILTER_TYPE ( filter_value [, ... ] ) } [, ... ]
  ```

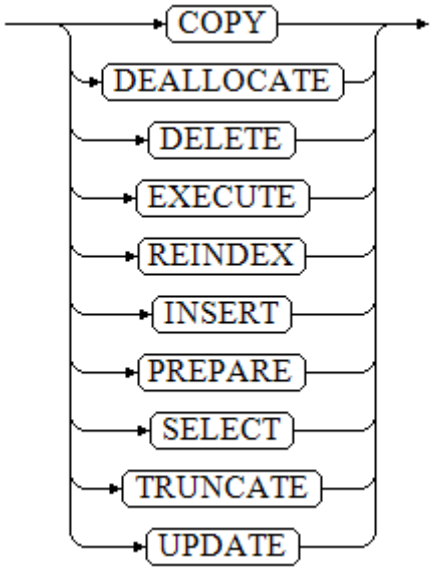
- PRIVILEGES:

  ```
  { ALTER | ANALYZE | COMMENT | CREATE | DROP | GRANT | REVOKE | SET | SHOW }
  ```

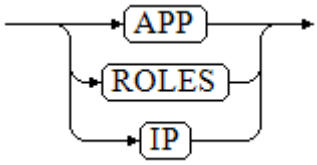
  

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- ACCESS:
  { COPY | DEALLOCATE | DELETE | EXECUTE | REINDEX | INSERT | PREPARE | SELECT | TRUNCATE | UPDATE }



- FILTER_TYPE
  { APP | ROLES | IP }



Parameters

- **IF NOT EXISTS**

  You can use this keyword to check whether a specified audit policy exists to avoid repeated creation.

- **policy_name**

  The audit policy name must be unique to avoid conflicts with existing policies.

  Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

  NOTE

  - If the audit policy name contains more than 63 characters, the database truncates it and retains the first 63 characters as the audit policy name.
  - If the audit policy name contains uppercase letters, the database automatically converts the uppercase letters into lowercase letters. To create an audit policy name that contains uppercase letters, enclose the audit policy name with double quotation marks ("").
  - The audit policy name must start with a letter or underscore (_) and can contain only lowercase letters (a to z), uppercase letters (A to Z), underscores (_), digits (0 to 9), and dollar signs ($).

- **resource_label_name**

  Specifies the resource label name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **PRIVILEGES**

  Specifies the operations that are audited in the database: CREATE, ALTER, DROP, ANALYZE, COMMENT, GRANT, REVOKE, SET, and SHOW.

  If the value is **ANALYZE**, the ANALYZE and VACUUM operations are audited.

- **ACCESS**

  Specifies the operations that are audited in the database: SELECT, COPY, DEALLOCATE, DELETE, EXECUTE, INSERT, PREPARE, REINDEX, TRUNCATE, and UPDATE.

- **ALL**

  Specifies the PRIVILEGES or ACCESS operations supported for the database. When the form is { PRIVILEGES | ALL }, **ALL** indicates all PRIVILEGES operations. When the form is { ACCESS | ALL }, **ALL** indicates all ACCESS operations.

- **FILTER_TYPE**

  Specifies the types of information to be filtered by the audit policy, including **APP**, **ROLES**, and **IP**.

- **filter_value**

  Specifies the detailed information to be filtered by the audit policy.

- **ENABLE | DISABLE**

  Enables or disables the unified audit policy. If not specified, **ENABLE** is used by default.

Examples

- Creates audit policy for executing **CREATE** on the database.

```
-- Create the adt1 policy.
gaussdb=# CREATE AUDIT POLICY adt1 PRIVILEGES CREATE;

-- View the adt1 policy.
gaussdb=# SELECT * FROM GS_AUDITING_POLICY;
 polname | polcomments |         modifydate         | polenabled
---------+-------------+----------------------------+------------
 adt1    |             | 2023-11-06 16:41:40.947417 | t

-- Check the location where audit logs are stored.
gaussdb=# SHOW audit_directory;

-- Drop the audit policy adt1.
gaussdb=# DROP AUDIT POLICY adt1;
```

- Create an audit policy to audit only the CREATE operation performed by the **dev_audit** user.

```
-- Create user dev_audit.
gaussdb=# CREATE USER dev_audit PASSWORD '********';

-- Create the tb_for_audit table.
gaussdb=# CREATE TABLE tb_for_audit(col1 text, col2 text, col3 text);

-- Create the adt_lb0 resource label based on the tb_for_audit table.
gaussdb=# CREATE RESOURCE LABEL adt_lb0 add TABLE(public.tb_for_audit);

-- Create the adt2 audit policy for the CREATE operation on the adt_lb0 resource.
gaussdb=# CREATE AUDIT POLICY adt2 PRIVILEGES CREATE ON LABEL(adt_lb0) FILTER ON
ROLES(dev_audit);

-- Drop the audit policy adt2.
gaussdb=# DROP AUDIT POLICY adt2;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop the tb_for_audit table.
gaussdb=# DROP TABLE tb_for_audit;

-- Drop the dev_audit user.
gaussdb=# DROP USER dev_audit;
```

- Create an audit policy to audit only the SELECT, INSERT, and DELETE operations performed on the **adt_lb0** resource by user **dev_audit** using client tool gsql on the servers whose IP addresses are **10.20.30.40** and **127.0.0.0/24**, respectively.

```
-- Create user dev_audit.
gaussdb=# CREATE USER dev_audit PASSWORD '********';

-- Create the audit policy adt3.
gaussdb=# CREATE AUDIT POLICY adt3 ACCESS SELECT ON LABEL(adt_lb0), INSERT ON
LABEL(adt_lb0), DELETE FILTER ON ROLES(dev_audit), APP(gsql), IP('10.20.30.40', '127.0.0.0/24');

-- Drop the audit policy adt3.
gaussdb=# DROP AUDIT POLICY adt3;

-- Drop the dev_audit user.
gaussdb=# DROP USER dev_audit;
```

Helpful Links

ALTER AUDIT POLICY and **DROP AUDIT POLICY**

1.13.9.12 CREATE BARRIER

Description

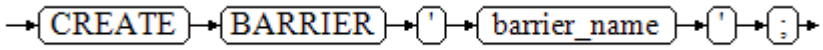
Creates a barrier for new cluster nodes, which can be used for data restoration.

Precautions

- This functionality is designed for internal use only. Avoid direct use by users.
- Generally, this functionality is used only for backup and restoration. Therefore, it can be executed only in the following scenarios:
  - The database initial user can run this command.
  - If the backup and restoration mode is enabled on the CN, that is, the GUC parameter **operation_mode** is set to **on**, users with the OPRADMIN permission can run this command.

Syntax

```
CREATE BARRIER 'barrier_name';
```



Parameters

barrier_name

Specifies the name of a barrier.

Value range: character strings complying with **Identifier Naming Conventions**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

```
-- Specify the barrier name.
gaussdb=# CREATE BARRIER 'barrier1';
```

1.13.9.13 CREATE CLIENT MASTER KEY

Description

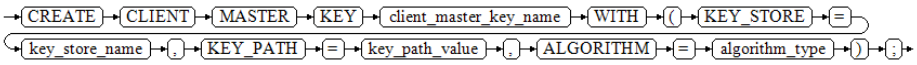
Creates a client master key (CMK).

Precautions

- This syntax is specific to the fully-encrypted database.
- Before using this syntax, set environment variables for the external key manager on the database driver side. For details, see "Feature Usage Guidelines > Setting Encrypted Equality Query" in *Developer Guide*.
- When connecting to the database, you need to enable the connection parameters for the encrypted equality query feature on the database driver side before running **CREATE CLIENT MASTER KEY**. The encrypted equality query feature adopts a multi-level encryption model. The master key encrypts the column key, and the column key encrypts data.
- The master key is provided by an external key manager. This syntax processes only information such as the key source and key ID. The following external key managers are supported:
  - Huawei Cloud KMS huawei_kms
  - Derived key from the user password or provided key user_token
  - Intrinsic key management based on the Kunpeng TEE and the SanSec SJJ1994 HSM sdf_kms

Syntax

```
CREATE CLIENT MASTER KEY client_master_key_name WITH (KEY_STORE = key_store_name, KEY_PATH =
key_path_value, ALGORITHM = algorithm_type);
```



Parameters

- **client_master_key_name**

  Specifies the name of the CMK to be created. In the same namespace, the CMK name must be unique.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **KEY_STORE**

  External key manager.

  Value range: See **Table 1-393**.

  - If **KEY_STORE** is set to **user_token**, you do not need to provide the **KEY_PATH** parameter.
  - If **KEY_STORE** is set to **sdf_kms**, users do not need to provide the **ALGORITHM** parameters. By default, **SGD_SM4_ECB** is supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **KEY_PATH**

  Each key is managed by an external key manager and the key path format varies depending on the key manager.

  Value range: character strings. See **Table 1-393**. The value of **KEY_PATH** must be enclosed in single or double quotation marks. If the length of a character string exceeds 64 characters, only single quotation marks can be used.

- **ALGORITHM**

  Algorithm used by the key.

  Value range: See **Table 1-393**.

Table 1-393 Parameter values for different key managers

KEY_STORE	KEY_PATH	ALGORITHM
huawei_kms	Format: '{KmsApiUrl}/{*Key ID*}' Reference: 'https://kms.{*Project*}.myhuaweicloud.com/v1.0/{*Project ID*}/kms/{*Key ID*}' Example: 'https://kms.cn-north-4.myhuaweicloud.com/v1.0/00000000000000000000000000000000/kms/00000000-0000-0000-0000-000000000000'	AES_256 SM4
user_token	Users do not need to provide **KEY_PATH**.	Currently, the following algorithms are supported: • AES_256_CBC • AES_256_GCM • SM4 • SM4_HMAC_SM3 • SM4_CTR_HMAC_SM3
sdf_kms	Format: '{*ID of the key encryption key (KEK)*}' Note: This parameter can be generated by the key management tool provided by iTrustee or by the key management tool in the SanSec SJJ1994 HSM.	Users do not need to provide **ALGORITHM**.

Examples

- user_token scenario:

```
-- Decompress the installation package GaussDB-Kernel_Database version number_OS version number_64bit_Gsql.tar.gz to find the gsql_env.sh script.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Use the script to automatically configure the environment variables.
source gsql_env.sh

-- Connect to the database, use the privileged account, and create a user, for example, alice.
gsql -p Port number -d postgres -r
gaussdb=# CREATE USER alice PASSWORD '*******';
gaussdb=# \q

-- Connect to the database. The -C parameter must be used.
gsql -p Port number -d postgres -U alice -r -C

-- Set the user password or derived key. gsql can use password=stdin or key_token=stdin for interactive input.
-- Set the user password. The password must contain at least eight characters, including three types of the following: uppercase letters, lowercase letters, digits, and symbols.
gaussdb=> \key_info keyType=user_token,password=*********
-- Alternatively, connect to a key that meets the security strength requirements. The key is a hexadecimal code.
gaussdb=> \key_info keyType=user_token,key_token=********************************

-- Create a master key.
gaussdb=> CREATE CLIENT MASTER KEY alice_cmk WITH ( KEY_STORE = user_token , ALGORITHM = AES_256_GCM );

-- Delete the master key.
gaussdb=> DROP CLIENT MASTER KEY alice_cmk;
gaussdb=> \q

-- Connect to the database, use the privileged account, and delete user alice.
gsql -p Port number -d postgres -r
gaussdb=# DROP USER alice;
```

Helpful Links

DROP CLIENT MASTER KEY

1.13.9.14 CREATE COLUMN ENCRYPTION KEY

Description

Creates a column encryption key (CEK).

Precautions

- This syntax is specific to a fully-encrypted database.
- When using gsql to connect to a database server, you need to use the **-C** parameter to enable the fully-encrypted database so that CREATE COLUMN ENCRYPTION KEY can be used.
- When defining a column in a table, you can specify a CEK to encrypt the column.

Syntax

```
CREATE COLUMN ENCRYPTION KEY column_encryption_key_name WITH VALUES(CLIENT_MASTER_KEY = client_master_key_name, ALGORITHM = algorithm_type [, ENCRYPTED_VALUE = encrypted_value]);
```

Parameters

- **column_encryption_key_name**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the name of the CEK to be created. In the same namespace, the CEK name must be unique. The CEK created using this syntax can be used for column-level encryption.

Value range: character strings complying with **Identifier Naming Conventions**.

- **CLIENT_MASTER_KEY**

  Specifies a CMK to encrypt a CEK.

  Value range: a CMK name. The CMK is created using the **CREATE CLIENT MASTER KEY** syntax.

- **ALGORITHM**

  Algorithm used by the key.

  Value range: AEAD_AES_256_CBC_HMAC_SHA256, AEAD_AES_128_CBC_HMAC_SHA256, AEAD_AES_256_CTR_HMAC_SHA256, AES_256_GCM, SM4_SM3, SM4_HMAC_SM3, or SM4_CTR_HMAC_SM3.

  - The data expansion rates of different algorithms are sorted as follows: AEAD_AES_256_CTR_HMAC_SHA256 = SM4_CTR_HMAC_SM3 < AES_256_GCM < AEAD_AES_256_CBC_HMAC_SHA256 = AEAD_AES_128_CBC_HMAC_SHA256 = SM4_SM3 = SM4_HMAC_SM3. You are advised to use the AEAD_AES_256_CTR_HMAC_SHA256, SM4_CTR_HMAC_SM3, or AES_256_GCM algorithm.

- **ENCRYPTED_VALUE**

  (Optional) Specifies a user-defined key. The key contains 28 to 256 characters.

  - A key with 28 characters provides the same security strength as AES-128.
  - If you want to use AES-256, the key must contain 39 characters.
  - If this parameter is not specified, a 256-character key is automatically generated.
  - When the master key generated by Huawei KMS is used to encrypt the column key, the length of the key passed through the **ENCRYPTED_VALUE** field must be an integer multiple of 16 bytes.

NOTICE

SM algorithm restrictions: SM2, SM3, and SM4 are Chinese national cryptography standards. To avoid legal risks, these algorithms must be used together. If the SM4, SM4_HMAC_SM3, or SM4_CTR_HMAC_SM3 algorithm is specified for encrypting the column key during master key creation, the SM4_SM3, SM4_HMAC_SM3, or SM4_CTR_HMAC_SM3 algorithm must be specified for encrypting the data during column key creation. This restriction does not apply to sdf_kms key management.

Helpful Links

ALTER COLUMN ENCRYPTION KEY and **DROP COLUMN ENCRYPTION KEY**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.9.15 CREATE CONVERSION

Description

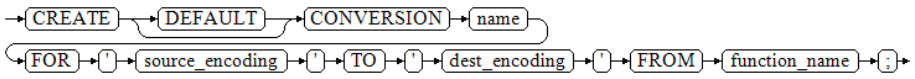
Creates a new encoding conversion rule to translate data from one character encoding (source encoding) to another character encoding (target encoding).

Precautions

- This functionality is for internal use only and cannot be directly used by users. Encoding conversion rules can only be created internally.

Syntax

```
CREATE [ DEFAULT ] CONVERSION name
   FOR 'source_encoding' TO 'dest_encoding' FROM function_name;
```



Parameters

- **DEFAULT**

  Sets the conversion executed between the source encoding and the target encoding by default between the client and the server.

  Within a schema, each encoding pair has only one default conversion.

- **name**

  Name of the conversion rule. The conversion rule name must be unique within a schema.

- **source_encoding**

  Source encoding format.

- **dest_encoding**

  Target encoding format.

- **function_name**

  Function that performs the actual encoding conversion.

  The function must be in the following format:

  ```
  conv_proc(
     integer, -- Source encoding ID
     integer, -- Target encoding ID
     cstring, -- Source character string (C character string ending with a null value)
     internal,-- Target (filled with a null-terminated C character string)
     integer  -- Length of the source string
  ) RETURNS void;
  ```

1.13.9.16 CREATE DATABASE

Description

Creates a database. By default, the new database is created by copying **template0** of the standard system database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

- Only users with the CREATEDB permission can create a database. System administrators possess this permission by default.
- This command cannot be executed in transaction blocks.
- **templatem** cannot be used as the database name when a database is created.
- If the **DBCOMPATIBILITY** and **TEMPLATE** keywords are not specified when you create a database, the default compatible database type is determined by the value of the GUC parameter **distribute_default_compatibility**.
- If a compatible database you create is not ORA-compatible, M-compatible, or MySQL-compatible, some information of the following system objects is different. For details, see **Table 1-394**.

Table 1-394 System object differences

System Object Type	System Object Name	Difference
System views	pg_catalog.my_tab_modifications	When a database is created, the **timestamp** column in the view is of the timestamp(0) without time zone type. After the view is upgraded, the column type changes to date.
	pg_catalog.db_tab_modifications	
	pg_catalog.v$open_cursor	When a database is created, the **last_sql_active_time** column in the view is of the timestamp(0) without time zone type. After the view is upgraded, the column type changes to date.
	pg_catalog.my_audit_trail	When a database is created, the **timestamp** and **logoff_time** columns in the view are of the timestamp(0) without time zone type. After the view is upgraded, the column type changes to date.
	sys.v_session	When a database is created, the **logon_time** and **prev_exec_start** columns in the view are of the timestamp(0) without time zone type. After the view is upgraded, the column type changes to date.
	sys.gv_session	
	sys.db_indexes	When a database is created, the **last_analyzed** column in the view is of the timestamp(0) without time zone type. After the view is upgraded, the column type changes to date.
	sys.adm_indexes	
	sys.adm_tab_cols	

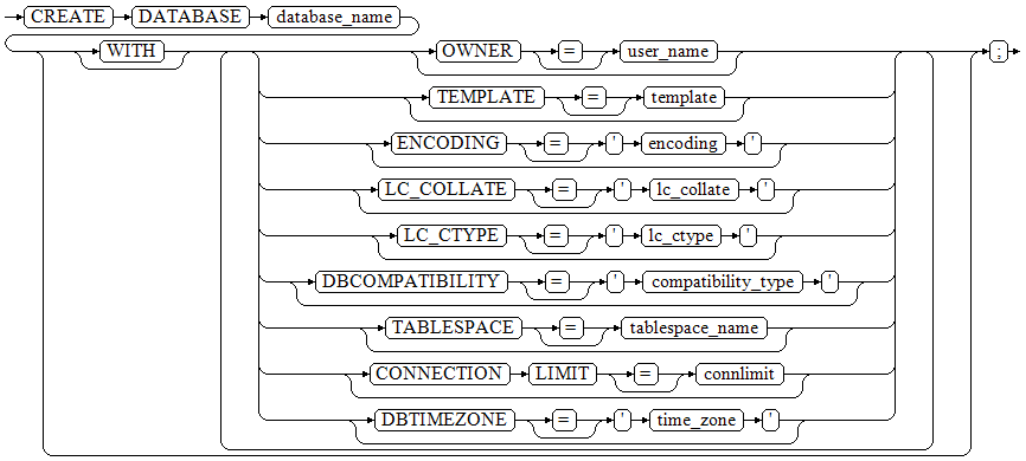
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

System Object Type	System Object Name	Difference
	sys.db_tab_columns	
	sys.adm_tab_columns	
	sys.db_tab_col_statistics	
	sys.adm_tab_col_statistics	
	sys.db_part_col_statistics	
	sys.adm_part_col_statistics	
	sys.db_subpart_col_statistics	
	sys.adm_subpart_col_statistics	
Advanced package function	dbe_scheduler.set_attribute	When a database is created, the dbe_scheduler.set_attribute(name text, attribute text, value date) overloaded function is missing. After the packages plug-in is upgraded, the dbe_scheduler.set_attribute(name text, attribute text, value date) overloaded function exists.

Syntax

```
CREATE DATABASE database_name
   [ [ WITH ] { [ OWNER [=] user_name ] |
            [ TEMPLATE [=] template ] |
            [ ENCODING [=] 'encoding' ] |
            [ LC_COLLATE [=] 'lc_collate' ] |
            [ LC_CTYPE [=] 'lc_ctype' ] |
            [ DBCOMPATIBILITY [=] 'compatibility_type' ] |
            [ TABLESPACE [=] tablespace_name ] |
            [ CONNECTION LIMIT [=] connlimit ] |
            [ DBTIMEZONE [=] 'time_zone' ] }[...] ];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- **database_name**

  Specifies the database name.

  Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

  NOTE

  - If the value contains more than 63 characters, the database truncates it and retains the first 63 characters as the database name.
  - If a database name contains uppercase letters, the database automatically converts the uppercase letters into lowercase letters. To create a database name that contains uppercase letters, enclose the database name with double quotation marks ("").

- **OWNER [=] user_name**

  (Optional) Specifies the owner of the new database.

  Value range: an existing username. If no username is specified, the owner of the new database is the current user.

- **TEMPLATE [=] template**

  (Optional) Creates a database based on the specified standard system database template.

  GaussDB creates a database by copying data from a template database. GaussDB contains default template databases **template0**, **template1**, **templatea**, and **templatem** and a default user database **postgres**.

  Value range: **template0**, **templatem**, and **templatea**.

- **ENCODING [=] 'encoding'**

  (Optional) Specifies the character encoding used by the database.

  Common values are **GBK**, **UTF8**, **Latin1**, and **GB18030**. The supported character encodings are listed in **Table 1-395**.

  - If no character encoding is specified, the M-compatible database uses the UTF-8 encoding by default, and other compatible databases use the encoding of the template database by default.
  - The default codes of the template databases **template0**, **template1**, and **templatea** are related to the OS environment.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

– The character encoding of the **template1** database cannot be changed. To modify the character encoding, use the **template0** or **templatea** database to create a database.

Table 1-395 Character encodings supported by GaussDB

Name	Description	Language	Server-side Encoding	ICU Support	Number of Bytes/ Characters	Alias
BIG5	Big Five	Traditional Chinese	No	No	1–2	WIN950 and Windows950
EUC_CN	Extended Unix Code-CN	Simplified Chinese	Yes	Yes	1–3	-
EUC_JP	Extended Unix Code-JP	Japanese	Yes	Yes	1–3	-
EUC_JIS_2004	Extended Unix Code-JP, JIS X 0213	Japanese	Yes	No	1–3	-
EUC_KR	Extended Unix Code-KR	Korean	Yes	Yes	1–3	-
EUC_TW	Extended Unix Code-Taiwan, China	Traditional Chinese	Yes	Yes	1–3	-
GB18030	National standards	Chinese	Yes	No	1–4	-
GB18030_2022	National standards	Chinese	Yes	No	1–4	-
GBK	Extended national standards	Simplified Chinese	Yes	No	1–2	WIN936 and Windows936

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Language	Server-side Encoding	ICU Support	Number of Bytes/ Characters	Alias
ISO_8859_5	ISO 8859-5, ECMA 113	Latin/ Cyrillic	Yes	Yes	1	-
ISO_8859_6	ISO 8859-6, ECMA 114	Latin/ Arabic	Yes	Yes	1	-
ISO_8859_7	ISO 8859-7, ECMA 118	Latin/ Greek	Yes	Yes	1	-
ISO_8859_8	ISO 8859-8, ECMA 121	Latin/ Hebrew	Yes	Yes	1	-
JOHAB	JOHAB	Korean	No	No	1–3	-
KOI8R	KOI8-R	Cyrillic (Russian)	Yes	Yes	1	KOI8
KOI8U	KOI8-U	Cyrillic (Ukrainian)	Yes	Yes	1	-
LATIN1	ISO 8859-1, ECMA 94	Western European languages	Yes	Yes	1	ISO88591
LATIN2	ISO 8859-2, ECMA 94	Central European languages	Yes	Yes	1	ISO88592
LATIN3	ISO 8859-3, ECMA 94	South European languages	Yes	Yes	1	ISO88593

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Language	Server-side Encoding	ICU Support	Number of Bytes/ Characters	Alias
LATIN4	ISO 8859-4, ECMA 94	North European languages	Yes	Yes	1	ISO88594
LATIN5	ISO 8859-9, ECMA 128	Turkish	Yes	Yes	1	ISO88599
LATIN6	ISO 8859-10, ECMA 144	Germanic languages	Yes	Yes	1	ISO885910
LATIN7	ISO 8859-13	Baltic languages	Yes	Yes	1	ISO885913
LATIN8	ISO 8859-14	Celtic languages	Yes	Yes	1	ISO885914
LATIN9	ISO 8859-15	LATIN1 with Euro and accents	Yes	Yes	1	ISO885915
LATIN10	ISO 8859-16, ASRO SR 14111	Romanian	Yes	No	1	ISO885916
MULE_INTERNAL	Mule internal code	Multilingual Emacs	Yes	No	1–4	-
SJIS	Shift JIS	Japanese	No	No	1–2	Mskanji, ShiftJIS, WIN932, and Windows932
SHIFT_JIS_2004	Shift JIS, JIS X 0213	Japanese	No	No	1–2	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Language	Server-side Encoding	ICU Support	Number of Bytes/ Characters	Alias
SQL_ASCII	Unspecified (see the text)	*Any*	Yes	No	1	-
UHC	Unified Hangul Code	Korean	No	No	1–2	WIN949, Windows949
UTF8	Unicode, 8-bit	*All*	Yes	Yes	1–4	Unicode
WIN866	Windows CP866	Cyrillic	Yes	Yes	1	ALT
WIN874	Windows CP874	Thai	Yes	No	1	-
WIN1250	Windows CP1250	Central European languages	Yes	Yes	1	-
WIN1251	Windows CP1251	Cyrillic	Yes	Yes	1	WIN
WIN1252	Windows CP1252	Western European languages	Yes	Yes	1	-
WIN1253	Windows CP1253	Greek	Yes	Yes	1	-
WIN1254	Windows CP1254	Turkish	Yes	Yes	1	-
WIN1255	Windows CP1255	Hebrew	Yes	Yes	1	-
WIN1256	Windows CP1256	Arabic	Yes	Yes	1	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Description	Language	Server-side Encoding	ICU Support	Number of Bytes/ Characters	Alias
WIN1257	Windows CP1257	Baltic languages	Yes	Yes	1	-
WIN1258	Windows CP1258	Vietnamese	Yes	Yes	1	ABC, TCVN, TCVN5712, VSCII
ZHS16GBK	Extended national standards	Simplified Chinese	Yes	No	1–2	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- Not all client APIs support the character encodings listed in **Table 1-395**.
- The setting of the SQL_ASCII encoding format is different from that of other encoding formats. If the character encoding of the server is SQL_ASCII, the server parses the byte values 0 to 127 according to the ASCII standard. The byte values 128 to 255 are regarded as the characters that cannot be parsed. If any non-ASCII data is used, do not set the character encoding to **SQL_ASCII**. If the character encoding is set to **SQL_ASCII**, the database cannot convert or verify non-ASCII characters.
- The newly specified database character encoding must be compatible with the local settings (**LC_COLLATE** and **LC_CTYPE**).
- When the specified character encoding is **GBK** or **ZHS16GBK**, some uncommon Chinese characters cannot be directly used as object names. This is because when the second byte of the GBK ranges from 0x40 to 0x7E, the byte encoding overlaps with the ASCII characters, including uppercase characters (A–Z), lowercase characters (a–z), and special characters @ [\ ] ^ _ ` { | } @, [, \, ], ^, _, `, {, |, and } are operators in the database. If they are directly used as object names, a syntax error will be reported.
- If the encoding format on the client is different from that on the server, the database must support encoding format conversion. You can view all supported encoding formats in the PG_CONVERSION system catalog. If the database does not support encoding format conversion, it is recommended that the client-side encoding be the same as the server-side encoding. You can change the client-side encoding by setting the GUC parameter **client_encoding**.
- If both the database character set encoding and client-side encoding are GB18030_2022, ensure that the client OS supports GB18030_2022. Otherwise, data inconsistency may occur due to incomplete compatibility between GB18030 character set versions. In addition, if historical data needs to be switched to the GB18030_2022 database, follow the database switchover process to migrate data.
- If the specified character encoding is **ZHS16GBK**, the euro sign (€) can be input, and the encoding value is **0x80**.

- **LC_COLLATE [=] 'lc_collate'**

  (Optional) Specifies the character set used by the new database.

  Value range: character sets supported by the OS. By default, the character set of the template database is used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- The use of this parameter affects the sorting order of strings, such as the sorting of the ORDER BY operation results or the sorting of indexes on text columns.
- This parameter does not take effect for M-compatible databases.
- The value ranges of the **lc_collate** and **lc_ctype** parameters are related to the character sets supported by the local environment. For example, in the Linux OS, you can run the **locale -a** command to obtain the list of character sets supported by the OS. When the **LC_COLLATE** and **LC_CTYPE** keywords are specified, select the required character set and character type from the obtained character set list.
- If you want to set the character encoding to **GB18030_2022**, ensure that the value ranges of **lc_collate** and **lc_ctype** are the same as those of GB18030.
- If you want to set the character encoding to **ZHS16GBK**, ensure that the value ranges of **lc_collate** and **lc_ctype** are the same as those of GBK character encoding.

- **LC_CTYPE [=] 'lc_ctype'**

  (Optional) Specifies the character type used by the new database.

  Value range: character type supported by the OS. By default, the character type of the template database is used.

NOTICE

- The use of this parameter affects the classification of characters, such as uppercase letters, lowercase letters, and digits.
- This parameter does not take effect for M-compatible databases.
- The value ranges of the **lc_collate** and **lc_ctype** parameters are related to the character sets supported by the local environment. For example, in the Linux OS, you can run the **locale -a** command to obtain the list of character sets supported by the OS. When the **LC_COLLATE** and **LC_CTYPE** keywords are specified, select the required character set and character type from the obtained character set list.
- If you want to set the character encoding to **GB18030_2022**, ensure that the value ranges of **lc_collate** and **lc_ctype** are the same as those of GB18030.
- If you want to set the character encoding to **ZHS16GBK**, ensure that the value ranges of **lc_collate** and **lc_ctype** are the same as those of GBK character encoding.

- **DBCOMPATIBILITY [=] 'compatibility_type'**

  (Optional) Specifies the compatible database type. If this parameter is not specified, the default compatibility mode is affected by the **TEMPLATE** keyword and the GUC parameter **distribute_default_compatibility**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: **MYSQL**, **TD**, **ORA**, **PG**, and **M**, indicating compatibilities with MySQL, Teradata, Oracle, PostgreSQL, and M-compatible databases, respectively.

NOTICE

- In an ORA-compatible database, empty strings are treated as **NULL**. When the GUC parameter **mapping_date_to_datea** parameter is disabled, the data type **DATE** is replaced by **TIMESTAMP(0) WITHOUT TIME ZONE**. When the GUC parameter **mapping_date_to_datea** is enabled, the data type **DATE** is replaced by **DATEA**.
- When a character string is converted to an integer, if the input is invalid, the MySQL-compatible database converts the input to 0. In other compatible databases, an error is reported.
- In PG-compatible databases, CHAR and VARCHAR are counted by character. In other compatible databases, CHAR and VARCHAR are counted by byte. For example, for the UTF-8 character encoding, CHAR(3) can store three Chinese characters in PG-compatible databases, but can store only one Chinese character in other compatible databases.
- If an error occurs during statement execution, some error messages may display "A-format" or "B-format", which represent "ORA-format" and "MYSQL-format", respectively.
- For details about M-compatible database features, see *M Compatibility Developer Guide*.

- **TABLESPACE [=] tablespace_name**

  (Optional) Specifies the tablespace of a database.

  Value range: an existing tablespace name.

- **CONNECTION LIMIT [=] connlimit**

  (Optional) Specifies the maximum number of concurrent connections that can be made to the database.

  Value range: an integer in the range [–1,2^{31} – 1]. The default value is **-1**, indicating that there is no limit.

NOTICE

- System administrators are not restricted by this parameter.
- **connlimit** of each node is calculated separately. The number of connections in a cluster is calculated using the following formula: Number of connections in a cluster = **connlimit** x Number of normal CNs.

- **DBTIMEZONE [=] 'time_zone'**

  Specifies the time zone of the new database. This parameter affects the time zone of the new database. The default value is **PRC**.

  The options are as follows:

  – Time zone names and abbreviations supported by the OS.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- [–15:59,+15:00]

> **NOTE**
>
> - This parameter is supported only when the ORA-compatible mode is used and **a_format_version** is set to **'10c'** and **a_format_dev_version** is set to **'s2'**.
> - The restrictions on character encoding are as follows:
>   - If the locale is set to **C** or **POSIX**, all encoding types are allowed. For other locale settings, the character encoding must be the same as that of the locale.
>   - The character encoding and locale settings must match the template database (except when **template0** is used as the template). When the database contains data that does not match the specified encoding or contains indexes whose sorting order is affected by **LC_COLLATE** and **LC_CTYPE**, copying data may result in the invalidation of indexes in the new database.

Examples

```
-- Create users jim and tom.
gaussdb=# CREATE USER jim PASSWORD '********';

-- Create the GBK-encoded database testdb1.
gaussdb=# CREATE DATABASE testdb1 ENCODING 'GBK' template = template0;

-- View information about the testdb1 database.
gaussdb=# SELECT datname,pg_encoding_to_char(encoding) FROM pg_database WHERE datname =
'testdb1';
 datname | pg_encoding_to_char
---------+---------------------
 testdb1 | GBK
(1 row)

-- Create the ORA-compatible database testdb2 and specify jim as the owner.
gaussdb=# CREATE DATABASE testdb2 OWNER jim DBCOMPATIBILITY = 'ORA';

-- View the information about testdb2.
gaussdb=# SELECT t1.datname,t2.usename,t1.datcompatibility
          FROM pg_database t1,gs_user t2
          WHERE t1.datname = 'testdb2' AND t1.datdba=t2.usesysid;
 datname | usename | datcompatibility
---------+---------+------------------
 testdb2 | jim     | ORA
(1 row)
gaussdb=# CREATE DATABASE testdb_tmp OWNER jim template=templatea;

-- View the information about testdb2.
gaussdb=# SELECT t1.datname,t2.usename,t1.datcompatibility
                  FROM pg_database t1,gs_user t2
                  WHERE t1.datname = 'testdb_tmp' AND t1.datdba=t2.usesysid;
  datname   | usename | datcompatibility
------------+---------+------------------
 testdb_tmp | jim     | ORA
(1 row)

-- Create a database that 1000 concurrent connections can be made.
gaussdb=# CREATE DATABASE testdb4 CONNECTION LIMIT=1000;

-- View the information about testdb4.
gaussdb=# SELECT datname,datconnlimit  FROM pg_database where datname = 'testdb4';
 datname | datconnlimit
---------+--------------
 testdb4 |         1000
(1 row)

-- Create a tablespace.
gaussdb=# CREATE TABLESPACE tbs_test RELATIVE LOCATION 'test_tablespace/tbs_test';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a database and specify tbs_test as the default tablespace.
gaussdb=# CREATE DATABASE testdb5 TABLESPACE = tbs_test;

-- View the information about testdb5.
gaussdb=# SELECT db.datname,ts.spcname FROM pg_database db JOIN pg_tablespace ts ON
db.dattablespace = ts.oid WHERE db.datname = 'testdb5';
 datname | spcname
---------+----------
 testdb5 | tbs_test
(1 row)

-- Switch to the ORA-compatible database testdb2 and set session parameters.
gaussdb=# \c testdb2
testdb2=# SET a_format_version='10c';
testdb2=# SET a_format_dev_version='s2';

-- Create the ORA-compatible database and specify the time zone.
testdb2=# CREATE DATABASE testdb3 DBCOMPATIBILITY 'ORA' DBTIMEZONE='+08:00';

-- View the information about testdb3.
testdb2=# SELECT datname,datcompatibility,dattimezone FROM pg_database WHERE datname = 'testdb3';
 datname | datcompatibility | dattimezone
---------+------------------+-------------
 testdb3 | ORA              | +08:00
(1 row)

-- Switch to the initial database. Change the database name based on actual situation.
testdb2=# \c postgres

-- Drop databases.
gaussdb=# DROP DATABASE testdb1;
gaussdb=# DROP DATABASE testdb2;
gaussdb=# DROP DATABASE testdb3;
gaussdb=# DROP DATABASE testdb4;
gaussdb=# DROP DATABASE testdb5;
gaussdb=# DROP DATABASE testdb_tmp;

-- Drop the user.
gaussdb=# DROP USER jim;

-- Drop the tablespace.
gaussdb=# DROP TABLESPACE tbs_test;
```

Helpful Links

ALTER DATABASE and **DROP DATABASE**

1.13.9.17 CREATE DATABASE LINK

Description

Creates a database link object.

Precautions

- For details about database links, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.
- The database link feature can be used only in ORA-compatible databases.
- Do not use database links to connect to the initial user. The initial user is not allowed to create, modify, or delete database link objects.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- When a database link is created, the system does not check whether the link can be successfully connected. If related keywords are missing, an error may be reported when the link is used.
- If the upgrade is not committed, the database link cannot be created.
- In non-multi-primary mode, you can use the local IP address and port to connect to the current database instance. CNs and DNs may be deployed on different servers. Therefore, you are advised not to set the **host** parameter to **127.0.0.1** or **localhost**. Otherwise, the connection may fail.

Syntax

```
CREATE [ PUBLIC ] DATABASE LINK dblink
  [ CONNECT TO { CURRENT_USER | 'user' IDENTIFIED BY 'password' } ] [ USING ( {option 'value'} [, ...]) ];
```

option is as follows:

```
{ HOST | PORT | DBNAME | FETCH_SIZE }
```

Parameters

- **PUBLIC**

  Creates a public database link visible to all users. If this parameter is not specified, the database connection is private and available only to the current user.

- **dblink**

  Specifies the name of the database link to be created.

- **user**

  Specifies the name of the user for connecting to the remote database through the created database link.

- **password**

  Specifies the password of the user for connecting to the remote database through the created database link.

- **CURRENT_USER**

  Uses the initial username and empty password of the current database for connection.

  NOTE

  In the current version, after the **CURRENT_USER** option is specified, the database link can be created, but it is not connected.

- **USING ({option 'value'} [, ...])**

  (Optional) Specifies parameters such as the IP address, port number, and remote database name of the database to be connected. The supported options are as follows:

  - **host**: specifies the IP address of the database to be connected. If this parameter is not specified, it is left empty by default. You can specify multiple IP addresses and separate them with commas (,), except for IP addresses of encrypted databases, SSL settings, and certificate authentication.
  - **port**: specifies the port number for connection. If this parameter is not specified, the default value **5432** is used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **dbname**: specifies the name of the database to be connected. If not specified, the username used for connecting to the remote end is used by default.
- **fetch_size**: specifies the amount of data obtained from the remote end each time. The value range is [0,2147483647]. The default value is **100**.

Examples

```
-- Create an ORA-compatible database.
gaussdb=# CREATE DATABASE ora_test_db DBCOMPATIBILITY 'ORA';
CREATE DATABASE

-- Switch to another database.
gaussdb=# \c ora_test_db

-- Create a user with the SYSADMIN permission.
ora_test_db=# CREATE USER local_user WITH SYSADMIN PASSWORD '********';
CREATE ROLE
ora_test_db=# SET ROLE local_user PASSWORD '********';
SET

-- Create a private database link. host can be an IPv6 address.
ora_test_db=> CREATE DATABASE LINK private_dblink CONNECT TO 'remote_user' IDENTIFIED BY '********'
USING (host '***.***.***.***',port '*****',dbname 'remote_db');
CREATE DATABASE LINK

-- Drop the private database link.
ora_test_db=> DROP DATABASE LINK private_dblink;
DROP DATABASE LINK

-- Create a public database link. host can be an IPv6 address.
ora_test_db=> CREATE PUBLIC DATABASE LINK public_dblink CONNECT TO 'remote_user' IDENTIFIED BY
'********' USING (host '***.***.***.***',port '*****',dbname 'remote_db');
CREATE DATABASE LINK

-- Drop the public database link.
ora_test_db=> DROP PUBLIC DATABASE LINK public_dblink;
DROP DATABASE LINK

-- Drop the created user.
ora_test_db=> RESET ROLE;
RESET
ora_test_db=# DROP USER local_user;
DROP ROLE

-- Switch back to the initial database and drop the test database. Replace postgres with the actual
database name.
ora_test_db=# \c postgres
gaussdb=# DROP DATABASE ora_test_db;
DROP DATABASE
```

Helpful Links

ALTER DATABASE LINK and **DROP DATABASE LINK**

1.13.9.18 CREATE DIRECTORY

Description

Creates a directory, which can define an alias for a path in the server file system and is used to store data files used by users. Users can read and write these files through the DBE_FILE advanced package. You can view existing directory objects in the PG_DIRECTORY catalog.

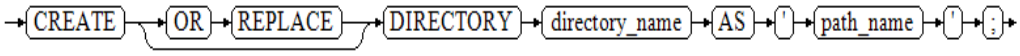
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

- The default owner of a directory object is the user who creates the directory object. By default, the user who creates the directory object has the permissions to read and write the directory object. When **enable_access_server_directory** is set to **off**, only the initial user is allowed to create the directory. When **enable_access_server_directory** is set to **on**, users with the SYSADMIN permission and users inheriting the permissions of the built-in role gs_role_directory_create can create directories.
- In the database scenario where the database is deployed by users, the initial configuration and management of the permission system are performed by users. When granting directory permissions, ensure that the target directory meets the following security conditions: The security baseline check is passed, the access control audit mechanism is available, and sensitive data is classified and graded. The permission configuration should comply with the least privilege principle, and a permission approval and tracing mechanism should be established.
- The READ and WRITE operation permissions can be granted to specified users on a directory object for DBE_FILE permission control. Create directory objects properly. If a directory object is created in a key directory (for example, the root directory) of the OS, unexpected security risks may occur when the configuration in the directory file is modified.

Syntax

```
CREATE [OR REPLACE] DIRECTORY directory_name
    AS 'path_name';
```



Parameters

- **directory_name**

  Specifies the name of the directory object to be created.

  Value range: character strings complying with **Identifier Naming Conventions**.

  NOTE

  The following validity check is performed during directory creation:

  - Check whether the path exists in the OS. If it does not exist, a message is displayed, indicating the potential risks.
  - Check whether the database initial user (for example, **omm**) has the read, write, and execute permissions on the created directory. If the user does not have all the permissions, a message is displayed, indicating the potential risks.

- **path_name**

  Specifies the directory path in the server file system.

  Value range: a valid OS path.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- In a cluster, ensure that the specified directory paths are the same on all the nodes. Otherwise, the path may fail to be matched on some nodes when the directory is used.
- The following directories cannot be created:
  - The path contains special characters.
  - The path is a relative path.

Examples

```
-- Create a directory.
gaussdb=# CREATE OR REPLACE DIRECTORY dir AS '/tmp/';

-- View the created directory.
gaussdb=# select * from pg_directory;
 dirname | owner | dirpath | diracl
---------+-------+---------+--------
 dir     |    10 | /tmp    |
(1 row)

-- Drop the directory.
gaussdb=# DROP DIRECTORY dir;
```

Helpful Links

ALTER DIRECTORY and **DROP DIRECTORY**

1.13.9.19 CREATE EXTENSION

NOTICE

Extensions are for internal use only. Only advanced packages or extensions in the database can be created (installed). Currently, only postgis, postgis_sfcgal, postgis_raster, postgis_topology, yukon_geomodel, yukon_geogridcoder, pkg_bpchar_opc, and tablefunc can be created or installed.

Description

Installs an extension.

Precautions

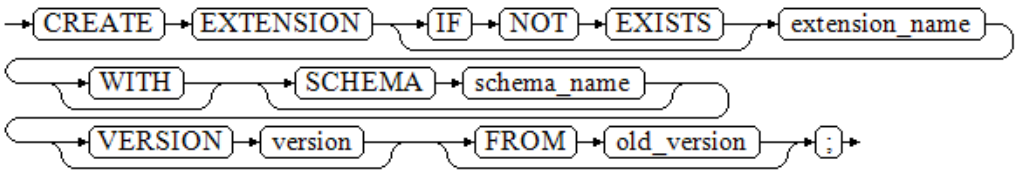
- Before running this command to install an extension to the database, you must install the files that the extension depends on.
- No duplicate extension is allowed in a database. To avoid extension installation failures caused by name conflicts, ensure that the target extension name is not occupied before installing the extension.
- If a database object (package, synonym, operator, directory, function, stored procedure, view, or table) with the same name as the extension to be installed exists in the database, the installation will fail.
- Installing an extension is to execute the corresponding extension script file. The script will create a new SQL entity, including but not limited to functions, data types, operators, and index-supported methods.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Installing an extension requires the same permissions as creating the extension. For most extensions, this means that the database owner's permissions are required. For subsequent permission checks and entities created by the extension script, the role that runs the **CREATE EXTENSION** command becomes the owner of the extension.
- By default, direct creation of database extensions is prohibited because extensions may cause unexpected errors and compatibility issues after database upgrades. To create an extension, set **enable_extension** to **true**.
- When you create an extension, if the GUC parameter **enable_object_special_character** is set to **off** and the extension script file uses the @extschema@ placeholder, the value of the **schema** parameter in the extension control file cannot contain any of the following special characters: [" $ ' \ ]

Syntax

```
CREATE EXTENSION [ IF NOT EXISTS ] extension_name
[ WITH ] [ SCHEMA schema_name ]
[ VERSION version ]
[ FROM old_version ];
```



Parameters

- **IF NOT EXISTS**

  Checks whether an extension with the same name exists when creating an extension.

  If the **IF NOT EXISTS** keyword is specified, the system checks whether an extension with the same name exists in the current database before creating an extension. If an extension with the same name exists, the system does not create the extension and returns a NOTICE message. If the **IF NOT EXISTS** keyword is not specified and an extension with the same name exists, the system returns an ERROR alarm.

- **extension_name**

  Specifies the name of the extension to be installed. The database creates the extension by using the information in the **SHAREDIR/extension/extension_name.control** file.

- **schema_name**

  Specifies the schema to which the objects (such as functions and data types) created by the script file of the extension belong. The specified schema must exist. If the schema to which the objects created by the script file of the extension belong is not specified, and no schema is either specified in the control file of the extension, the default schema is used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> The files on which the extension depends (that is, the extension itself) do not belong to any schema. The name of the extension is unique in the database and is not restricted by the schema. However, the objects (such as functions and data types) created by the extension belong to a specific schema.

- **version**

  Specifies the version of the extension to install. It can be written as either an identifier or a string. The default version is whatever is specified in the extension's control file.

- **old_version**

  You need to specify **FROM old_version** only if the content that does not exist in the existing module needs to be upgraded. This option uses the **CREATE EXTENSION FROM old_version** command to run a script for installing new content into the extension of an existing module instead of creating an object. Note that **Schema** should be specified as the schema that includes these existing objects.

Examples

Install an extension in the current database. For example, install **security_plugin**:

```
-- Before installing the extension, set enable_extension to true.
gaussdb=# SET enable_extension = true;

-- Install the expansion.
gaussdb=# CREATE EXTENSION IF NOT EXISTS security_plugin;

-- Drop the extension.
gaussdb=# DROP EXTENSION security_plugin;
```

1.13.9.20 CREATE FOREIGN DATA WRAPPER

Description

Creates a foreign data wrapper.

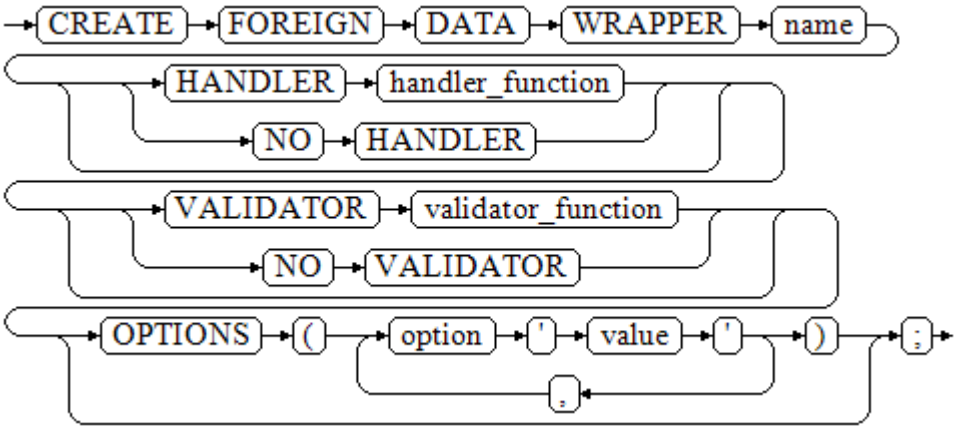
Precautions

A foreign data wrapper can be created only during the internal initialization phase or within an upgrade script. In other scenarios, users cannot directly use this syntax.

Syntax

```
CREATE FOREIGN DATA WRAPPER name
[ HANDLER handler_function | NO HANDLER ]
[ VALIDATOR validator_function | NO VALIDATOR ]
[ OPTIONS ( { option 'value' } [, ... ] ) ];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- **name**

  Specifies the name of the foreign data wrapper to be created.

  The name of the foreign data wrapper must be unique in the database.

- **HANDLER handler_function**

  handler_function is the name of a registered function used to retrieve handler functions for foreign tables. The handler function must have no parameter, and its return type must be fdw_handler.

  You can create foreign data wrappers without using a handler function. However, foreign tables using such foreign data wrappers can only be declared and cannot be accessed.

- **NO HANDLER**

  When you create a foreign data wrapper without specifying (or using) a handler function by specifying **NO HANDLER**, the foreign tables of the foreign data wrapper can only be declared and cannot be accessed.

- **VALIDATOR validator_function**

  validator_function is the name of a registered function used to check the common validation options provided to the foreign data wrapper, as well as the validation options for the foreign server, user mapping, and foreign table that use the foreign data wrapper. If there is no validator option (function) or **NO VALIDATOR** is declared, options are not checked at creation time (foreign data wrappers may ignore or reject invalid option descriptions at runtime).

  The validator option (function) **validator_function** must accept two parameters:

  – An array of options stored in the system directory. The value is of the text[] type.

  – OID of the system directory that contains these options. The value is of the oid type.

- **NO VALIDATOR**

  The validator function is not set during creation.

- **OPTIONS (option 'value' [, ...])**

  Declares options of the foreign data wrapper. **OPTIONS (option 'value' [, ...])** can be specific to each foreign data wrapper and is validated by the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

validator function in the foreign data wrapper. **OPTIONS** must meet the UNIQUE constraint.

Helpful Links

ALTER FOREIGN DATA WRAPPER and **DROP FOREIGN DATA WRAPPER**

1.13.9.21 CREATE FOREIGN TABLE

Description

In GaussDB, CREATE FOREIGN TABLE is a convenient mechanism for accessing, managing, and querying foreign data sources in a unified manner. By creating a foreign table, you can integrate data stored in foreign data sources, such as file systems, tables, and views, into GaussDB and perform query, join, and parallel operations in the same way as you perform operations on ordinary tables. Creates a GDS foreign table in the current database for parallel data import and export. The GDS foreign table can be read-only or write-only, used for parallel data import and export, respectively. The foreign table is read-only by default.

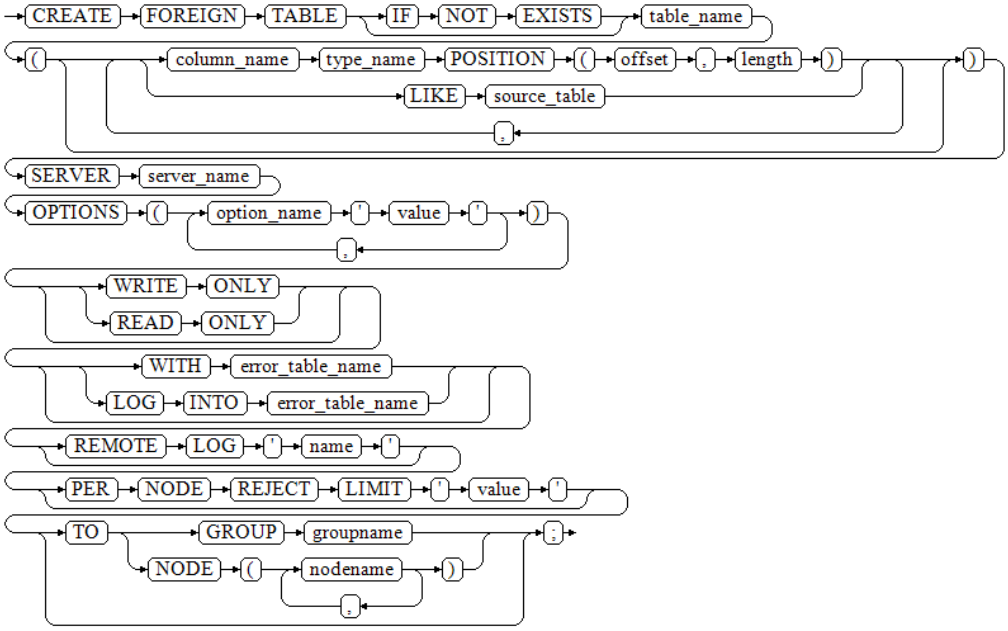
Precautions

- The user who creates the foreign table will be the owner of the foreign table.
- The distribution mode of a GDS foreign table does not need to be explicitly specified. The default mode is **ROUNDROBIN**.
- All constraints (including column and row constraints) are invalid to the GDS foreign table.
- When multi-layer quotation marks are used for sensitive columns (such as **password** and **secret_access_key**) in **OPTIONS**, the semantics is different from that in the scenario where quotation marks are not used. Therefore, sensitive columns are not identified for anonymization.
- When executing **GDS** to import data to a base table with a GSI, the **enable_stream_operator** parameter must be enabled to achieve optimal data import performance.
- When GDS is used to import or export data, the data cannot contain **\0** character. Otherwise, the data will be truncated.

Syntax

```
CREATE FOREIGN TABLE [ IF NOT EXISTS ] table_name
   ( [ { column_name type_name  POSITION ( offset, length ) | LIKE source_table } [, ...] ] )
   SERVER server_name
   OPTIONS ( { option_name 'value' } [, ...] )
   [ WRITE ONLY | READ ONLY  ]
   [ WITH error_table_name | LOG INTO error_table_name]
   [REMOTE LOG 'name']
   [PER NODE REJECT LIMIT 'value']
   [ TO { GROUP groupname | NODE ( nodename [, ... ] ) } ];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameter Overview

The syntax for creating a foreign table provides multiple parameters. **Table 1-396** describes the common parameters.

Table 1-396 Foreign table parameters

Required/ Optional	Parameter	
Required	• **table_name** • **column_name** • **type_name** • **POSITION(offset,length)** • **LIKEsource_table** • **SERVER server_name** • **OPTIONS**	
Optional	Data source location parameter of a foreign table	**location**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Required/ Optional	Parameter	
	Data format parameters	• **format** • **header** (only for CSV and FIXED source data files) • **fileheader** (only for CSV and FIXED source data files) • **out_filename_prefix** • **delimiter** • **quote** (only for CSV source data files) • **escape** (only for CSV source data files) • **null** • **noescaping** (only for TEXT source data files) • **encoding** • **eol**
	Error-tolerance parameters	• **fill_missing_fields** • **ignore_extra_data** • **reject_limit** • **compatible_illegal_chars** • **WITH error_table_name** • **LOG INTO error_table_name** • **REMOTE LOG 'name'** • **PER NODE REJECT LIMIT 'value'**

Parameters

- **IF NOT EXISTS**

  Specifies whether a table with the same name exists. If a table with the same name already exists in the database, no error is reported. Instead, a message is displayed, indicating that the table relationship already exists.

- **table_name**

  Specifies the name of a foreign table.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **column_name**

  Specifies the name of a column in the foreign table.

  Value range: character strings complying with **Identifier Naming Conventions**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **type_name**

  Specifies the data type of the column.

- **POSITION (offset, length)**

  Defines the place of each column in the data file in fixed length mode. **offset** is the start of the column in the source file, and **length** is the length of the column.

  Value range: **offset** must be greater than 0, and its unit is byte. The length of each record must be less than or equal to 1 GB. By default, columns not in the file are replaced with null.

- **LIKE source_table**

  Specifies a foreign table. A new foreign table automatically inherits all column names and their data types from this foreign table.

- **SERVER server_name**

  Specifies the server name of the foreign table. For the GDS foreign table, its server is created by the initial database, which is **gsmpp_server**.

  NOTE

  The name of the foreign-data wrapper corresponding to the server cannot be **file_fdw**.

- **OPTIONS ({ option_name ' value ' } [, ...])**

  Specifies all types of parameters of foreign table data.

  – location

    Specifies the data source location of the foreign table, which can be expressed through URLs or in local files. Multiple URLs and local files are separated by vertical bars (|).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- For a read-only foreign table imported by GDS from a remote server in parallel, its URL must end with its corresponding schema or file name. (Read-only is the default file attribute.) For example: **gsfs://192.168.0.90:5000/***, **file:///data/data.txt**, or **gsfs:// 192.168.0.90:5000/* | gsfs:// 192.168.0.91:5000/***.
- For a writable foreign table exported by GDS to a remote server in parallel, its URL does not need to contain a file name. If the data source is local, for example, **file:///data/**, only one data source in the foreign table can be specified, and the hosting directory must be created in advance on each node. If the data source location is a remote URL, for example, **gsfs:// 192.168.0.90:5000/**, multiple data sources can be specified. If the number of exported data file locations is less than or equal to the number of DNs, when you use the foreign table for export, the system traverses nodes and distributes them to the data source locations one by one. After a round of distribution is complete, the system re-distributes nodes from the beginning. Therefore, the more nodes there are, the more evenly the distribution is. If the number of exported data file locations is greater than the number of DNs, when you export data, the data is evenly distributed to data source locations corresponding to the DNs. Blank data files are created on the excess data source locations.
- For a read-only foreign table imported by GDS from a remote server in parallel, the number of URLs must be less than the number of DNs, and URLs in the same location cannot be used.
- If the URL begins with **gsfss://**, data is imported and exported in encrypted mode, and concurrency cannot exceed 10.
- If SSL is used for encrypted transmission and the certificate is replaced, GDS must be restarted.

– format

Specifies the format of the data source file in a foreign table.

Value range: **CSV**, **TEXT**, and **FIXED**. The default value is **TEXT**.

- **CSV**: Escape sequences are processed as common strings. Therefore, newline characters are processed as data.
- **TEXT**: Escape sequences are processed as they are. Therefore, newline characters are not processed as data.
- **FIXED**: The column length of each record is the same, and spaces are used for padding.

NOTE

- An escape sequence is a string starting with a backslash (\), including **\b** (backspace), **\f** (form feed), **\n** (new line), **\r** (carriage return), **\t** (horizontal tab), **\v** (vertical tab), *****digit* (octal number), and **\x***digit* (hexadecimal number). In TEXT files, strings are processed as they are. In files of other formats, strings are processed as data.
- **POSITION** must be specified for each column when **FIXED** is used. **FIXED** is defined as follows:
  - The column length of each record is the same.
  - Spaces are used for column padding. Columns of the numeric type are left-aligned and columns of the string type are right-aligned.
  - No delimiters are used between columns.

– header

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies whether a file contains a header with the information about each column in the file. **header** is available only for CSV and FIXED files.

- When data is imported, if **header** is set to **true** or **on**, the first row of the data text is identified as the header and ignored. If **header** is set to **false** or **off**, the first row is identified as data.
- When data is exported, if **header** is set to **true** or **on**, **fileheader** must be specified. **fileheader** specifies the format of the exported header file. If **header** is set to **false** or **off**, the exported data file does not contain the header.

Value range: **true**/**on** or **false**/**off**. The default value is **false**/**off**.

– fileheader

Specifies a file that defines the content in the header for exported data. The file contains one row of data description of each column.

For example, to add a header in a file containing product information, define the file as follows: **The information of products.\n**

NOTE

- This parameter is available only when **header** is **true** or **on**. The file must be prepared in advance.
- In **REMOTE** mode, the definition file must be put to the working directory of the GDS (the **-d** directory specified when the GDS is started).
- In **LOCAL** mode, the definition file must be put to the same path of each node, and **fileheader** specifies the absolute path.
- The file can contain only one row of header information, and end with a newline character. Excess rows will be discarded. (Header information cannot contain newline characters).
- The length of the file including newline characters cannot exceed 1 MB.

– out_filename_prefix

Specifies the name prefix of the data file exported using GDS from a write-only foreign table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The file name prefix must be valid and compliant with the restrictions of the file system in the physical environment where the GDS is deployed. Otherwise, the file will fail to be created.
  - The prefix of the exported file can contain lowercase letters, uppercase letters, digits, and underscores (_) but must not contain invalid characters, such as / ? * : | \\ < > @ # $ & ( ) + -
  - The prefix of the exported file cannot contain feature fields reserved for Windows and Linux, including but not limited to con, aux, nul, prn, com0, com1, com2, com3, com4, com5, com6, com7, com8, com9, lpt0, lpt1, lpt2, lpt3, lpt4, lpt5, lpt6, lpt7, lpt8, and lpt9.
  - The total length of the absolute path consisting of the exported file prefix, the path specified by **gds –d**, and **.dat.x** should be as required by the file system where GDS is deployed.
  - It is required that the prefix can be correctly parsed and identified by the receiver (including but not limited to the original database where it was exported using GDS) of the data file. Identify and modify the option that causes the file name resolution problem (if any).
- To concurrently perform export jobs, do not use the same file name prefix for them. Otherwise, the exported files may overwrite each other or be lost in the OS or file system.

– delimiter

Specifies the column delimiter of data. If no delimiter is specified, the default one will be used. The default delimiter in the TEXT format is a tab and that in the CSV format is a comma (,). No delimiter is used in the FIXED format.

NOTE

- The value of **delimiter** cannot be **\r** or **\n**.
- A delimiter cannot be the same as the null value. The delimiter for the CSV format cannot be same as the **quote** value.
- In TEXT format, the delimiter cannot contain backslashes (\), dots (.), letters (a to z), or digits (0 to 9).
- The data length of a single row should be less than 1 GB. A row that has many columns using long delimiters cannot contain much valid data.
- You are advised to use a multi-character or invisible characters, such as 0x07, 0x08, and 0x1b as the delimiter.

Value range: multi-character delimiters within 10 bytes.

– quote

Specifies a quote character in a CSV data source file. The default value is double quotation marks (").

NOTE

- The value of **quote** cannot be the same as that of the **delimiter** or **null** parameter.
- The value of **quote** must be a single-byte character.
- You are advised to use invisible characters (such as 0x07, 0x08, and 0x1b) as quotes.

– escape

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies an escape character in a CSV data source file. Escape characters can only be single-byte characters. The default value is double quotation marks (").

If the escape character is the same as the quote character, the escape character is replaced with the **\0** character.

- null

  Specifies the null value in a data source file.

  NOTE

  - A null value cannot be **\r** or **\n**. The maximum length is 100 characters.
  - A null value cannot be the same as the **delimiter** or **quote** value.

  Value range:

  - The default value for the TEXT format is **\n**.
  - The default value for the CSV format is an empty string without quotation marks.

- noescaping

  Specifies that the backslash (\) and its following characters are not escaped in the TEXT format.

  NOTE

  noescaping is available only for TEXT source data files.

  Value range: **true**/**on** or **false**/**off**. The default value is **false**/**off**.

- encoding

  Specifies the encoding of a data source file, that is, the encoding used to parse, check, and generate a data source file. Its default value is the default **client_encoding** value of the current database.

  Value range: See **ENCODING [=] 'encoding'** in section **CREATE DATABASE**.

  NOTE

  - When importing a foreign table, you are advised to set **encoding** to the encoding format of the file or set **client_encoding** based on the character set of the file before the import. Otherwise, redundant parsing, verification errors, import errors, or rollback may occur, and even invalid data may be imported to the database.
  - When exporting a foreign table, you are advised to specify this option to avoid unexpected results caused by using the default character set.
  - If the **encoding** option is not specified when you create a foreign table, a warning message is displayed on the client.
  - Currently, GDS cannot parse or write in a file using multiple encoding formats during foreign table import or export, respectively.

- fill_missing_fields

  Specifies how to handle the missing of the last column in a row of a data source file during data import.

  Value range: **true**/**on** or **false**/**off**. The default value is **false**/**off**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **true** or **on**: If the last column in a row of a data source file is missing during data import, the column is set to **NULL** and no import error message is reported.
- **false** or **off**: If the last column in a row of a data source file is missing during data import, the following error message is reported:
  ```
  missing data for column "tt"
  ```

– ignore_extra_data

Specifies whether to ignore the extra columns in the source file if the number of columns in the source file is greater than that defined in the foreign table. This parameter is used only during data import.

Value range: **true**/**on** or **false**/**off**. The default value is **false**/**off**.

- **true**/**on**: If the number of columns in a data source file is greater than that defined by the foreign table, the extra columns at the end of a row are ignored.
- **false** or **off**: If the number of columns in a data source file is greater than that defined by the foreign table, the following error message is reported:
  ```
  extra data after last expected column
  ```

NOTE

If a newline character at the end of a row is missing and two rows of data are combined into one row, setting this parameter to **true** or **on** will cause the data in the latter row to be ignored.

– reject_limit

Specifies the maximum number of data format errors allowed during data import. If the number of errors does not reach the threshold, the data import is considered successful.

Value range: an integer or **unlimited**. The default value is **0**, indicating that an error message is returned immediately. Enclose positive integers with single quotation marks (').

NOTE

- You are advised to use PER NODE REJECT LIMIT 'value' to replace this syntax.
- Data format errors include the following: a column is lost, an extra column exists, a data type is incorrect, and encoding is incorrect. Once a non-data-format error occurs, the whole data import process is stopped.

– mode

Specifies the data import mode during a specific data import process.

Value range:

- **Normal** (default): supports all file formats (CSV, TEXT, FIXED). You need to enable GDS to help data import.
- **Shared**: supports the TEXT format. It does not need assistance from GDS, but requires that all the user data be added to the same path on all nodes through NFS.
- **Private**: User data has been stored in the same path under the local directory of the DN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If the shared or private mode is used, the creation of a local foreign table is controlled by the GUC parameters **enable_copy_server_files** and **safe_data_path**.
- When the GUC parameter **enable_copy_server_files** is disabled, only the initial user is allowed to create a local foreign table in shared or private mode. When **enable_copy_server_files** is enabled, users with the SYSADMIN permission or users inherited the permission of the built-in gs_role_copy_files role are allowed to create a local foreign table, preventing unauthorized users from viewing or modifying sensitive files.
- Database administrators can define an import/export path whitelist for all users by setting the GUC parameter **safe_data_path**. The file path must be a subpath of **safe_data_path**. If **safe_data_path** is left blank, no interception will occur on the path being used.

– eol

Specifies the newline character style of the imported or exported data file.

Value range: multi-character newline characters within 10 bytes. Common newline characters include **\r** (0x0D), **\n** (0x0A), and **\r\n** (0x0D0A). Special newline characters include **$** and **#**.

NOTE

- The **eol** parameter supports only TEXT data import and export and does not support CSV or FIXED data import. For compatibility with earlier versions, the **eol** parameter can be set to **0x0D** or **0x0D0A** for CSV or FIXED data export.
- The value of **eol** cannot be the same as that of the delimiter or **null** parameter.
- The **eol** parameter cannot contain dots (.), letters (a to z), or digits (0 to 9).

– fix

Specifies the length of fixed-length data in each row, in bytes. This parameter is available only for read-only foreign tables.

Value range: less than 1 GB, and greater than or equal to the total length specified by **POSITION** (The total length is the sum of **offset** and **length** in the last column of the table definition.)

– out_fix_alignment

Specifies how the columns of the types BYTEAOID, CHAROID, NAMEOID, TEXTOID, BPCHAROID, VARCHAROID, NVARCHAR2OID, and CSTRINGOID are aligned during fixed-length export.

Value range: **align_left** and **align_right**. The default value is **align_right**.

NOTE

When importing byte data that must strictly comply with the hexadecimal or octal format, ensure that the data is left-aligned and does not contain leading spaces. Therefore, if the exported file needs to be imported using a GDS foreign table and the file data length is less than that specified by the foreign table formatter, the exported file must be left aligned. Otherwise, an error is reported during the import.

– date_format

Specifies the DATE format for data import. This syntax is available only for read-only foreign tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: valid DATE format. For details, see **Date and Time Processing Functions and Operators**.

NOTE

For a database in ORA-compatible mode:

- If the GUC parameter **mapping_date_to_datea** is set to **off**, the database creates the DATE type that will be implemented as the TIMESTAMP(0) WITHOUT TIME ZONE type. If you need to specify the format when importing data, see **timestamp_format**.
- If the GUC parameter **mapping_date_to_datea** is set to **on**, the database creates the DATE type that will be implemented like the DATEA type. If you need to specify the format when importing data, see **datea_format**.

– time_format

Specifies the TIME format for data import. This syntax is available only for read-only foreign tables.

Value range: a valid TIME value. Time zones cannot be used. For details, see **Date and Time Processing Functions and Operators**.

– timestamp_format

Specifies the TIMESTAMP format for data import. This syntax is available only for read-only foreign tables.

Value range: a valid TIMESTAMP value. Time zones cannot be used. For details, see **Date and Time Processing Functions and Operators**.

– datea_format

Specifies the DATEA format for data import. This syntax is available only for read-only foreign tables.

Value range: a valid DATEA value. Time zones are not supported. For details, see **Date and Time Processing Functions and Operators**.

– smalldatetime_format

Specifies the SMALLDATETIME format for data import. This syntax is available only for read-only foreign tables.

Value range: a valid SMALLDATETIME value. For details, see **Date and Time Processing Functions and Operators**.

– compatible_illegal_chars

Specifies whether to tolerate invalid characters during data import. This syntax is available only for read-only foreign tables.

Value range: **true**/**on** or **false**/**off**. The default value is **false**/**off**.

- **true** or **on**: No error message is reported and data import is not interrupted when there are invalid characters. Invalid characters are converted into valid ones, and then imported to the database.
- **false** or **off**: An error message is reported when there are invalid characters, and the import stops.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

The rules for converting invalid characters are as follows:

- Invalid characters are converted to the characters specified by the GUC parameter **convert_illegal_char_mode**. The default value is a question mark (?).
- Setting **compatible_illegal_chars** to **true** or **on** enables toleration of invalid characters. If **null**, **delimiter**, **quote**, and **escape** are set to spaces or question marks, error messages like "illegal chars conversion may confuse COPY escape 0x20" will be displayed to prompt you to modify parameters that may cause confusion, preventing importing errors.
- If the GUC parameter **copy_special_character_version** is set to **'no_error'**, invalid characters will not be checked during data import and query, and will be imported in the original format and displayed as garbled characters in query results. Therefore, exercise caution when enabling this parameter.
- The priority of the GUC parameter **copy_special_character_version** being set to **'no_error'** is higher than that of the GUC parameter **compatible_illegal_chars** being set to **true** or **on**.

- **READ ONLY**

  Specifies whether a foreign table is read-only. This parameter is available only for data import.

- **WRITE ONLY**

  Specifies whether a foreign table is write-only. This parameter is available only for data export.

- **WITH error_table_name**

  Specifies the table where data format errors generated during parallel data import are recorded. You can query the error information table after data is imported to obtain error details. This parameter is available only after **reject_limit** is set.

  Value range: character strings complying with **Identifier Naming Conventions**.

  NOTE

  To be compatible with PG open-source APIs, you are advised to replace this syntax with LOG INTO.

- **LOG INTO error_table_name**

  Specifies the table where data format errors generated during parallel data import are recorded. You can query the error information table after data is imported to obtain error details. This parameter is valid only when **PER NODE REJECT LIMIT 'value'** is set.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **REMOTE LOG 'name'**

  Specifies that the data format error information is saved as files in GDS. **name** is the prefix of the error data file.

- **PER NODE REJECT LIMIT 'value'**

  Specifies the maximum number of data format errors allowed on each DN during data import. If the number of errors exceeds the specified value on any DN, data import fails, an error is reported, and the system exits data import.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: an integer or **unlimited**. The default value is **0**, indicating that an error message is returned immediately.

NOTE

- This syntax specifies the error tolerance of a single node.
- Data format errors include the following: a column is lost, an extra column exists, a data type is incorrect, and encoding is incorrect. Once a non-data-format error occurs, the whole data scan process is stopped.

- **TO { GROUP groupname | NODE (nodename [, ...]) }**

  Currently, this syntax is not supported. It is only available for internal scale-out tools and should not be used by common users.

Examples

```
-- Create a foreign table to import data from GDS servers 192.168.0.90 and 192.168.0.91 in text format. Record errors that occur during data import to the err_HR_staffS table.
gaussdb=# CREATE FOREIGN TABLE foreign_HR_staffS
(
  staff_ID       NUMBER(6) ,
  FIRST_NAME     VARCHAR2(20),
  LAST_NAME      VARCHAR2(25),
  EMAIL          VARCHAR2(25),
  PHONE_NUMBER   VARCHAR2(20),
  HIRE_DATE      DATE,
  employment_ID  VARCHAR2(10),
  SALARY         NUMBER(8,2),
  COMMISSION_PCT NUMBER(2,2),
  MANAGER_ID     NUMBER(6),
  section_ID  NUMBER(4)
) SERVER gsmpp_server OPTIONS (location 'gsfs://192.168.0.90:5000/* | gsfs://192.168.0.91:5000/*', format 'TEXT', delimiter E'\x20',  null '') WITH err_HR_staffS;

-- Create a foreign table to import data from GDS servers 192.168.0.90 and 192.168.0.91 in text format and record error messages in the import process to the err_HR_staffS table. A maximum of two data format errors are allowed during the import.
gaussdb=# CREATE FOREIGN TABLE foreign_HR_staffS_ft3
(
  staff_ID       NUMBER(6) ,
  FIRST_NAME     VARCHAR2(20),
  LAST_NAME      VARCHAR2(25),
  EMAIL          VARCHAR2(25),
  PHONE_NUMBER   VARCHAR2(20),
  HIRE_DATE      DATE,
  employment_ID  VARCHAR2(10),
  SALARY         NUMBER(8,2),
  COMMISSION_PCT NUMBER(2,2),
  MANAGER_ID     NUMBER(6),
  section_ID  NUMBER(4)
) SERVER gsmpp_server OPTIONS (location 'gsfs://192.168.0.90:5000/* | gsfs://192.168.0.91:5000/*', format 'TEXT', delimiter E'\x20',  null '',reject_limit '2') WITH err_HR_staffS_ft3;

-- Create a foreign table to import all files in the input_data directory in CSV format.
gaussdb=# CREATE FOREIGN TABLE foreign_HR_staffS_ft1
(
  staff_ID       NUMBER(6) ,
  FIRST_NAME     VARCHAR2(20),
  LAST_NAME      VARCHAR2(25),
  EMAIL          VARCHAR2(25),
  PHONE_NUMBER   VARCHAR2(20),
  HIRE_DATE      DATE,
  employment_ID  VARCHAR2(10),
  SALARY         NUMBER(8,2),
  COMMISSION_PCT NUMBER(2,2),
  MANAGER_ID     NUMBER(6),
  section_ID     NUMBER(4)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
) SERVER gsmpp_server OPTIONS (location 'file:///input_data/*', format 'csv', mode 'private', delimiter ',')
WITH err_HR_staffS_ft1;

-- Create a foreign table to export data to the output_data directory in CSV format.
gaussdb=# CREATE FOREIGN TABLE foreign_HR_staffS_ft2
(
  staff_ID       NUMBER(6) ,
  FIRST_NAME     VARCHAR2(20),
  LAST_NAME      VARCHAR2(25),
  EMAIL          VARCHAR2(25),
  PHONE_NUMBER   VARCHAR2(20),
  HIRE_DATE      DATE,
  employment_ID  VARCHAR2(10),
  SALARY         NUMBER(8,2),
  COMMISSION_PCT NUMBER(2,2),
  MANAGER_ID     NUMBER(6),
  section_ID  NUMBER(4)
) SERVER gsmpp_server OPTIONS (location 'file:///output_data/', format 'csv', delimiter '|', header 'on')
WRITE ONLY;

-- Drop the foreign table.
gaussdb=# DROP FOREIGN TABLE foreign_HR_staffS;
gaussdb=# DROP FOREIGN TABLE foreign_HR_staffS_ft1;
gaussdb=# DROP FOREIGN TABLE foreign_HR_staffS_ft2;
gaussdb=# DROP FOREIGN TABLE foreign_HR_staffS_ft3;
```

Helpful Links

DROP FOREIGN TABLE

Suggestions

- delimiter
  - A delimiter cannot be **\r** or **\n**, or the same as the null value. The delimiter of CSV data cannot be same as the **quote** value.
  - The data length of a single row should be less than 1 GB. A row that has many columns using long delimiters cannot contain much valid data.
  - You are advised to use a multi-character or invisible characters, such as 0x07, 0x08, and 0x1b as the delimiter.
- quote
  - The value of **quote** cannot be the same as that of the **delimiter** or null parameter. The value must be a single-byte character.
  - You are advised to use invisible characters (such as 0x07, 0x08, and 0x1b) as quotes.
- mode Normal

  Supports all file formats (including CSV, TEXT, and FIXED). You need to enable GDS to help data import.
- mode Shared

  Supports the TEXT format. It does not need assistance from GDS, but requires that all the user data be mounted to the same path of the same nodes through NFS.
- mode Private

  User data has been stored in the same path under the local directory of the DN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.9.22 CREATE FUNCTION

Description

Creates a function.

Precautions

- Permission:
  - By default, the EXECUTE permission on a function is granted to the **PUBLIC** role when the function is created. For details, see **GRANT**. Since users inherit permissions from **PUBLIC** by default, they gain the ability to execute the function and view its definition. However, to execute the function, users must also possess the USAGE permission on the schema where the function is located. It is possible to revoke the default EXECUTE permission from **PUBLIC** during function creation and grant it to specific users instead. To avoid having a window where the new function is accessible to all, create it and set the permissions within a single transaction. When the database object isolation attribute is enabled, common users can only view the definitions of functions for which they have the EXECUTE permission.
  - Users with the CREATE ANY FUNCTION permission can create or replace functions in the user schemas.
  - Non-initial users do not have the permission to perform the REPLACE operation on the functions of the initial user.
  - If a definer function is created in the schema of another user, it will be executed with that user's permissions, which may lead to unauthorized operations. Exercise caution.
  - The default permission on a function is SECURITY INVOKER. To change the default permission to SECURITY DEFINER, set the GUC parameter **behavior_compat_options** to **'plsql_security_definer'**.
- Compatibility:
  - Oracle Database-compatible functions are created for ORA-compatible databases, and PostgreSQL-compatible functions are created for PG-compatible databases. Hybrid creation is not recommended.
  - Functions compatible with Oracle Database support viewing, exporting, and importing parameter comments, and viewing, exporting, and importing comments between IS/AS and plsql_body.
- Function creation:
  - An agg function or a system function cannot be nested in another avg function.
  - You cannot create a function that has the same name and parameter list as a stored procedure.
  - The default value of the input parameter cannot be the function itself.
  - If the **current_schema** or **search_path** is set using the SET statement in a function, the setting takes effect only for the function during its execution. After the function execution is complete, the value is automatically restored to the value before the function is called. That is, the **search_path** and **current_schema** set in the function are the same before and after the function is executed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - The type of the return value in the function is not checked. If the parameter or return value has precision, the precision is not checked.
  - In common cluster mode, return values, parameters, and variables cannot be set to the tables of the node groups that are not installed in the system by default. The internal statements of SQL functions cannot be executed on such tables.
- Function overloading:
  - Only the functions compatible with PG or those with the PACKAGE attribute can be overloaded. After **REPLACE** is specified, if the number of parameters, parameter type, or return value of a function is different, another function will be created instead of replacing the original one.
  - If a function supports overloading, you need to add the keyword PACKAGE. When an overloaded function is called, the variable type must be specified.
  - Overloaded functions cannot be created if they differ only in formal parameter names (that is, when function names and parameter list types are identical).
  - The following functions do not support function overloading: functions with only different return types, functions with only different default values, and functions with only different parameter forms in user-defined ref cursor and sys_refcursor.
- Function call:
  - Using an undeclared variable within a function will result in an error upon function calls. When a function without parameters is called in another function, you can directly call it by name (that is, you can omit the brackets).
  - The same function can be called via SELECT by specifying different parameters. The CALL syntax does not support calling identical functions without the PACKAGE attribute.
  - When the complex function is called, for example, func(x).a, a composite type is returned. Cross-schema call is supported, but the database.schema.package.func(x).b call is not supported.
  - In the RETURN statement of a function, when the constructor of the composite type is called, if the actual return type is different from the defined return type, the result can be implicitly converted to the defined return type. Cross-schema call, such as RETURN schema.record, is supported. Cross-database call, such as RETURN package.schema.record, is not supported. In the RETURN statement of a function, when the function is called, the function with the **OUT** parameter in an expression of an arithmetic operation is not supported, for example, **RETURN func(c out) + 1**.
  - A function with OUT parameters cannot be called in SQL statements (including the SELECT INTO syntax) and cannot be called in nested mode.
- GUC behavior control:
  - Functions contain syntax and functions that use GUC parameters to control features. If GUC parameters are modified in a session, the result may be inconsistent with the expected result. If the parameters are modified, the function may retain the behavior before the modification. Therefore, exercise caution when modifying GUC parameters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - If a function has output parameters, the GUC parameter **behavior_compat_options** must be set to **'proc_outparam_override'** for the output parameters to take effect. When the function is called using SELECT or CALL, an argument must be provided in the position of the output parameter. Otherwise, the function fails to be called.
  - When other functions with output parameters are called in a function, specifically an assignment expression, set the GUC parameter **behavior_compat_options** to **'proc_outparam_override'**, define variables of the same type as the output parameters in advance, and use the variables as output parameters to call other functions with output parameters for the output parameters to take effect. Otherwise, the output parameters of the called functions will be ignored.
  - After the GUC parameter **behavior_compat_options** is set to **'proc_outparam_override'**, if the return type of the function is setof, the OUT parameter will not take effect. If the return value of the function is of the TABLE type, you must provide the actual parameter of the TABLE type when calling the function (SELECT or CALL). Otherwise, the function fails to be called.
  - If the GUC parameter **behavior_compat_options** does not contain **'proc_outparam_override'**, the OUT and INOUT parameters of the function directly called by an anonymous block or stored procedure cannot be of the composite type, and the return value is used as the first value of the OUT parameter. As a result, the call fails. To correctly use the OUT and INOUT parameters, set the GUC parameter **behavior_compat_options** to **'proc_outparam_override'**.
  - When a stored procedure with the OUT parameter is called, you can set the GUC parameter **behavior_compat_options** to **'proc_outparam_transfer_length'** to pass the parameter length. The specifications and constraints are as follows:
    - The following types are supported: CHAR(n), CHARACTER(n), NCHAR(n), VARCHAR(n), VARYING(n), VARCHAR2(n), and NVARCHAR2(n).
    - If the **OUT** parameter does not take effect (for example, **perform**), the length does not need to be passed.
    - Precisions cannot be specified for the following types: NUMERIC, DECIMAL, NUMBER, FLOAT, DEC, INTEGER, TIME, TIMESTAMP, INTERVAL, TIME WITH TIME ZONE, TIMESTAMP WITH TIME ZONE, TIME WITHOUT TIME ZONE, and TIMESTAMP WITHOUT TIME ZONE.
    - The parameter length can be specified regardless of whether the GUC parameter **behavior_compat_options** is set to **proc_outparam_override**.
    - To pass the length of elements of the collection type and the length of elements of the array type nested by the collection type, you need to set the GUC parameter **behavior_compat_options** to **tableof_elem_constraints**.

- Other behavior:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The **OUT** parameter does not take effect in the following scenarios when it is used in an expression: The EXECUTE IMMEDIATE sqlv USING func syntax is used to execute a function. The SELECT func INTO syntax is used to execute a function. DML statements such as INSERT and UPDATE are used to execute a function. The SELECT WHERE a=func() method is used. A function with an OUT parameter is used as an input parameter (that is, fun(func(OUT b), a)) and the OUT parameter **b** does not take effect. A function with an OUT parameter is used to assign a default value to a variable.

Syntax

- PG-compatible syntax for creating a user-defined function:

```
CREATE [ OR REPLACE  ] FUNCTION function_name
   ( [  { argname [ argmode  ] argtype [  { DEFAULT  | :=  | =  } expression  ]}  [, ...]  ] )
   [ RETURNS rettype [ DETERMINISTIC  ]  | RETURNS TABLE (  { column_name column_type  }
[, ...] )]
   LANGUAGE lang_name
   [
     {IMMUTABLE  | STABLE  | VOLATILE }
      | {SHIPPABLE | NOT SHIPPABLE}
      | WINDOW
      | [ NOT  ] LEAKPROOF
      | {CALLED ON NULL INPUT  | RETURNS NULL ON NULL INPUT | STRICT }
      | {[ EXTERNAL  ] SECURITY INVOKER | [ EXTERNAL  ] SECURITY DEFINER | AUTHID DEFINER  |
AUTHID CURRENT_USER}
      | {fenced | not fenced}
      | {PACKAGE}
      | COST execution_cost
      | ROWS result_rows
      | SET configuration_parameter { {TO | =} value | FROM CURRENT }
   ][...]
   {
      AS 'definition'
      | AS 'obj_file', 'link_symbol'
   };
```

- Oracle-compatible syntax of creating a user-defined function:

```
CREATE [ OR REPLACE  ] FUNCTION function_name
   ( [  { argname [ argmode  ] argtype [  { DEFAULT | := | =  } expression  ] }  [, ...]  ] )
   RETURN rettype [ DETERMINISTIC  ]
   [
      {IMMUTABLE  | STABLE  | VOLATILE }
      | {SHIPPABLE | NOT SHIPPABLE}
      | {PACKAGE}
      | {FENCED | NOT FENCED}
      | [ NOT  ] LEAKPROOF
      | {CALLED ON NULL INPUT | RETURNS NULL ON NULL INPUT | STRICT }
      | {[ EXTERNAL  ] SECURITY INVOKER  | [ EXTERNAL  ] SECURITY DEFINER | AUTHID DEFINER |
AUTHID CURRENT_USER}
      | COST execution_cost
      | ROWS result_rows
      | SET configuration_parameter { {TO | =} value  | FROM CURRENT }
      | LANGUAGE lang_name
   ][...]
   {
     IS  | AS
   } plsql_body
/
```

Parameters

- **function_name**

  Name of the function to be created, which can be specified using a specific schema.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

NOTE

- If the value contains more than 63 characters, the database truncates it and retains the first 63 characters as the function name.
- When creating a function for operations on tables, you are advised to explicitly specify the schemas of tables in the function definition. Otherwise, the function may fail to be executed.
- You are advised not to create a function with the same name as a system function. Otherwise, you need to specify the schema of the function when calling the function.

- **argname**

  Specifies the parameter name of the function.

  Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

  NOTE

  If the value contains more than 63 characters, the database truncates it and retains the first 63 characters as the function parameter name.

- **argmode**

  Specifies the parameter mode of the function.

  Value range: **IN**, **OUT**, **INOUT**, and **VARIADIC**. The default value is **IN**.

  NOTE

  - **VARIADIC** specifies parameters of the array type.
  - **VARIADIC** can be added only to parameters in OUT mode, and parameters in OUT and INOUT modes cannot be used in the RETURNS TABLE function definition.

- **argtype**

  Specifies the data type of a function parameter. You can use **%ROWTYPE** to indirectly reference the type of a table, or **%TYPE** to indirectly reference the type of a column in a table or composite type.

  For details, see **Variable Definition Statements**.

- **expression**

  Specifies the default expression of a parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s2**, the default expression is not supported when the parameter is in INOUT mode.
- It is recommended that you define all default parameters after all non-default parameters.
- If a function with default parameters is called, input parameters are added to the function from left to right. If inputs of non-default parameters are missing, an error is reported.
- If **proc_uncheck_default_param** is enabled and a function with default parameters is called, input parameters are added to the function from left to right. The number of default inputs depends on the number of default parameters. If an input of a non-default parameter is missing, the previous default value is used to fill this parameter.
- When **a_format_version** is set to **10c**, **a_format_dev_version** is set to **s1**, **proc_outparam_override** is disabled, and the function parameters include **OUT** and **DEFAULT**, the default value cannot be used.

- **rettype**

  Specifies the return data type. Same as **argtype**, **%TYPE** or **%ROWTYPE** can also be used to indirectly reference types.

  When there is **OUT** or **INOUT** parameter, the RETURNS clause can be omitted. If the clause is not omitted, the result type of the clause must be the same as that of the output parameter. If there are multiple output parameters, the result type of the clause is RECORD. Otherwise, the result type of the clause is the same as that of a single output parameter. The SETOF modifier indicates that the function will return a set of items, rather than a single item.

- **column_name**

  Specifies the column name.

- **column_type**

  Specifies the column type.

  NOTE

  When the return type is RETURNS TABLE, each column is regarded as an **OUT** parameter. When the GUC parameter **behavior_compat_options** is set to **'proc_outparam_override'**, you must call functions using SELECT and CALL based on the actual parameters provided by the corresponding column. Otherwise, an error is reported, indicating that the function does not exist.

- **definition**

  Specifies a string constant defining a function. Its meaning depends on the language. It can be an internal function name, a path pointing to a target file, an SQL query, or text in a procedural language.

- **DETERMINISTIC**

  Specifies an API compatible with the SQL syntax, which is not supported in the current version.

- **LANGUAGE lang_name**

  Specifies the name of the language that the function is implemented in. It can be **SQL**, **C**, **internal**, or the name of a customized procedural language. To ensure compatibility, the name can be enclosed in single quotation marks (''). Contents in single quotation marks must be capitalized.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Due to compatibility issues, no matter which language is specified when an Oracle-compatible database is created, the language used is plpgsql.

NOTE

- When you define an internal function, if the AS clause specifies it as an internal system function, the parameter type, number of parameters, and return type of the new function must match those of the internal system function. Additionally, the user must have permission to execute the internal system function.
- Only users with the SYSADMIN permission can create internal functions.

- **WINDOW**

  Specifies that the function is a window function. This is currently only useful for functions written in C. The **WINDOW** attribute cannot be changed when replacing an existing function definition.

NOTICE

For a customized window function, the value of **LANGUAGE** can only be **internal**, and the referenced internal function must be a window function.

- **IMMUTABLE**

  Specifies that the function always returns the same result if the parameter values are the same.

- **STABLE**

  Specifies that the function cannot modify the database. Specifies that within a single table scan, it will consistently return the same result for the same parameter value, but its result varies by SQL statements.

- **VOLATILE**

  Specifies that the function value can change in a single table scan and no optimization is performed.

- **SHIPPABLE | NOT SHIPPABLE**

  Specifies whether the function can be pushed down to DNs for execution.

  – Functions of the IMMUTABLE type can always be pushed down to DNs.

  – Functions of the STABLE or VOLATILE type can be pushed down to DNs only if their attribute is **SHIPPABLE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- If IMMUTABLE, STABLE, or VOLATILE is not specified, VOLATILE is used by default.
- If SHIPPABLE or NOT SHIPPABLE is not specified, NOT SHIPPABLE is used by default.
- The function or stored procedure of the SHIPPABLE or IMMUTABLE type cannot contain EXCEPTION or call functions or stored procedures that contain EXCEPTION.
- If you specify the IMMUTABLE or SHIPPABLE type when creating a function, do not perform INSERT, UPDATE, DELETE, MERGE, or DDL operations in the function. Otherwise, the execution result may be incorrect. The CN determines the execution node for the preceding operations. If DDL statements are executed in the function of the IMMUTABLE or SHIPPABLE type, database objects on each node may be inconsistent. To resolve this problem, create the VOLATILE PL/SQL function on the CN, run the EXECUTE statement in the function definition to dynamically execute the DDL operation for repairing system objects, and then use the EXECUTE DIRECT ON syntax to call the repair function on the specified DN.
- If the result of a function depends on the scanning of data tables or a function has modified data or performed DDL operations on data tables, do not set the function attribute to **IMMUTABLE** or **SHIPPABLE**. Otherwise, database execution may be abnormal.

- **PACKAGE**

  Specifies whether the function can be overloaded.

  – All PACKAGE and non-PACKAGE functions cannot be overloaded or replaced.

  – PACKAGE functions do not support parameters of the VARIADIC type.

  – The **PACKAGE** attribute of functions cannot be modified.

- **LEAKPROOF**

  Specifies that the parameters of the function include only the return value. Only users with the SYSADMIN permission can set **LEAKPROOF**.

- **CALLED ON NULL INPUT**

  Declares that some parameters of the function can be called in normal mode if the parameter values are **NULL**. This parameter can be omitted.

- **RETURNS NULL ON NULL INPUT | STRICT**

  Specifies that a function always returns **NULL** whenever any of its parameters is **NULL**. If **STRICT** is specified, the function is not executed when there is a NULL value. Instead, a NULL result is returned automatically.

  RETURNS NULL ON NULL INPUT and **STRICT** have the same functions.

- **EXTERNAL**

  (Optional) The purpose is to be compatible with SQL statements. **EXTERNAL** applies to all functions, not only external functions.

- **SECURITY INVOKER | AUTHID CURRENT_USER**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies that the function will be executed with the permissions of the user who calls it. This parameter can be omitted.

SECURITY INVOKER and **AUTHID CURRENT_USER** have the same functions.

- **SECURITY DEFINER | AUTHID DEFINER**

  Specifies that the function will be executed with the permissions of the user who created it.

  AUTHID DEFINER and **SECURITY DEFINER** have the same functions.

- **FENCED | NOT FENCED**

  Specifies whether the function is executed in fenced mode or not fenced mode. In **NOT FENCED** mode, a function is executed in a CN or DN process. In **FENCED** mode, a function is executed in a new fork process, which does not affect CN or DN processes. You can choose **FENCED** or **NOT FENCED** by referring to the following:

  – Develop or debug a function in **FENCED** mode and execute it in **NOT FENCED** mode. This reduces the overhead of the fork process and communication.
  – Perform complex OS operations, such as opening a file, and processing signals and threads in **FENCED** mode; otherwise, the GaussDB database execution may be affected.
  – Customize PL/SQL functions. If this parameter is not specified, the default value **NOT FENCED** is used and the **FENCED** execution mode is not supported.

- **COST execution_cost**

  Estimates the execution cost of a function. The unit of **execution_cost** is **cpu_operator_cost**.

  Value range: a number greater than or equal to 0.

- **ROWS result_rows**

  Estimates the number of rows returned by the function. It is used only when the function returns a set.

  Value range: a number greater than or equal to 0. The default value is **1000**.

- **configuration_parameter**

  Set the parameters.

  – **value**

    Sets a specified database session parameter to a specified value.

    Value range: a string.

    - **DEFAULT**: The default system setting is used in the new session.
    - **OFF**: The setting is disabled.
    - **RESET**: The default system setting is used in the new session.
    - User-defined default value.

  – **FROM CURRENT**

    Uses the value of **configuration_parameter** of the current session.

- **plsql_body**

  Specifies the PL/SQL stored procedure body.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

When a stored procedure is created, the **PLSQL_BODY** can end with "END;" or "END function_name;". For details about the constraints for ending with "END function_name;", see **Note** in **CREATE PROCEDURE**.

NOTE

When you are performing operations in the function body related to passwords or keys, such as creating a user, changing a password, encrypting, or decrypting, the system catalogs and logs record plaintext information about the passwords or keys. To prevent sensitive information leakage, you are advised not to perform operations on the function body related to sensitive information, such as passwords or keys.

Examples

```
-- Create an ORA-compatible database.
gaussdb=# CREATE DATABASE ora_compatible_db DBCOMPATIBILITY 'ORA';
CREATE DATABASE

-- Switch to another database.
gaussdb=# \c ora_compatible_db

-- Define a function as SQL query.
ora_compatible_db=# CREATE FUNCTION func_add_sql(integer, integer) RETURNS integer
    AS 'select $1 + $2;'
    LANGUAGE SQL
    IMMUTABLE
    RETURNS NULL ON NULL INPUT;

-- Add an integer by parameter name using PL/pgSQL.
ora_compatible_db=# CREATE OR REPLACE FUNCTION func_increment_plsql(i integer) RETURNS integer AS
$$
        BEGIN
            RETURN i + 1;
        END;
    $$ LANGUAGE plpgsql;

-- Return the RECORD type.
ora_compatible_db=# CREATE OR REPLACE FUNCTION compute(i int, out result_1 bigint, out result_2
bigint)
    RETURNS SETOF RECORD
    AS $$
    BEGIN
        result_1 = i + 1;
        result_2 = i * 10;
    RETURN next;
    END;
    $$ LANGUAGE plpgsql;

-- Return a record containing multiple output parameters.
ora_compatible_db=# CREATE FUNCTION func_dup_sql(in int, out f1 int, out f2 text)
    AS $$ SELECT $1, CAST($1 AS text) || ' is text' $$
    LANGUAGE SQL;

-- Call the func_dup_sql function.
ora_compatible_db=# SELECT * FROM func_dup_sql(42);
 f1 |     f2
----+------------
 42 | 42 is text
(1 row)

-- Compute the sum of two integers and returning the result (if the input is null, the returned result is null):
ora_compatible_db=# CREATE FUNCTION func_add_sql2(num1 integer, num2 integer) RETURN integer
    AS
    BEGIN
        RETURN num1 + num2;
    END;
    /
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create an overloaded function with the PACKAGE attribute.
ora_compatible_db=# CREATE OR REPLACE FUNCTION package_func_overload(col int, col2  int)
    RETURN integer PACKAGE
    AS
    DECLARE
        col_type text;
    BEGIN
         col := 122;
         dbe_output.print_line('two int parameters ' || col2);
         RETURN 0;
    END;
    /

-- Create an overloaded function whose second formal parameter type is different from that of the
package_func_overload function.
ora_compatible_db=# CREATE OR REPLACE FUNCTION package_func_overload(col int, col2 smallint)
    RETURN integer PACKAGE
    AS
    DECLARE
        col_type text;
    BEGIN
         col := 122;
         dbe_output.print_line('two smallint parameters ' || col2);
         RETURN 0;
    END;
    /

-- Drop functions.
ora_compatible_db=# DROP FUNCTION func_add_sql;
ora_compatible_db=# DROP FUNCTION func_increment_plsql;
ora_compatible_db=# DROP FUNCTION compute;
ora_compatible_db=# DROP FUNCTION func_dup_sql;
ora_compatible_db=# DROP FUNCTION package_func_overload(col int, col2  int);
ora_compatible_db=# DROP FUNCTION package_func_overload(col int, col2 smallint);
ora_compatible_db=# DROP FUNCTION func_add_sql2;



-- If the parameter behavior_compat_options is not set to 'proc_outparam_override', the OUT and
INOUT parameters of the function directly called by an anonymous block or stored procedure cannot be of
the composite type, and the return value of the function directly called by an anonymous block or stored
procedure is used as the first value of the OUT parameter. As a result, the call fails.
ora_compatible_db=# CREATE TYPE rec AS(c1 int, c2 int);
ora_compatible_db=# CREATE OR REPLACE FUNCTION func(a in out rec, b in out int) RETURN int
    AS
    BEGIN
      a.c1:=100;
      a.c2:=200;
      b:=300;
      RETURN 1;
    END;
    /
ora_compatible_db=# DECLARE
        r rec;
        b int;
    BEGIN
  func(r,b);        -- Not supported
    END;
    /
ERROR:  A non-composite value cannot be assigned to a row variable.
CONTEXT:  PL/pgSQL function inline_code_block line 4 at SQL statement
ora_compatible_db=# DROP FUNCTION func;
ora_compatible_db=# DROP TYPE rec;

-- The following examples can be executed only in an ORA-compatible database:
ora_compatible_db=# CREATE OR REPLACE PACKAGE pkg_type AS
        type table_of_index_int is table of integer index by integer; -- Create an integer type.
        type table_of_index_int01 is table of table_of_index_int index by integer; -- Create a nested integer
type.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    type table_of_index_var is table of integer index by varchar(5); -- Create a varchar type.
    type table_of_index_var01 is table of table_of_index_var index by varchar(5); -- Create a nested
varchar type.
  END pkg_type;
  /

-- Create a function that returns results of the table of integer index by integer type.
ora_compatible_db=# CREATE OR REPLACE FUNCTION func_001(a in out pkg_type.table_of_index_int, b in
out pkg_type.table_of_index_var) --#add in & inout #defult value
  RETURN pkg_type.table_of_index_int
  AS
    table_of_index_int_val pkg_type.table_of_index_int;
    table_of_index_var_val pkg_type.table_of_index_var;
  BEGIN
    for i in 1..2 loop
      table_of_index_int_val(i) := i;
      a(i) := i;
      table_of_index_var_val(i) := i;
      b(i) := i;
    end loop;
    raise info '%',table_of_index_int_val;
    raise info '%',table_of_index_var_val;
    raise info '%',a;
    raise info '%',b;
  RETURN table_of_index_int_val;
  END;
  /
ora_compatible_db=# DECLARE
    table_of_index_int_val pkg_type.table_of_index_int;
    table_of_index_var_val pkg_type.table_of_index_var;
  BEGIN
    func_001(table_of_index_int_val,table_of_index_var_val);
  END;
  /

-- Create a function containing IN/OUT parameters.
ora_compatible_db=# CREATE OR REPLACE FUNCTION func_001(a in out date, b in out date) --#add in &
inout #defult value
  RETURN integer
  AS
  BEGIN
    raise info '%', a;
    raise info '%', b;
    RETURN 1;
  END;
  /
ora_compatible_db=# DECLARE
    date1 date := '2022-02-02';
    date2 date := '2022-02-02';
  BEGIN
    func_001(date1, date2);
  END;
  /

-- Create a function containing IN/OUT parameters.
ora_compatible_db=# CREATE OR REPLACE FUNCTION func_001(a in out INT, b in out date) --#add in &
inout #default value
  RETURN INT
  AS
  BEGIN
    raise info '%', a;
    raise info '%', b;
  RETURN a;
  END;
  /
ora_compatible_db=# DECLARE
    date1 int := 1;
    date2 date := '2022-02-02';
  BEGIN
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    func_001(date1, date2);
  END;
  /
INFO:  1
CONTEXT:  PL/pgSQL function inline_code_block line 5 at SQL statement
INFO:  2022-02-02 00:00:00
CONTEXT:  PL/pgSQL function inline_code_block line 5 at SQL statement
ANONYMOUS BLOCK EXECUTE

-- Drop functions.
ora_compatible_db=# DROP FUNCTION func_001;

-- Drop the package.
ora_compatible_db=# DROP PACKAGE pkg_type;

-- Switch back to the initial database and drop the test database. Replace postgres with the actual
database name.
ora_compatible_db=# \c postgees
gaussdb=# DROP DATABASE ora_compatible_db;
```

Helpful Links

ALTER FUNCTION and **DROP FUNCTION**

1.13.9.23 CREATE GLOBAL INDEX

Description

Creates a global secondary index (GSI) on a specified table.

The GSIs allow users to define indexes that are inconsistent with the distribution of base tables. In this way, single-node plans for querying non-distributed keys of base tables and unique/primary key constraints on non-distributed keys of base tables are achieved.

NOTE

You are advised to create GSIs on a base table as ordinary indexes to improve DML operation execution efficiency. In addition, after the GSIs are created on the base table, the DML operation performance is affected and deteriorates.

Limitations

- Only the GTM-lite mode is supported.
- GSIs can contain expressions but have the following constraints:
  - Similar to the base table limitations, distribution keys cannot contain expressions. In addition, GSIs must not contain expressions only because the distribution keys would be expressions definitely in this case.
  - Similar to ordinary index constraints, CONTAINING columns cannot contain expressions.
  - If any column name is prefixed with "expr", no GSI containing an expression can be created.
- Creating a GSI:
  - A GSI whose distribution is the same as that of the base table cannot be created.
  - The base table cannot be a replication table or list/range distributed table. GSIs support only hash distribution.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - If the base table is a partitioned non-hash bucket table, the GSI allows a maximum of 27 columns. If the base table is a non-partitioned hash bucket table, the GSI allows a maximum of 27 columns. If the base table is a partitioned hash bucket table, the GSI allows a maximum of 26 columns. If the base table is a non-partitioned non-hash bucket table, the GSI allows a maximum of 28 columns.

- Online GSI creation/rebuild:

  CREATE GSI CONCURRENTLY cannot be used for expression indexes or partial indexes.

Precautions

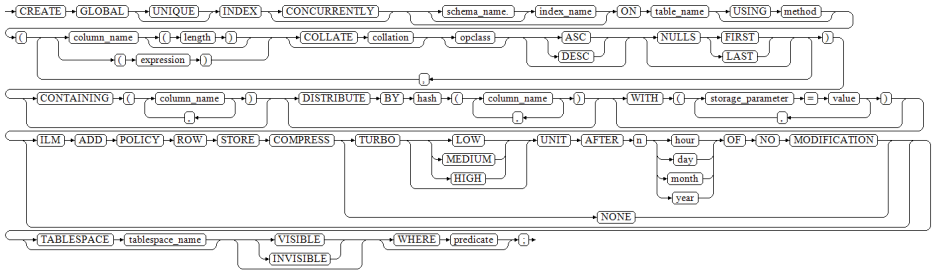
- For GSI creation, GSI rebuild, and operations related to GSI rebuild, for example, partitioning operations (such as DROP, TRUNCATE, MERGE, SPLIT, and EXCHANGE PARTITION) on a partitioned table, UPDATE DISTRIBUTED GLOBAL INDEX, ALTER TABLE involving data rebuild, as well as ALTER SET TABLESPACE and MOVE PARTITION operations on a hash bucket table, you are advised to enable the STREAM mode to achieve optimal performance. To enable the STREAM mode, set **enable_stream_operator** to **on** and **create_gsi_opt** to **build**.
- If the VACUUM FULL, CLUSTER, or REINDEX operation on a table is interrupted, the GSI on the table may turn to the **UNUSABLE** state. In this case, an error will be reported when the GSI is queried. You are advised to run **REINDEX INDEX** to rebuild the GSI.
- When executing **COPY** or **GDS** to import data to a base table with a GSI, the **enable_stream_operator** parameter must be enabled to achieve optimal data import performance.
- TABLE ACCESS BY INDEX ROWID supports STREAM and non-STREAM distributed plans. The STREAM plans utilizing TABLE ACCESS BY INDEX ROWID inherit STREAM-related limitations. Considering the STREAM communication latency, when the selectivity is too low or the number of predicate hit rows is small, the performance is not optimal. In this case, you are advised not to use a STREAM plan utilizing TABLE ACCESS BY INDEX ROWID. For a non-STREAM plan utilizing TABLE ACCESS BY INDEX ROWID, GSI table hints are required. This plan is applicable to scenarios where the number of predicate hit rows is small. If the performance of a STREAM plan utilizing TABLE ACCESS BY INDEX ROWID is not optimal, you can use a non-STREAM plan utilizing TABLE ACCESS BY INDEX ROWID. For details, see "Optimization Cases > Case: Using Global Secondary Indexes to Accelerate Query" in *SQL Optimization Guide*.
- In the batch INSERT INTO SELECT, UPDATE, DELETE, and UPSERT scenarios, you are advised to enable the **enable_stream_operator** parameter to execute the STREAM plan. (If the base table is a hash bucket table or tamper-proof table, the CN plan is still used.) If **enable_stream_operator** is disabled, the execution plan goes back to the CN, and the performance is poor.
- For the DELETE, UPDATE, and UPSERT statements executed on a base table with GSIs, if the STREAM plan is used, the restrictions are the same as those of UPDATE. The same row cannot be concurrently updated.
- GSI supports UPSERT. The restrictions are the same as those of common UPSERT. For details, see **Parameters**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The following operations will invalidate GSIs: database-level REINDEX, database-level/partition-level CLUSTER, and ALTER TABLE PARTITION (DROP, TRUNCATE, MERGE, SPLIT, and EXCHANGE PARTITION). If **UPDATE DISTRIBUTED GLOBAL INDEX** is not specified, all GSIs on both partitioned and ordinary tables will be invalidated.
- GSIs cannot be created for a base table when any column name, such as **ctid**, **xc_node_hash**, **xmin**, **xmax**, **tableoid** (for a partitioned table), or **tablebucketid** (for a hash bucket table) of the base table is duplicate with an existing column name after it is suffixed with "_new$$" or "_NEW$$."
- Concurrent GSI rebuild using REINDEX INDEX is not supported, which may block services.
- GSI-based table access by index row ID is a cross-DN operation. When the optimizer preferentially generates a pushdown plan, it cannot generate a plan that uses the GSI for table access by index row ID.

Syntax

```
CREATE GLOBAL [ UNIQUE ] INDEX [ CONCURRENTLY ] [ [schema_name.]index_name ] ON table_name
[ USING method ]
    ({ { column_name [ ( length ) ] | ( expression ) } [ COLLATE collation ] [ opclass ] [ ASC | DESC ]
[ NULLS { FIRST | LAST } ] }[, ...] )
    [CONTAINING (column_name [ ,... ])] [DISTRIBUTE BY hash(column_name [ ,... ])][ WITH
( {storage_parameter = value} [, ... ] )] [ ILM ADD POLICY ROW STORE COMPRESS { TURBO [ LOW |
MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION| NONE}]
    [ TABLESPACE tablespace_name ] [ VISIBLE | INVISIBLE ] [ WHERE predicate];
```



Parameters

- **UNIQUE**

  Creates a unique index. This causes the system to check for duplicate values in the table each time data is added. If the inserted or updated value causes duplicate records, an error is reported.

- **CONCURRENTLY**

  Creates an index (with ShareUpdateExclusiveLock) in a mode that does not block DML statements. When an index is created, other statements cannot access the table on which the index depends. If this keyword is specified, DML statements are not blocked during the creation.

  – The CREATE GLOBAL INDEX statement can be run within a transaction, but CREATE GLOBAL INDEX CONCURRENTLY cannot.

  – An index may fail to be created online. The possible causes are as follows: A user issues an instruction for terminating the online creation command. The definition of the index is modified. The definition of the base table is modified and the definition of the index is affected. For a non-critical error, an error message is displayed when online index building fails and

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

the system attempts to automatically clear the index. For a critical error, such as FATAL, PANIC, and database fault, and a clearing failure, an unusable index is left. This index is ignored by queries, but it still consumes the IUD overhead. Therefore, you are advised to use the DROP INDEX IF EXISTS syntax to drop the index and create the index online, or use the REINDEX/REBUILD syntax to rebuild the index.

- Do not use CREATE GLOBAL INDEX CONCURRENTLY together with other DDL operations. Otherwise, deadlocks or errors may occur. In this case, reconnect to the client and try again.
- For other precautions, see **CONCURRENTLY** in **CREATE INDEX**.

- **schema_name**

  Specifies a schema name.

  Value range: existing schema names.

- **index_name**

  Specifies the name of the index to be created. No schema name can be included here; the index is always created in the same schema as its parent table.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **table_name**

  Specifies the name (optionally schema-qualified) of the table to be indexed.

  Value range: existing table names.

- **USING method**

  Specifies the method used to create the index.

  Value range: UB-tree. The multi-version B-tree index is provided. The index page contains transaction information.

- **column_name**

  Specifies the name of the column on which the index is to be created.

  If the index mode supports composite indexes, multiple columns can be declared. A maximum of 28 columns can be declared for a non-partitioned base table, and a maximum of 27 columns can be declared for a partitioned base table.

- **column_name (length)**

  Prefix keys are supported. For details, see **column_name (length)** in **CREATE INDEX**.

  As an expression index, the prefix index has the same constraints as the expression key in GSI.

- **expression**

  Specifies an expression based on one or more columns of the table. The expression usually must be written with surrounding parentheses, as shown in the syntax. However, the parentheses can be omitted if the expression has the form of a function call. Expression indexes can be used to obtain fast access to data based on some transformation of the basic data. For example, an index computed on upper(col) would allow the clause WHERE upper(col) = 'JIM' to use an index. If the expression includes the IS NULL clause, the index will be invalid. In this case, you are advised to create a partial index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **COLLATE collation**

  Specifies a collation for the column to be indexed (which must be of a collatable data type). If not specified, the default collation is used. You can run the **SELECT * FROM pg_collation;** command to query collations from the pg_collation system catalog. The row starting with **default** in the query result represents the default collation.

- **opclass**

  Specifies the name of an operator class. An operator class can be specified for each column of an index. The operator class identifies the operators to be used by the index for that column.

- **ASC**

  Sorts data in ascending order (default).

- **DESC**

  Sorts data in descending order.

- **NULLS FIRST**

  Specifies that null values appear before non-null values in the sort ordering. This is the default when **DESC** is specified.

- **NULLS LAST**

  Specifies that null values appear after non-null values in the sort ordering. This is the default when **DESC** is not specified.

- **CONTAINING**

  Specifies the base table attributes other than index keys contained in the GSI.

- **CONTAINING (column_name [, ...])**

  Specifies the base table attributes other than index keys contained in the GSI.

- **DISTRIBUTE BY**

  Specifies the distribution key of the GSI, which is different from that of the base table, and only the hash distribution can be used.

- **DISTRIBUTE BY hash(column_name [,...])**

  Specifies the distribution key of the GSI.

  It needs to be included in the index key.

- **WITH ({storage_parameter = value} [, ...])**

  Specifies the storage parameters of the GSI.

  Value range:

  - **FILLFACTOR**

    Specifies a fillfactor for the index. It is a percentage between 10 and 100.

    Value range: 10–100.

  - **STORAGE_TYPE**

    Specifies the GSI storage type.

    Value range: **USTORE**

  - **INDEXSPLIT**

    Controls the method of selecting a proper split point for UB-tree, which is the same as that for B-tree.

    Value range: **DEFAULT** or **INSERTPT**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **enable_tde**

  Specifies the index as an encryption index. The database automatically encrypts the data in the encrypted index before storing it. Before using this parameter, ensure that the TDE function has been enabled using the GUC parameter **enable_tde**, the information for accessing the key service has been set using the GUC parameter **tde_key_info**, and the **enable_tde** attribute has been set for the base table. For detailed usage, consult "Feature Usage Guidelines > Transparent Data Encryption" in *Developer Guide*. This parameter supports only B-tree and UB-tree indexes. Other indexes such as hash indexes are not supported.

  Value range: **on** and **off**. If **enable_tde** is set to **on**, the values of **key_type**, **tde_cmk_id**, and **dek_cipher** are automatically generated in the database.

  Default value: **off**

- **encrypt_algo**

  Specifies the algorithm for encrypting indexes. This parameter must be used together with **enable_tde**.

  Value range: a string. Valid values are **AES_128_CTR**, **SM4_CTR**, **AES_128_CTR_SHA256_HMAC**, **SM4_CTR_SM3_HMAC**, and **AES_128_GCM**.

  Default value: If **enable_tde** is set to **on** and the subparameter **table_algorithm** in the GUC parameter **tde_encrypt_config** is not empty, the default value is the same as that of **table_algorithm**. If **table_algorithm** is empty, the default value is **AES_128_CTR**. If **enable_tde** is set to **off**, the default value is null.

- **dek_cipher**

  Specifies the DEK ciphertext. This parameter cannot be manually set or modified. After you set the **enable_tde** parameter for the index, the index automatically copies **dek_cipher** from the base table.

  Value range: a string.

  Default value: null

- **key_type**

  Specifies the type of the master key. This parameter cannot be manually set or modified. After you set the **enable_tde** parameter for the index, the index automatically copies **key_type** from the base table.

  Default value: null

- **cmk_id**

  Specifies the ID of the master key. This parameter cannot be manually set or modified. After you set the **enable_tde** parameter for the index, the index automatically copies **cmk_id** from the base table.

  Value range: a string.

  Default value: null

- opt_with_pkey

  Specifies that the created GSI is a primary key optimization index. This parameter can be specified only during creation and cannot be set or modified.

  Value range: **on** and **off**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **off**

NOTE

- A primary key can be created on GSIs using the **opt_with_pkey** option (referred to as primary key optimization GSI). When a table is updated, if the updated columns do not involve the columns of the primary key optimization GSI, the primary key optimization GSI is not updated, thereby improving the update performance.
- When creating or rebuilding a primary key optimization GSI, ensure that a non-GSI primary key index in the USABLE state exists in the table.
- If the cluster upgrade is not committed, the primary key optimization GSI cannot be created.
- Expression index GSIs and partial index GSIs with primary key optimization cannot be created.
- If a primary key optimization GSI exists in the target table, the primary key cannot be directly deleted. It can only be deleted in cascading mode (CASCADE).
- If the primary key optimization GSI exists in the target table, the primary key index is invalidated (UNUSABLE), and all primary key optimization GSIs are invalidated.
- The primary key optimization status (**opt_with_pkey**) can only be specified during creation and cannot be changed.

– new_partbucket_gsi

Specifies whether the new data structure is used for the GSI created on the partitioned hash bucket table. This parameter is automatically specified when the GSI is created.

Value range: **on**.

Default value: **on**.

NOTE

- During the upgrade observation period of the first upgrade to a version that supports the **new_partbucket_gsi** option, the **new_partbucket_gsi** option cannot be specified.
- If you attempt to specify the **new_partbucket_gsi** option on a GSI that is not on a partitioned hash bucket table or a non-GSI, an error will be reported.
- REINDEX does not affect the value of the **new_partbucket_gsi** option.
- The value of the **new_partbucket_gsi** option cannot be changed using the ALTER syntax.

- **[ILM ADD POLICY ROW STORE COMPRESS { TURBO [LOW | MEDIUM | HIGH] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}]**

  – When creating a GSI, you can call ILM ADD POLICY to add a cross-page transparent compression policy. For example:

  ```
  CREATE GLOBAL INDEX gsi_idx on t1(a) ILM ADD POLICY ROW STORE COMPRESS TURBO HIGH UNIT AFTER 3 HOUR OF NO MODIFICATION;
  ```

  The command creates the gsi_idx index and adds a cross-page transparent compression policy with a high policy level and a 3-hour cold and hot data management interval.

  Where, AFTER n { hour | day | month | year } OF NO MODIFICATION indicates the compression groups that are not modified in *n* hours, days, months, or years.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - When creating an index, you can call ILM ADD POLICY ROW STORE COMPRESS NONE to explicitly specify that no cross-page transparent compression policy is applied to the index.

- **TABLESPACE tablespace_name**

  Specifies a tablespace for the index. If not specified, the default tablespace is used.

  Value range: existing tablespace names.

- **VISIBLE**

  Sets the index to be visible. This is the default option.

  NOTE

  - VISIBLE can be set only in an ORA-compatible database (that is, **sql_compatibility** is set to **'ORA'**).
  - When **disable_keyword_options** is set to **"visible"**, this keyword cannot be used.
  - This keyword is not supported in the upgrade uncommitted phase.

- **INVISIBLE**

  Sets the index to be invisible.

  NOTE

  - INVISIBLE can be set only in an ORA-compatible database (that is, **sql_compatibility** is set to **'ORA'**).
  - When **disable_keyword_options** is set to **"invisible"**, this keyword cannot be used.
  - This keyword is not supported in the upgrade uncommitted phase.

- **WHERE predicate**

  Creates a partial index. A partial index contains entries for only a portion of a table, usually a portion that is more useful for indexing than the rest of the table. For example, if you have a table that contains both billed and unbilled orders where the unbilled orders take up a small portion of the total table and yet that is an often used portion, you can improve performance by creating an index on just that portion. In addition, the WHERE clause with a UNIQUE constraint can be used to enforce uniqueness over a subset of a table.

  Value range: The predicate expression can only refer to columns of the underlying table, but it can use all columns, not just the ones being indexed. Currently, subqueries and aggregate expressions are forbidden in the WHERE clause. You are advised not to use a predicate of numeric types such as int because such types can be implicitly converted to Boolean values (non-zero values are implicitly converted to **true** and **0** is implicitly converted to **false**), which may cause unexpected results.

Examples

```
-- Create a base table test1 that contains three columns.
gaussdb=# CREATE TABLE test1(c1 int, c2 int, c3 int);

-- Create a GSI on the c2 column of the test table containing the c3 column based on the hash distribution of the c2 column.
gaussdb=# CREATE GLOBAL INDEX idx_gsi_1 ON test1(c2) CONTAINING(c3) DISTRIBUTE BY HASH(c2);

-- Create a base table test2 that contains three columns.
gaussdb=# CREATE TABLE test2(c1 int, c2 int, c3 int);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a GSI on the c2 column of the test2 table containing the c3 column based on the hash distribution of the c2 column.
gaussdb=# CREATE GLOBAL INDEX idx_gsi_2 ON test2(c2) CONTAINING(c3);

-- Create a base table test3 that contains three columns.
gaussdb=# CREATE TABLE test3(c1 int, c2 int, c3 int);

-- Create a GSI in UNIQUE form on the c2 column of the test3 table, which is based on the hash distribution of the c2 column by default.
gaussdb=# CREATE GLOBAL UNIQUE INDEX idx_gsi_3 ON test3(c2) DISTRIBUTE BY HASH(c2);

-- Drop the index.
gaussdb=# DROP INDEX idx_gsi_1;
gaussdb=# DROP INDEX idx_gsi_2;
gaussdb=# DROP INDEX idx_gsi_3;

-- Drop the table.
gaussdb=# DROP TABLE test1;
gaussdb=# DROP TABLE test2;
gaussdb=# DROP TABLE test3;
```

Helpful Links

ALTER INDEX, **CREATE INDEX**, and **DROP INDEX**

1.13.9.24 CREATE GROUP

Description

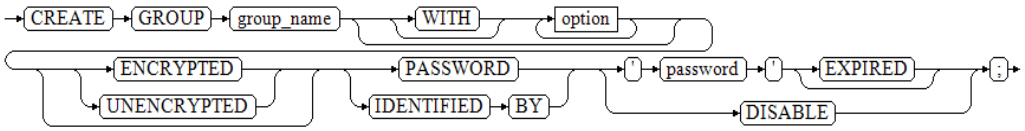
Creates a user group.

Precautions

CREATE GROUP is an alias for **CREATE ROLE** and is not a standard SQL syntax; its use is therefore discouraged. You are advised to use **CREATE ROLE** directly.

Syntax

```
CREATE GROUP group_name [ [ WITH ] option [ ... ] ]
    [ ENCRYPTED | UNENCRYPTED ] { PASSWORD | IDENTIFIED BY } { 'password' [ EXPIRED ] | DISABLE };
```



The syntax of the option clause (optional) is as follows:

```
{{SYSADMIN | NOSYSADMIN}
    | {MONADMIN | NOMONADMIN}
    | {OPRADMIN | NOOPRADMIN}
    | {AUDITADMIN | NOAUDITADMIN}
    | {CREATEDB | NOCREATEDB}
    | {USEFT | NOUSEFT}
    | {CREATEROLE | NOCREATEROLE}
    | {INHERIT | NOINHERIT}
    | {LOGIN | NOLOGIN}
    | {REPLICATION | NOREPLICATION}
    | {PERSISTENCE | NOPERSISTENCE}
    | CONNECTION LIMIT connlimit
    | VALID BEGIN 'timestamp'
    | VALID UNTIL 'timestamp'
    | RESOURCE POOL 'respool'
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  | USER GROUP 'groupuser'
  | PERM SPACE 'spacelimit'
  | TEMP SPACE 'tmpspacelimit'
  | SPILL SPACE 'spillspacelimit'
  | NODE GROUP logic_group_name
  | IN ROLE role_name [, ...]
  | IN GROUP role_name [, ...]
  | ROLE role_name [, ...]
  | ADMIN role_name [, ...]
  | USER role_name [, ...]
  | SYSID uid
  | DEFAULT TABLESPACE tablespace_name
  | PROFILE DEFAULT
  | PROFILE profile_name
  | PGUSER}
```

Parameters

For details, see **CREATE ROLE** in **Parameters**.

Examples

```
-- Create a user group. The effect is the same as that of CREATE ROLE.
gaussdb=# CREATE GROUP test_group WITH PASSWORD "********";

-- Run CREATE ROLE to create a role. You cannot log in to the database by default.
-- Run ALTER ROLE role_name WITH LOGIN to enable users to log in to the database.
gaussdb=# CREATE ROLE test_role WITH PASSWORD "********";

-- Run CREATE USER to create a user. A schema with the same name is automatically created and the user has the login permission.
gaussdb=# CREATE USER test_user WITH PASSWORD "********";

-- View the user information.
gaussdb=# \du test*
            List of roles
 Role name  |  Attributes  | Member of
------------+--------------+-----------
 test_group | Cannot login | {}
 test_role  | Cannot login | {}
 test_user  |              | {}

-- Query the schema automatically created by the CREATE USER command.
gaussdb=# \dn test*
    List of schemas
   Name    |   Owner
-----------+-----------
 test_user | test_user
(1 row)

-- Drop.
gaussdb=# DROP ROLE test_role;
gaussdb=# DROP GROUP test_group;
gaussdb=# DROP USER test_user;
```

Helpful Links

ALTER GROUP, **DROP GROUP**, and **CREATE ROLE**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.9.25 CREATE INCREMENTAL MATERIALIZED VIEW

Description

Creates a fast-refresh materialized view. You can refresh the data of a materialized view using REFRESH MATERIALIZED VIEW (complete-refresh) or REFRESH INCREMENTAL MATERIALIZED VIEW (fast-refresh).

The CREATE INCREMENTAL MATERIALIZED VIEW statement has similar functions as the CREATE TABLE AS statement. However, the CREATE INCREMENTAL MATERIALIZED VIEW statement persistently stores the initialization query statement to support subsequent data refresh operations. Materialized views have the same persistence property as ordinary tables. However, unlike temporary tables, the system does not support the creation of temporary materialized views.

NOTE

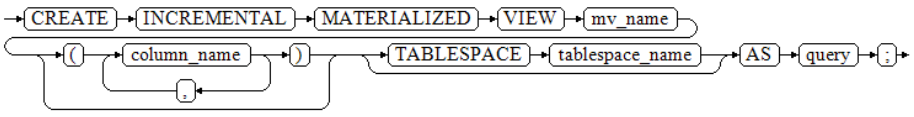
For details about the application scenarios, see "Basic Concepts > Materialized Views" in *Developer Guide*.

Precautions

- Fast-refresh materialized views cannot be created on database link tables, temporary tables, or global temporary tables.
- Fast-refresh materialized views support only simple filter queries and UNION ALL queries of base tables.
- Fast-refresh materialized views do not support clauses WITH, GROUP BY, ORDER BY, LIMIT, and WINDOW. Operators DISTINCT and AGG are not supported. Subqueries except UNION ALL are not supported.
- The distribution key cannot be specified when a fast-refresh materialized view is created.
- After a fast-refresh materialized view is created, most DDL operations in the base table are no longer supported.
- The IUD operation cannot be performed on fast-refresh materialized views.
- After a fast-refresh materialized view is created, you need to run the **REFRESH** command to synchronize the materialized view with the base table when the base table data changes.
- The Ustore does not support the creation and use of materialized views.

Syntax

```
CREATE INCREMENTAL MATERIALIZED VIEW mv_name
   [ (column_name [, ...] ) ]
   [ TABLESPACE tablespace_name ]
   AS query;
```



Parameters

- **mv_name**

Name of the created materialized view, which can be specified using a schema.

Value range: character strings complying with **Identifier Naming Conventions**.

- **column_name**

  Name of a column in the new materialized view. The materialized view supports specified columns. The number of specified columns must be the same as that of columns in the result of the subsequent query statement. If no column name is specified, the column name is obtained from the output column name of the query.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **TABLESPACE tablespace_name**

  (Optional) Tablespace to which the new materialized view belongs. If it is not specified, the default tablespace is used.

- **AS query**

  SELECT or **TABLE** command. This query will be run in a security-constrained operation.

Examples

```
-- Create a tablespace.
gaussdb=# CREATE TABLESPACE tbs_data1 RELATIVE LOCATION 'tablespace1/tbs_data1';

-- Create an ordinary table.
gaussdb=# CREATE TABLE my_table (c1 int, c2 int);

-- Create a fast-refresh materialized view.
gaussdb=# CREATE INCREMENTAL MATERIALIZED VIEW my_imv (col1,col2) TABLESPACE tbs_data1 AS
SELECT * FROM my_table;

-- Write data to the base table.
gaussdb=# INSERT INTO my_table VALUES(1,1);

-- Query fast-refresh materialized view data.
gaussdb=# SELECT * FROM my_imv;
 col1 | col2
------+------
(0 rows)

-- Fast refresh the fast-refresh materialized view my_imv.
gaussdb=# REFRESH INCREMENTAL MATERIALIZED VIEW my_imv;

-- Query fast-refresh materialized view data.
gaussdb=# SELECT * FROM my_imv;
 col1 | col2
------+------
    1 |    1
(1 row)

-- Delete a fast-refresh materialized view.
gaussdb=# DROP MATERIALIZED VIEW my_imv;

-- Delete the ordinary table my_table.
gaussdb=# DROP TABLE my_table;

-- Delete the tablespace.
gaussdb=# DROP TABLESPACE tbs_data1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

ALTER MATERIALIZED VIEW, **CREATE MATERIALIZED VIEW**, **CREATE TABLE**, **DROP MATERIALIZED VIEW**, **REFRESH INCREMENTAL MATERIALIZED VIEW**, and **REFRESH MATERIALIZED VIEW**

1.13.9.26 CREATE INDEX

Description

Defines a new index.

Indexes are primarily used to enhance database performance (though inappropriate use can result in slower performance). You are advised to create indexes on:

- Columns that are often queried.
- Join conditions. For a query on joined columns, you are advised to create a composite index on the columns. For example, for **SELECT * FROM t1 JOIN t2 ON t1.a=t2.a AND t1.b=t2.b**, you can create a composite index on columns **a** and **b** in the **t1** table.
- Columns having filter criteria (especially scope criteria) of a WHERE clause.
- Columns that are frequently used after **ORDER BY**, **GROUP BY**, and **DISTINCT**
- The system determines whether to create an index on a partitioned table based on the following rules:
  - If the GLOBAL or LOCAL keyword is specified during index creation, the index of the corresponding type will be created.
  - If a partition name is specified during index creation, a local index will be created.
  - If a UNIQUE index contains all partition keys, a local index is created. Otherwise, a global index is created.
  - By default, a global index is created.

Precautions

- If the base table is a hash distributed table, to create a primary key or unique index that does not contain the distribution key of the base table, use a GSI (using CREATE GLOBAL INDEX); to create a primary key or unique index that contains the distribution key of the base table, use an ordinary index (using CREATE INDEX). In single-DN deployment mode, both the GSI and ordinary index can be successfully created. If the base table is a non-hash distributed table, you can only create the primary key or unique index as an ordinary index (using CREATE INDEX). That is, the index key must contain the distribution key of the base table.
- Indexes consume storage and computing resources. Creating excessive indexes may negatively impact database performance—particularly during data import. It is advisable to import data before creating indexes
- All functions and operators used in an index definition must be "immutable", that is, their results must depend only on their input parameters and never on any outside influence (such as the contents of another table or the current

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

time). To use a user-defined function in an index or WHERE clause, remember to mark the function immutable when you create it.

- Indexes on partitioned tables are classified into local or global indexes. A local index is specific to a partition within a partitioned table, while a global index spans the entire partitioned table.
- In the same set of columns, you cannot create both a local index and a global index.
- Users with the CREATE ANY INDEX permission can create indexes in the public and user schemas.
- If the base table is distributed in hash, range, or list mode, the distribution key of the base table must be included when a unique index is created, and no expression is allowed.
- If a user-defined function is called in an expression index, the expression index function is executed based on the permission of the table owner.
- All users can create expression indexes on their own tables. Non-initial user cannot create expression indexes on the tables of the initial user. The initial user can create expression indexes on all tables. When separation of duties is disabled, a system administrator can create expression indexes on non-initial user tables. When separation of duties is enabled, a system administrator can create expression indexes on tables of common users, but cannot create expression indexes on tables of the initial user and the AUDITADMIN and CREATEROLE roles.
- Data of the XML type cannot be used as ordinary indexes, unique indexes, global indexes, local indexes, or partial indexes.
- Only B-tree and UB-tree indexes can be created online. Only ordinary indexes of non-partitioned tables, as well as global and local indexes of partitioned tables can be created. PCR UB-tree indexes, subpartitions, and GSIs are not supported. Only ordinary indexes, global indexes, and local indexes of Astore and Ustore can be created online in parallel.
- Currently, UGINs support only the Ustore and single-column indexes. For details about other scenarios, see **Description for the USING ugin syntax**.
- GIN and GiST indexes are only supported by Astore. For details about other scenarios, see the syntax description.
- Creating an index using CREATE INDEX may change the table access mode. As a result, the query execution plan changes.
- Due to structural constraints, the B-tree index requires at least three index tuples per page, resulting in a maximum index length limitation. The maximum length of a single index tuple for B-tree and UB-tree indexes is 2696 bytes. However, for the UB-tree index of Ustore, the actual maximum length of a single index tuple is 2688 bytes due to the inclusion of transaction information. Creating an index for a field exceeding the maximum length triggers an error, and migrating from the Astore table to the Ustore table may cause the corresponding index creation to fail.
- Indexes containing **rowid** and **rowno** system columns cannot be created.
- If the SQL statement for creating an index is too long (greater than or equal to 5250 characters), a warning is reported. You are advised to shorten the SQL statement for creating an index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> If the SQL statement for creating an index is too long, it may cause failures during index backup and restoration. The statement for creating an index during backup and restoration is exported by gs_dump. When gs_dump exports the index definition, it adds quotation marks, parentheses, and column types to the function name, condition, and column name. Therefore, the original SQL statement becomes longer. As a result, the pg_index tuple corresponding to the statement cannot be stored on a page, and the index fails to be created.

- When creating a composite index, you need to create it based on query conditions and the leftmost matching principle of the composite index.

  > **NOTE**
  >
  > - Leftmost matching principle of a composite index: If a query condition contains one or more columns of a composite index, the leftmost consecutive columns of the composite index must match the query condition.
  > - When the query condition is **WHERE a = ?, b = ?,c = ?, d =?** or **WHERE a = ?, b = ?,c = ?**, the optimal index matching is used. When the query condition is WHERE **b = ?,c = ?, d =?** or **WHERE c = ?, d = ?**, the index **idx_test_abcd** may be executed after cost calculation. However, in such cases, all pages of the index are scanned, resulting in unsatisfactory SQL performance. In similar cases, you are advised to create a composite index suitable for the query condition based on the leftmost matching principle.
  >
  > ```
  > -- Create the test table.
  > gaussdb=# CREATE TABLE test(a int, b int, c int, d int, e int, f text);
  > Create a composite index.
  > gaussdb=# CREATE INDEX idx_test_abcd ON test(a,b,c,d);
  > ```

Syntax

- Create an index on a table.

```
CREATE [ UNIQUE ] INDEX [ CONCURRENTLY ] [ IF NOT EXISTS ] [ [schema_name.] index_name ]
ON table_name [ USING method ]
    ({ { column_name [ ( length ) ] | ( expression ) } [ COLLATE collation ] [ opclass ] [ ASC | DESC ]
[ NULLS { FIRST | LAST } ] }[, ...] )
    [ INCLUDE ( column_name [, ...] ) ]
    [ WITH ( {storage_parameter = value} [, ... ] ) ]
    [ ILM ADD POLICY ROW STORE COMPRESS
 { TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO
MODIFICATION
 | NONE}]
    [ TABLESPACE tablespace_name ]
    [ COMMENT 'string' ] [ ... ]
    [ VISIBLE | INVISIBLE ]
    [ WHERE predicate ];
```

- Create an index on a partitioned table.

```
CREATE [ UNIQUE ] INDEX [ CONCURRENTLY ] [ IF NOT EXISTS ] [ [schema_name.] index_name ]
ON table_name [ USING method ]
    ( { { column_name [ ( length ) ] | ( expression ) } [ COLLATE collation ] [ opclass ] [ ASC | DESC ]
[ NULLS LAST ] } [, ...] )
    [ LOCAL
        [ ( PARTITION index_partition_name [ FOR { partition_name | ( partition_value [, ...] ) } ]
[ TABLESPACE index_partition_tablespace ]
              [ ( [SUBPARTITION index_subpartition_name] [ FOR { partition_name | ( partition_value
[, ...] ) } ]  [ TABLESPACE index_partition_tablespace ]
                    [, ...] ) ]
           [, ...] ) ]
       | GLOBAL ]
    [ INCLUDE ( column_name [, ...] ) ]
    [ WITH ( { storage_parameter = value } [, ...] ) ]
    [ ILM ADD POLICY ROW STORE COMPRESS
 { TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO
MODIFICATION
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
| NONE}]
  [ TABLESPACE tablespace_name ]
  [ COMMENT 'string' ] [ ... ]
  [ VISIBLE | INVISIBLE ];
```

Parameters

- **UNIQUE**

  Creates a unique index. This causes the system to check for duplicate values in the table each time data is added. Attempts to insert or update data which would result in duplicate entries will generate an error.

  Currently, only B-tree indexes and UB-tree indexes in row-store tables support unique indexes.

- **CONCURRENTLY**

  Creates an index (with ShareUpdateExclusiveLock) in a mode that does not block DML statements. When an index is created, other statements cannot access the table on which the index depends. If this keyword is specified, DML statements are not blocked during the creation.

  – This option can only specify a name of one index.

  – The CREATE INDEX statement can be run within a transaction. However, CREATE INDEX CONCURRENTLY cannot be run within a transaction or in a stored procedure.

  – For temporary tables, you can use CONCURRENTLY to create indexes. However, indexes are created in blocking mode because no other sessions concurrently access the temporary tables and the blocking mode is more cost-effective.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This keyword is specified when an index is created. The Astore needs to scan the entire table twice to build it. During the first scan, an index is created and the read and write operations are not blocked. During the second scan, changes that have occurred since the first scan are merged and updated. Ustore needs to scan the entire table only once. During the scan, the data generated by concurrent DML operations is inserted into the temporary table **index_oid_cctmp**. After the scan is complete, the data in the temporary table is merged to the primary index and the temporary table is deleted, the index is created. You can use the GUC parameter **delete_cctmp_table** to specify whether to delete temporary tables after indexes are created online. The default value of this parameter is **on**, indicating that temporary tables are deleted by default. If this parameter is set to **off**, temporary tables are retained.
- For Astore, the table needs to be scanned and built twice, and all existing transactions that may modify the table must be completed. This means that the creation of the index takes a longer time than normal. In addition, the CPU and I/O consumption also affects other services. Although the Ustore only needs to scan the entire table once to create indexes, the preceding consumption also exists.
- Exceptions may occur during online index creation (for example, manual cancellation, duplicate UNIQUE index key values, insufficient resources, thread startup failure, and lock timeout). As a result, online index creation fails. In this case, indexes in the **not valid** state may be retained to avoid occupying system resources. The system automatically clears the failed indexes. Before clearing the failed indexes, wait until the transaction in the table ends. If a long transaction exists, the system may keep waiting. If you cancel the thread, the failed indexes remain. Specifically, if the user cancels the online index creation thread and the three phases of online index creation are complete, the process of clearing failed indexes will not be started and the failed indexes will remain. If the three phases of online index creation are not complete, the process of clearing failed indexes will be started first. If the user cancels the thread again, the process will end and the failed indexes will remain. The automatic index clearing process includes a phase where the system must wait for other database transactions to end. As a result, issues such as lock timeout and deadlock may occur, causing automatic index clearing failures. For critical errors, such as FATAL, PANIC, and database faults, you need to manually clear indexes and temporary tables. If both online index creation and automatic residual clearing fail, indexes that are not valid are retained. The residual indexes may be ready or not ready (depending on the phase in which the online index creation fails. For example, the index failed in the first phase is not ready, and the index failed in the third phase is ready). If the residual indexes are ready, the DML statements still maintain the residual indexes. During the maintenance, other errors may occur (for example, the size of the index column exceeds the maximum value or the UNIQUE index constraint is violated). To prevent residual indexes from occupying system resources and generating unexpected errors, you need to manually delete them as soon as possible. In a distributed system, only DNs automatically clear failed indexes. (If an exception occurs on a DN and the three phases of online index creation have been completed, the failed indexes will not be automatically cleared.) Failed indexes on the CN need to be manually cleared. If a critical error occurs, the metadata of the failed indexes cannot be matched on some nodes in the distributed system. The DROP INDEX statement cannot be used to drop these indexes. You need to use the DROP INDEX IF EXISTS statement to drop them.
- For Astore, after the second scan, index creation must wait for any transaction that holds a snapshot earlier than the snapshot taken by the second scan to end. In addition, the ShareUpdateExclusiveLock (level 4) added during index creation conflicts with a lock whose level is greater than or equal to 4. Therefore, when such an index is created, the system is prone to hang or deadlock. Example:
  - If two sessions create an index by using CONCURRENTLY for the same table, a deadlock occurs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

    - If a session creates an index by using CONCURRENTLY for a table and the other session drops a table, a deadlock occurs.
    - There are three sessions. Session 1 locks table **a** and does not commit it. Session 2 creates an index by using CONCURRENTLY for table **b**. Session 3 writes data to table **a**. Before the transaction of session 1 is committed, session 2 is blocked.
    - When an index is created by using CONCURRENTLY for a table concurrently with the TRUNCATE operation on the same table, a deadlock occurs.
    - The transaction isolation level is set to repeatable read (read committed by default). Two sessions are started. Session 1 writes data to table **a** and does not commit it. Session 2 creates an index by using CONCURRENTLY for table **b**. Until the transaction of session 1 is committed, session 2 is always blocked.
- When an index is being created or fails to be created, you need to check the index progress or status. You can call the gs_get_index_status('schema_name', 'index_name') function to check the index status on all nodes. The input parameters **schema_name** and **index_name** are used to specify the index schema name and index name, respectively. The return values are **node_name**, **indisready**, and **indisvalid**, indicating the node name, whether the index can be inserted on the node, and whether the index is available on the node. The index is available only when **indisready** and **indisvalid** on all nodes are set to **true**; otherwise, wait until the index creation is complete. If the index fails to be created, drop the index and create it again.
- When the I/O, CPU, and network bandwidth resources are not limited, the service performance deterioration caused by online index creation can be controlled within 10%. However, in special scenarios, the service performance deterioration may exceed 10%. This is because online index creation is a long transaction that consumes a large number of I/O, CPU, and network bandwidth resources. It consumes more resources than offline index creation. The longer the online index creation transaction lasts, the greater the impact on service performance. The time for creating indexes online is positively correlated with the data volume of base tables and the data volume generated by concurrent DML statements. When the I/O, CPU, and network bandwidth resources are not limited, the time for creating indexes online is about two to six times that for creating indexes offline. However, when the number of concurrent transactions is large (> 10000 TPS) or resource contention occurs, the time may be even longer. If a long transaction exists during online index creation, the long transaction running time must be added. In Astore and Ustore modes, you can create indexes in parallel to shorten the index creation time. The performance of online parallel index creation increases to a certain value and becomes stable as the number of parallel worker threads increases. Compared with the performance of creating indexes in serial mode, the performance of creating indexes in parallel online is improved by about 30%. You are advised to create indexes online during off-peak hours to avoid great impact on services. In addition, do not execute long transactions during online index creation. Although online index creation provides the capability of uninterrupted services to some extent, it still needs to be implemented with caution.
- When a unique index is created online (by using CREATE UNIQUE INDEX CONCURRENTLY), the table is scanned to check whether the data in the table is unique. If the data in the table is not unique, an error is reported and the online index creation exits. During online index creation, whether data in the table is unique may change. Whether the creation fails depends on whether duplicate data is scanned. Consider the following situations: 1. Tuple A and tuple B are inserted during table scan, and their index columns are the same. If both tuple A and tuple B are scanned during online index scan, a uniqueness violation may be reported. However, if tuple B is deleted later, in this case, an error may be reported during online index creation, indicating that the uniqueness is violated. However, after the online index creation fails and exits, the data in the table is found to be unique. 2. During online index creation, if tuple A in a table is inserted, deleted, and then inserted, and SnapshotNow is used to scan the table, tuple A may be scanned twice, which may violate the UNIQUE constraint. During online index creation,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Astore uses SnapshotMVCC to scan tables, and Ustore uses SnapshotNow to scan tables.

- **IF NOT EXISTS**
  - When **IF NOT EXISTS** is specified, the system checks whether a relation with the same name already exists in the current schema before creating an index. If it finds such a relation, the table creation process will be halted, and a NOTICE will be returned.
  - When IF NOT EXISTS is not specified and a relation with the same name is present in the schema, an ERROR will be returned.
- **schema_name**

  Specifies the schema name.

  Value range: names of existing schemas.
- **index_name**

  Specifies the name of the index to be created. No schema name can be included here; the index is always created in the same schema as its parent table.

  Value range: character strings complying with **Identifier Naming Conventions**.
- **table_name**

  Specifies the name (optionally schema-qualified) of the table to be indexed.

  Value range: names of existing tables.
- **USING method**

  Specifies the name of the index method to be used.

  Value range:
  - **btree**: B-tree indexes store key values of data in a B+ tree structure. This structure helps users to quickly search for indexes. B-tree is applicable to comparison query and range query.
  - **ubtree**: multi-version B-tree index used only for Ustore tables. The index page contains transaction information and can be recycled. By default, the INSERTPT functionality is enabled for UB-tree indexes.
  - **ugin**: UGINs are generalized inverted indexes used only for Ustore tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- Currently, UGIN supports opclass of the character type ugin_trgm_ops.
  - **ugin_trgm_ops** is not the default **opclass**.
  - **ugin_trgm_ops** supports index keys of the text, varchar, nvarchar2, and clob data types and alias data types of the four types, but does not support the char(n) type (bpchar).
  - If the index key of ugin_trgm_ops is specified, the collation cannot be M-compatible, such as utf8mb4_bin, utf8mb4_general_ci, utf8mb4_unicode_ci, gbk_chinese_ci, gbk_bin, gb18030_chinese_ci and gb18030_bin. Otherwise, an error is reported during index creation.
- Currently, UGIN indexes support the opclass ugin_tsvector_ops of the text search type tsvector.
  - ugin_tsvector_ops is the default operator class.
  - ugin_tsvector_ops supports index keys of the tsvector data type and supports the @@/@@@ operator for index scans.
- Currently, UGIN support the following opclasses of the array type: _bool_ops, _bpchar_ops, _char_ops, _float4_ops, _float8_ops, _int2_ops, _int4_ops, _int8_ops, _interval_ops, _numeric_ops, _oid_ops, _text_ops, _time_ops, _timetz_ops, _timestamp_ops, _timestamptz_ops and _varchar_ops.
  - The preceding opclasses of the array type are default operator classes.
  - UGIN can be created for the following array types: bool[], char[], "char"[], float4[], float8[], int2[], int4[], int8[], interval[], numeric[], oid[], text[], time[], timestamp[], timestamptz[], timetz[] and varchar[].
  - The "&&", "@>", "<@", and "=" operators are supported.
- Only single-column indexes are supported. Index keys do not support the ASC/DESC syntax, NULLS FIRST/LAST syntax, or INCLUDE syntax.
- Only Bitmap Index Scan is supported.
- The GUC parameter **gin_fuzzy_search_limit** can only be set to the default value **0**. If the value is not 0, an error is reported during index scan for UGINs.
- Hash bucket tables, hash bucket scale-out, and segment-page are not supported.
- Only ordinary table indexes and local indexes are supported. Global indexes are not supported. It cannot be used as an index for unlogged tables or temporary tables.
- Online operations such as rebuild, creation, deletion, and VACUUM FULL are not supported, nor are parallel creation.
- UGINs cannot be used for flashback queries.
- It cannot be used as primary keys, unique keys, or virtual indexes, and does not support transparent data encryption or fully-encrypted databases.
- UGINs can be used only for LIKE fuzzy query.
- It cannot be created or used in the database where **sql_compatibility** is set to **'M'**.
- UGINs support strings of all character sets. Ensure that the character set settings (**server_encoding**, **LC_CTYPE**, and **LC_COLLATE**) of the database align with the data definition. In the word segmentation algorithm, the concept of a character is determined by the character set, and the letter is determined by **LC_CTYPE**. Mismatched database character set settings can cause abnormal word segmentation.

– **gin**: General inverted indexes, which can process values that contain multiple keys (for example, arrays).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- GINs support the following data types and index operators:
  - Array types: abstime[], bit[], bool[], bpchar[], bytea[], char[], cidr[], date[], float4[], float8[], inet[], int2[], int4[], int8[], interval[], macaddr[], money[], name[], numeric[], oid[], oidvector[], reltime[], text[], time[], timestamp[], timestamptz[], timetz[], tinterval[], varbit[], and varchar[].

    Indexable operators: =, &&, @>, and <@.
  - Text search type: tsvector.

    Indexable operators: @@ and @@@.
  - JSON/JSONB type: jsonb.

    For the JSONB type, two indexable operator classes (jsonb_ops and jsonb_hash_ops) are provided. The default indexable operator class jsonb_ops supports the @>, ?, ?&, and ?| operators, while the non-default indexable operator class jsonb_hash_ops supports only the @> operator.
- Index keys do not support the ASC/DESC syntax, NULLS FIRST/LAST syntax, or INCLUDE syntax.
- Only Bitmap Index Scan is supported.
- Indexes of ordinary tables and local temporary tables, and local indexes of partitioned tables are supported. Global indexes, and indexes of global temporary tables and unlogged tables are not supported.
- Hash bucket tables and hash bucket scale-out are not supported.
- Online rebuild, creation, deletion, VACUUM FULL, and parallel creation are not supported.
- It cannot be used as a primary key, unique key, virtual index, or for transparent or full encryption.
- Index compression is not supported.
- The array, jsonb, and tsvector data types do not support collecting statistical information, making the cost estimation of GINs on these types inaccurate.

– **gist**: Suitable for the set data type and multidimensional data types, such as geometric and geographic data types. The following data types are supported: box, point, poly, circle, tsvector, tsquery, and range.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- GiST indexes support the following data types and index operators:
  - Box: Indexable operators include <<, &<, &&, &>, >>, ~=, @>, <@, &<|, <<|, |>>, |&>, ~, and @.
  - Circle: Indexable operators include <<, &<, &&, &>, >>, ~=, @>, <@, &<|, <<|, |>>, and |&>.
  - Point: Indexable operators include <<, >>, ~=, <@, <^, >^, and <->.
  - Polygon: Indexable operators include <<, &<, &>, >>, ~=, <<|, &<|, |&>, and |>>.
  - Range types include int4range, int8range, numrange, tsrange, tstzrange, daterange, and custom range types. Indexable operators include =, &&, @>, <@, <<, >>, @<, &>, and -|-.
  - Text search type tsquery: Indexable operators include <@ and @>.
  - Text search type tsvector: Indexable operator is @@.
- GiST-supported data types do not support collecting statistical information, making the cost estimation of GiST indexes created on these types is inaccurate.
- Only Astore is supported. Ustore, hash bucket, transparent data encryption, full-disk encryption, materialized views, parallel index creation, online rebuild, creation, deletion, and VACUUM FULL are not supported.
- Bitmap Index Scan and Index Scan are supported for executing planned scans.
- Indexes on ordinary tables/local temporary tables and partitioned indexes of partitioned tables are supported. Indexes on global temporary tables or unlogged tables are not supported.
- The ASC/DESC syntax, NULLS FIRST/LAST syntax, and INCLUDE syntax are not supported.
- Cannot be used as a primary key, unique key, or virtual index.
- gs_check_table and gs_check_index do not support GiST indexes.
- In ORA-compatible mode, **daterange** does not support **@>(daterange, date)**.
- Index compression is not supported.

NOTE

- B-tree and UB-tree are closely related to the table storage type Astore or Ustore. When an index is created, if the specified index type does not correspond to the main table, the index type is automatically converted.
- Row-store tables (Astore) support the index type B-tree (default), GIN, and GiST. Row-store tables (Ustore) support the index types UB-tree and UGIN.
- Recommendations for using GINs and UGINs:
  - Modifying indexes in high-concurrency environments is not advised, as synchronizing and converging incremental data can cause performance fluctuations.
  - Frequent index modifications are not recommended because space recycling and statistics updates depend on the VACUUM operation. Frequent insertions and deletions between VACUUM operations may cause space bloat and inaccurate execution plans.
  - If the scan performance is poor, manually clear the incremental cache, as its scan speed is lower than that of a B-tree scan.
  - Indexes are applicable only to scenarios where a small amount of data is returned from large datasets but are not suitable for retrieving large amounts of data.

- **column_name**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the name of the column on which an index is to be created.

Multiple columns can be specified if the index method supports multi-column indexes. A global index supports a maximum of 31 columns, and other indexes support a maximum of 32 columns.

- **column_name (length)**

  Creates a prefix key index based on a column in the table. **column_name** indicates the column name of the prefix key, and **length** indicates the prefix length.

  The prefix key uses the prefix of the specified column data as the index key value, which reduces the storage space occupied by the index. Indexes can be used for partial filter and join conditions that contain prefix key columns.

  NOTE

  - The index methods supported by the prefix key include B-tree and UB-tree.
  - The data type of the prefix key column must be binary or character (excluding special characters).
  - The prefix length must be a positive integer that does not exceed 2676 and cannot exceed the maximum length of the column. For the binary type, the prefix length is measured in bytes. For non-binary character types, the prefix length is measured in characters. The actual length of the prefix key value is restricted by the internal page. If a column contains multi-byte characters or an index has multiple keys, the length of the index row may exceed the upper limit. As a result, an error is reported. Consider this situation when setting a long prefix length.
  - In the CREATE INDEX syntax, the following keywords cannot be used as prefix keys for column names: COALESCE, EXTRACT, GREATEST, LEAST, LNNVL, NULLIF, NVL, NVL2, OVERLAY, POSITION, REGEXP_LIKE, SUBSTRING, TIMESTAMPDIFF, TREAT, TRIM, XMLCONCAT, XMLELEMENT, XMLEXISTS, XMLFOREST, XMLPARSE, XMLPI, XMLROOT and XMLSERIALIZE.
  - Prefix keys represent a special type of expression key. Certain constraints and limitations not explicitly outlined here align with those governing expression keys. For comprehensive information, refer to the documentation on expression indexes.
  - The prefix index is a kind of expression index and cannot be used as the unique index or primary key.

- **expression**

  Specifies an expression index based on one or more columns of the table. Generally, the expression index must be written in parentheses. However, the parentheses can be omitted if the expression is in the form of a function call.

  The expression index can be used to obtain fast access to certain transformation of basic data. For example, an index computed on upper(col) would allow the WHERE upper(col) = 'JIM' clause to use an index.

  If an expression contains the IS NULL clause, the index for this expression is invalid. In this case, you are advised to create a partial index.

- **COLLATE collation**

  Specifies a collation for the column to be indexed (which must be of a collatable data type). If no collation is specified, the default collation is used. You can run the **SELECT * FROM pg_collation;** command to query collations from the pg_collation system catalog. The default collation rule is the row starting with **default** in the query result.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> In MYSQL-compatible mode (that is, **sql_compatibility** is set to **'MYSQL'**), when the MYSQL-compatible collation is used, ensure that the index collation is the same as the column collation. Otherwise, the index fails to be created and an error is reported.

- **opclass**

  Specifies the name of an operator class. You can specify an operator class for each column of an index to identify the operator used by that column. For example, a B-tree index on the type int4 would use the int4_ops class; this operator class includes comparison functions for values of type int4. In practice, the default operator class for the column's data type is sufficient. The operator class applies to data with multiple sorts. For example, users might want to sort a complex-number data type either by absolute value or by real part. They could do this by defining two operator classes for the data type and then selecting the proper class when making an index. In addition, if the value of **COLLATE** of an index of the string type (such as varchar, varchar2, and text) is not **C** or **POSIX**, but you want the index to support prefix matching, you need to specify the **varchar_pattern_ops** option.

- **ASC**

  Specifies an ascending (default) sort order.

- **DESC**

  Specifies a descending sort order.

- **NULLS FIRST**

  Specifies that null values appear before non-null values in the sort ordering. This is the default when **DESC** is specified.

- **NULLS LAST**

  Specifies that null values appear after non-null values in the sort ordering. This is the default when **DESC** is not specified.

- **LOCAL**

  Specifies that the partitioned index to be created is a local index.

- **GLOBAL**

  Specifies the partitioned index to be created as a global index. If no keyword is specified, a global index is created by default.

- **INCLUDE (column_name [, ...]**)

  The optional INCLUDE clause specifies that some non-key columns are included in indexes. Non-key columns cannot be used as search criteria for accelerating index scans, and they are omitted when the UNIQUE constraints of the indexes are checked.

  An index-only scan can directly return content in the non-key columns without accessing the heap table corresponding to the indexes.

  Exercise caution when adding non-key columns as INCLUDE columns, especially for wide columns. If the size of an index tuple exceeds the maximum size allowed by the index type, data insertion fails. Note that in any case, adding non-key columns to an index increases the space occupied by the index, which may slow down the search speed.

  Currently, only UB-tree index access mode supports this feature. Non-key columns are stored in the index leaf tuple corresponding to the heap tuple and are not included in the tuple on the upper-layer index page.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **PARTITION clause**

  PARTITION index_partition_name [FOR { partition_name | (partition_value [, ...]) }] [TABLESPACE index_partition_tablespace]

  If **FOR partition_name** or **FOR partition_value** is specified in the PARTITION clause, the created partitioned index has the sparsely partitioned index attribute. If **FOR partition_name** or **FOR partition_value** is not specified, the number of PARTITION clauses must be the same as the number of partitions in the target table.

  NOTE

  - A sparsely partitioned index is a type of local index for partitioned table. It is created only on a single partition, whereas an ordinary local index needs to be created on every partition of the table.
  - Sparsely partitioned indexes can be created only on a single partition, including a partition of a partitioned table or a partition and a subpartition of a subpartitioned table.
  - The sparsely partitioned index supports only B-tree and UB-tree indexes.
  - The sparsely partitioned index does not support the UNIQUE attribute.
  - Currently, the global index is not supported.
  - A created sparsely partitioned index contains the **sparsely_partitioned=true** option. A sparsely partitioned index created on the partition of a subpartitioned table contains the **level1_sparsely_partitioned=true** option. The **sparsely_partitioned=true** and **level1_sparsely_partitioned=true** options cannot be explicitly specified as sparsely partitioned indexes using the CREATE TABLE or ALTER TABLE statement. If the created index is a sparsely partitioned index, the **sparsely_partitioned=false** or **level1_sparsely_partitioned=false** option will be ignored even if it is specified.
  - If a version earlier than 505.0.0 is upgraded to 505.0.0 or later, sparsely partitioned indexes cannot be created during upgrade observation.
  - When a version is upgraded to 506.1.0 or later, sparsely partitioned indexes cannot be created on partitions of subpartitioned tables during the upgrade observation period.

  – **PARTITION index_partition_name**

    Specifies the name of an index partition.

    Value range: character strings complying with **Identifier Naming Conventions**.

  – **FOR partition_name**

    Specifies the name of the target partition. If the specified partition does not exist, an error is reported.

  – **FOR (partition_value [, ...])**

    Specifies the value of the specified partition key. The partition key value must be placed in the partition key value FOR list. If there are multiple columns of partition keys, multiple **partition_value** values must be specified.

  – **TABLESPACE index_partition_tablespace**

    Specifies the tablespace of an index partition.

    Value range: If this parameter is not specified, the value of **index_tablespace** is used.

- **WITH ({storage_parameter = value} [, ...])**

  Specifies the storage parameter used for an index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

NOTE

- Indexes other than Psort, UGIN, and GIN support the **FILLFACTOR** parameter.
- Only UB-tree indexes support the **INDEXSPLIT** parameter.
- Only B-tree indexes on non-partitioned tables support the **DEDUPLICATION** parameter.
- Only UB-tree indexes support the **INDEX_TXNTYPE** parameter.
- UGINs and GINs also support the **gin_pending_list_limit**, **fastupdate**, **pending_pages**, **total_pages**, **entry_pages**, **data_pages**, and **entries** parameters.
- GiST indexes also support the **buffering** parameter.

– **STORAGE_TYPE**

Specifies the storage engine type of the table where the index is located. The storage engine type will be automatically corrected if it conflicts with the index type. Only B-Tree indexes, UB-Tree indexes, GINs, and UGINs are supported. You cannot modify this parameter after it is set.

Value range:

- **USTORE**: The table where the index is located uses an in-place update storage engine.
- **ASTORE**: The table where the index is located uses an append-only storage engine.

Default value: **USTORE** for Ustore tables and **ASTORE** for Astore tables.

– **FILLFACTOR**

Specifies a fill factor for the index. It is a percentage between 10 and 100. In the scenario where a large number of concurrent insertions are performed and the key value range is dense, the contention of the same index page is high during the insertion. Therefore, a smaller fill factor is more appropriate.

Value range: 10–100.

– FASTUPDATE

Specifies whether to enable fast update for a GIN/UGIN. Currently, the incremental cache (pending list) cannot be disabled for UGINs. Therefore, this parameter can only be set to **on**. If it is set to **off**, an error is reported.

Value range: **on** or **off**.

Default value: **on**

– **GIN_PENDING_LIST_LIMIT**

Sets the upper limit of the GIN/UGIN cache (pending list). If the pending list grows larger than this maximum size, it is cleaned up by moving the entries in it to the main index data structure in batches.

Value range: 64–2147483647. The unit is KB.

Default value: depends on the GUC parameter **gin_pending_list_limit** whose default value is **4**, in MB.

– **CROSSBUCKET**

Specifies whether cross-hash bucket indexes are used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: **on** or **off**.

Default value: **off**.

- **ACTIVE_PAGES**

  Specifies the number of index pages, which may be less than the actual number of physical file pages and can be used for optimization. Currently, this parameter is valid only for the local index of the Ustore partitioned table and will be updated by VACUUM (including AUTOVACUUM) and ANALYZE. You are advised not to manually set this parameter because it is invalid in the current mode.

- **DEDUPLICATION**

  Specifies whether to deduplicate and compress tuples with duplicate key values. This is an index parameter. When there are a large number of indexes with duplicate key values, enabling this parameter can effectively reduce the space occupied by indexes. This parameter does not take effect for primary key indexes and unique indexes. If non-unique indexes are used and the index key value duplication rate is low or the index key values are unique, enabling this parameter will slightly deteriorate the index insertion performance. Currently, local and global indexes of partitioned tables are not supported.

  Value range: Boolean value, depending on the value of the GUC parameter **enable_default_index_deduplication** (the default value is **off**).

- **INDEXSPLIT**

  Specifies the splitting policy of UB-tree indexes.

  Value range:

  - **INSERTPT**: The splitting policy is the same as that of B-tree.
  - **DEFAULT**: The index space usage can be significantly reduced in some scenarios.

  Default value: **INSERTPT**.

  Setting suggestion: In scenarios where a large amount of data with the same index key value is concurrently inserted at the same time, using the **INSERTPT** splitting algorithm may cause index bloat. In this case, you are advised to change the splitting policy to **DEFAULT**.

  For example:

  ```
  CREATE INDEX idx ON tbl(col) WITH (indexsplit=default);
  ```

- **INDEX_TXNTYPE**

  Specifies the UB-tree index type (only UB-tree indexes support this parameter). When the value is **PCR**, flashback queries can be performed using UB-tree. PCR UB-tree indexes do not support online index creation, GSIs, ultimate RTO replay, or read on standby. If this parameter is not specified, the type of the index to be created is specified by the GUC parameter **index_txntype**. **INDEX_TXNTYPE** cannot be modified using ALTER INDEX INDEX_NAME SET (INDEX_TXNTYPE=PCR or RCR).

  Type: character string (case-insensitive).

  Value range: **RCR** or **PCR**.

  Default value: **RCR**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

For example:

```
CREATE UNIQUE INDEX t2_b_pkey ON t(b) WITH(index_txntype='pcr');
```

- **STAT_STATE**

  Specifies whether index statistics are locked. If locked, the statistics cannot be updated. This parameter does not take effect in the current mode.

  Value range: **locked** or **unlock**.

  Default value: **unlock**.

- **enable_tde**

  Specifies whether the index is encrypted. If the index is encrypted, the database automatically encrypts the data in the encrypted index before storing it.

  Before using this parameter, ensure that the TDE functionality has been enabled using the GUC parameter **enable_tde**, the information for accessing the key service has been set using the GUC parameter **tde_key_info**, and the **enable_tde** attribute has been set for the base table. For details about how to use this parameter, see "Feature Usage Guidelines > Transparent Data Encryption" in *Developer Guide*. This parameter supports only B-tree and UB-tree indexes. Other indexes are not supported.

  Value range: **on** or **off**. If **enable_tde** is set to **on**, the values of **key_type**, **tde_cmk_id**, and **dek_cipher** are automatically generated in the database.

  Default value: **off**.

- **encrypt_algo**

  Specifies the cryptographic algorithm of the index. After you set the **enable_tde** parameter for the index, the index automatically copies **encrypt_algo** from the base table. This parameter cannot be manually set or modified.

  Value range: **AES_128_CTR**, **SM4_CTR**, **AES_128_CTR_SHA256_HMAC**, **SM4_CTR_SM3_HMAC**, and **AES_128_GCM** (character string).

  Default value: empty.

- **dek_cipher**

  Specifies the DEK ciphertext. This parameter cannot be manually set or modified. After you set the **enable_tde** parameter for an index, the index automatically copies the **dek_cipher** parameter in the base table.

  Value range: character strings.

  Default value: empty.

- **key_type**

  Specifies the type of the master key. This parameter cannot be manually set or modified. After you set the **enable_tde** parameter for the index, the index automatically copies **key_type** from the base table.

  Default value: empty.

- **cmk_id**

  Specifies the ID of the master key. This parameter cannot be manually set or modified. After you set the **enable_tde** parameter for the index, the index automatically copies **cmk_id** from the base table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: character strings.

Default value: empty.

- **LPI_PARALLEL_METHOD**

  Index parameter, which is used to set the parallel creation mode of local indexes for partitioned tables.

  Type: String

  Value range:

  - **PAGE**: creates page-level indexes in parallel. Multiple worker threads are started to scan and sort data. Each worker thread processes a data page at a time. After scanning and sorting, the main thread combines the sorting results in serial mode and inserts tuples into the indexes.
  - **PARTITION**: creates partition-level indexes in parallel. Multiple worker threads are started, and each worker thread scans, sorts, and inserts indexes into a partition.
  - **AUTO**: estimates the costs of parallel index creation at the page level and partition level based on the partitioned table statistics, and selects the parallel index creation mode with a lower cost. (The statistics may be different from the actual data, resulting in inaccurate calculation results.)

  Default value: **PAGE**

  Setting suggestion: Set this parameter to **PARTITION** if the partitioned table data is evenly distributed in each partition. This parameter supports only the B-tree local index of Astore partitioned tables. It does not support the global index of partitioned tables, non-partitioned table indexes, segment-page table indexes, hash bucket table indexes, or online index creation.

  For example:

  ```
  CREATE INDEX idx ON tbl(col) WITH (lpi_parallel_method = 'partition');
  ```

- **pending_pages**, **total_pages**, **entry_pages**, **data_pages**, and **entries**

  These parameters represent the number of UGIN and GIN fast cache pages, total index pages, word segmentation tree pages, posting tree pages, and total word segments.

  NOTE

  - The preceding parameters are page statistics unique to UGIN and GINs. They are updated by the **ANALYZE** command and cannot be manually set by using the **CREATE INDEX** or **ALTER INDEX** command.
  - **CREATE TABLE LIKE INCLUDING {index | all}** does not inherit the index statistics.
  - gs_dump does not export unique index information.
  - The pg_get_indexdef function does not obtain unique index information.
  - Running **\d+ table_name** does not display unique index information.
  - Transaction-level temporary tables do not support collection of page statistics for GINs.

- **buffering**

  Specifies whether GiST indexes use buffering technology to build indexes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **off**: Buffering is disabled.
- **on**: Buffering is enabled.
- **auto**: Buffering is initially disabled, but is enabled immediately when the index size reaches **effective_cache_size**.

The default value is **auto**. The default behavior is suitable for most cases, but disabling the buffering feature can accelerate the building process if the input data is already sorted.

– wait_clean_cbi

Specifies whether the CBI contains residual tuples generated during bucket migration for scaling. After scale-out and redistribution, **wait_clean_cbi** is set to **y** for the CBIs that contain residual tuples. After these indexes are cleared using VACUUM or REINDEX, **wait_clean_cbi** automatically changes to **n**. This parameter can only be used in the scale-out tool and cannot be set or modified.

Value range: **y** or **n**.

Default value: **n**.

- **[ILM ADD POLICY ROW STORE COMPRESS { TURBO [LOW | MEDIUM | HIGH] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}]**
  - When creating an index, you can call ILM ADD POLICY to add a cross-page transparent compression policy. For example:
    ```
    CREATE INDEX i1 on t1(a) ILM ADD POLICY ROW STORE COMPRESS TURBO HIGH UNIT AFTER 3 HOUR OF NO MODIFICATION;
    ```
    The command creates the i1 index and adds a cross-page transparent compression policy with a high policy level and a 3-hour cold and hot data management interval.
  - **AFTER n { hour | day | month | year } OF NO MODIFICATION**: indicates the compression groups that have not been modified for *n* hours, days, months, or years.
  - When creating an index, you can call ILM ADD POLICY ROW STORE COMPRESS NONE to explicitly specify that no cross-page transparent compression policy is applied to the index.
  - If an index compression policy is explicitly specified, the specified policy is used (the compression level is MEDIUM by default if no compression level is specified). If no index compression policy is explicitly specified, the index compression status and compression level are the same as those of the table.
- **TABLESPACE tablespace_name**

  Specifies the tablespace for an index. If no tablespace is specified, the default tablespace is used.

  Value range: names of existing tablespaces.
- **WHERE predicate**

  Creates a partial index.

  A partial index contains entries for only a portion of a table, usually a portion that is more useful for indexing than the rest of the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: The predicate expression can only refer to columns of the table, but it can use all these columns, not just the ones being indexed. Currently, subqueries and aggregate expressions are forbidden in the WHERE clause. You are advised not to use numeric types such as INT as conditions because such types can be implicitly converted to Boolean (non-zero values are implicitly converted to **true** and **0** is implicitly converted to **false**), which may cause unexpected results.

For a partitioned table index, if the created index contains the GLOBAL keyword or the created index is a GLOBAL index, the WHERE clause cannot be used to create an index.

- **COMMENT 'string'**

  Adds comments to an index.

  NOTE

  - It is valid only in a MYSQL-compatible database (that is, **sql_compatibility** is set to **'MYSQL'**).
  - An index-level comment can contain a maximum of 1024 characters.

- **VISIBLE**

  Sets the index to be visible. This is the default option.

  NOTE

  - **VISIBLE** can be set only in an ORA-compatible database (that is, **sql_compatibility** is set to **'ORA'**).
  - When **disable_keyword_options** is set to **visible**, this keyword cannot be used.
  - This keyword is not supported in the upgrade uncommitted phase.

- **INVISIBLE**

  Sets the index to be invisible.

  NOTE

  - **INVISIBLE** can be set only in an ORA-compatible database (that is, **sql_compatibility** is set to **'ORA'**).
  - When **disable_keyword_options** is set to **invisible**, this keyword cannot be used.
  - This keyword is not supported in the upgrade uncommitted phase.

Examples

- Ordinary index

```
-- Create the tbl_test1 table.
gaussdb=# CREATE TABLE tbl_test1(
    id int,           -- User ID
    name varchar(50), -- Username
    postcode char(6)  -- Postal code
);

-- Create the tbs_index1 tablespace.
gaussdb=# CREATE TABLESPACE tbs_index1 RELATIVE LOCATION 'test_tablespace/tbs_index1';

-- Create the idx_test1 index for the tbl_test1 table and specify a tablespace.
gaussdb=# CREATE INDEX idx_test1 ON tbl_test1(name) TABLESPACE tbs_index1;

-- Query information about the idx_test1 index.
gaussdb=# SELECT indexname,tablename,tablespace FROM pg_indexes WHERE indexname =
'idx_test1';
 indexname | tablename | tablespace
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-----------+-----------+------------
 idx_test1 | tbl_test1 | tbs_index1
(1 row)

-- Drop the index.
gaussdb=# DROP INDEX idx_test1;

-- Delete the tablespace.
gaussdb=# DROP TABLESPACE tbs_index1;
```

- Unique index

```
-- Create the unique index idx_test2 for the tbl_test1 table.
gaussdb=# CREATE UNIQUE INDEX idx_test2 ON tbl_test1(id);

-- Query index information.
gaussdb=# \d tbl_test1
          Table "public.tbl_test1"
  Column  |         Type          | Modifiers
----------+-----------------------+-----------
 id       | integer               |
 name     | character varying(50) |
 postcode | character(6)          |
Indexes:
    "idx_test2" UNIQUE, btree (id) TABLESPACE pg_default

-- Drop the index.
gaussdb=# DROP INDEX idx_test2;
```

- Expression index

```
-- Create an expression index for the tbl_test1 table.
gaussdb=# CREATE INDEX idx_test3 ON tbl_test1(substr(postcode,2));

-- Query index information.
gaussdb=# \d tbl_test1
          Table "public.tbl_test1"
  Column  |         Type          | Modifiers
----------+-----------------------+-----------
 id       | integer               |
 name     | character varying(50) |
 postcode | character(7)          |
Indexes:
    "idx_test3" btree (substr(postcode::text, 2)) TABLESPACE pg_default

-- Drop the index.
gaussdb=# DROP INDEX idx_test3;
```

- Partial index

```
-- Create an index for data whose ID is not empty in the tbl_test1 table.
gaussdb=# CREATE INDEX idx_test4 ON tbl_test1(id) WHERE id IS NOT NULL;

-- Drop the index.
gaussdb=# DROP INDEX idx_test4;

-- Drop the table.
gaussdb=# DROP TABLE tbl_test1;
```

- Partitioned index

```
-- Create a table.
gaussdb=# CREATE TABLE student(id int, name varchar(20)) PARTITION BY RANGE (id) (
    PARTITION p1 VALUES LESS THAN (200),
    PARTITION pmax VALUES LESS THAN (MAXVALUE)
);

-- Create a local partitioned index without specifying the index partition name.
gaussdb=# CREATE INDEX idx_student1 ON student(id) LOCAL;

-- Check the index partition information. It is found that the number of local index partitions is the
same as the number of table partitions.
gaussdb=# SELECT relname FROM pg_partition WHERE parentid = 'idx_student1'::regclass;
   relname
-------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 p1_id_idx
 pmax_id_idx
(2 rows)

-- Drop the local partitioned index.
gaussdb=# DROP INDEX idx_student1;

-- Create a global index.
gaussdb=# CREATE INDEX idx_student2 ON student(id) GLOBAL;

-- Check the index partition information. It is found that the number of global index partitions is
different from the number of table partitions.
gaussdb=# SELECT relname FROM pg_partition WHERE parentid = 'idx_student2'::regclass;
 relname
---------
(0 rows)

-- Drop the global index.
gaussdb=# DROP INDEX idx_student2;

-- Create a local expression index without specifying the index partition name.
gaussdb=# CREATE INDEX idx_student3 ON student(abs(id)) LOCAL;

-- Check the index partition information. It is found that the number of local index partitions is the
same as the number of table partitions.
gaussdb=# SELECT relname FROM pg_partition WHERE parentid = 'idx_student3'::regclass;
   relname
-------------
 p1_abs_idx
 pmax_abs_idx
(2 rows)

-- Drop the expression index of the local partition.
gaussdb=# DROP INDEX idx_student3;

-- Create a global expression index.
gaussdb=# CREATE INDEX idx_student4 ON student(abs(id)) GLOBAL;

-- Check the index partition information. It is found that the number of global expression index
partitions is different from the number of table partitions.
gaussdb=# SELECT relname FROM pg_partition WHERE parentid = 'idx_student4'::regclass;
 relname
---------
(0 rows)

-- Drop the global expression index.
gaussdb=# DROP INDEX idx_student4;

-- Drop the table.
gaussdb=# DROP TABLE student;
```

- UGIN

```
-- Create a table.
gaussdb=# CREATE TABLE test_trgm(t varchar(100)) WITH (STORAGE_TYPE=USTORE);

-- Insert data.
gaussdb=# INSERT INTO test_trgm VALUES('qwertyu0001');
gaussdb=# INSERT INTO test_trgm VALUES('qwertyu0099');
gaussdb=# INSERT INTO test_trgm VALUES ('2024 Happy New Year 7 days');

-- Create a UGIN.
gaussdb=# CREATE INDEX trgm_idx ON test_trgm USING UGIN (t ugin_trgm_ops);

-- Query using the LIKE statement.
gaussdb=#  SELECT * FROM test_trgm WHERE t LIKE '%24 Happy%'

-- Drop the table.
gaussdb=# DROP TABLE test_trgm CASCADE;

-- Create a UGIN for an array type.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a table.
gaussdb=# CREATE TABLE test_array(a int, b bool[]) WITH (STORAGE_TYPE=USTORE);

-- Insert data.
gaussdb=# INSERT INTO test_array VALUES(1, ARRAY[true, false]);

-- Create a UGIN.
gaussdb=# CREATE INDEX ugin_idx_array_bool ON test_array USING UGIN (b _bool_ops);

-- Query data.
gaussdb=# SELECT b FROM test_array WHERE b = ARRAY[true, false];
   b
-------
 {t,f}
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE test_array CASCADE;
```

- GIN

```
-- Create a table.
gaussdb=# CREATE TABLE test_gin (id INT, info INT[]) WITH(STORAGE_TYPE = ASTORE);

-- Create an index.
gaussdb=# CREATE INDEX test_gin_idx ON test_gin USING GIN(info);

-- Insert data.
gaussdb=# INSERT INTO test_gin SELECT g, ARRAY[1, g % 5, g] FROM generate_series(1, 20000) g;

-- Query statement.
gaussdb=# SELECT * FROM test_gin WHERE info @> '{2}' AND info @> '{22}' ORDER BY id, info;

-- Drop the table.
gaussdb=# DROP TABLE test_gin CASCADE;
```

- GiST

```
-- Create a table.
gaussdb=# CREATE TABLE gist_box_tbl(f1 point, f2 int)WITH(storage_type = astore);

-- Create an index.
gaussdb=# CREATE INDEX gist_box_index ON gist_box_tbl USING gist(f1);

-- Insert data.
gaussdb=# INSERT INTO gist_box_tbl VALUES(point(1,2),1);
gaussdb=# INSERT INTO gist_box_tbl VALUES(point(3,4),3);

-- Query statement.
gaussdb=# SELECT count(*) FROM gist_box_tbl WHERE f1 ~= point(1, 2);

-- Drop the table.
gaussdb=# DROP TABLE gist_box_tbl CASCADE;
```

Helpful Links

ALTER INDEX and **DROP INDEX**

1.13.9.27 CREATE LANGUAGE

Description

Defines a new procedural language.

Precautions

This syntax is not supported currently.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.9.28 CREATE MASKING POLICY

Description

Creates a masking policy.

Precautions

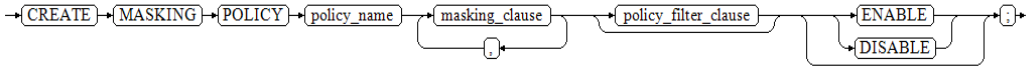
- Only users with the POLADMIN or SYSADMIN permission and the initial user can perform this operation.
- The masking policy takes effect only after the security policy is enabled, that is, the GUC parameter **enable_security_policy** is set to **on**.

NOTE

When you use database links to perform operations on remote objects, the client initiates a database link request. The actual sender is the server, and the attributes such as the IP address of the sender are the values of the server. For details, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.

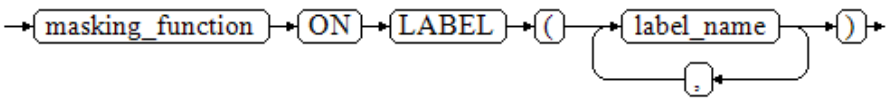
Syntax

```
CREATE MASKING POLICY policy_name masking_clause[, ...] [ policy_filter_clause ] [ENABLE | DISABLE];
```



- masking_clause

  ```
  masking_function ON LABEL(label_name[, ...])
  ```

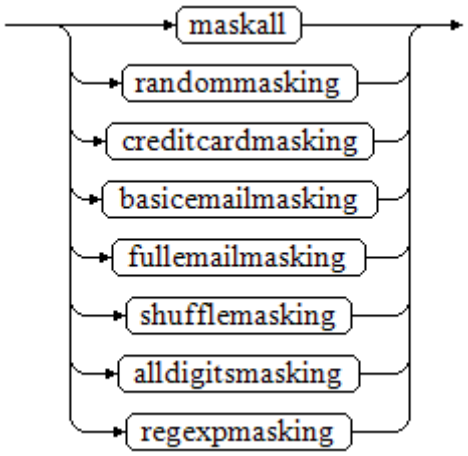
  

- masking_function

  maskall is not a preset function and cannot be displayed by running **\df**.

  The preset masking methods are as follows:

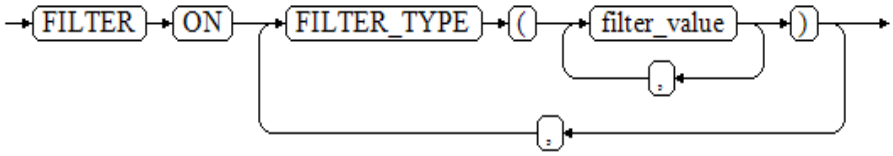
  ```
  { maskall | randommasking | creditcardmasking | basicemailmasking | fullemailmasking |
  shufflemasking | alldigitsmasking | regexpmasking }
  ```

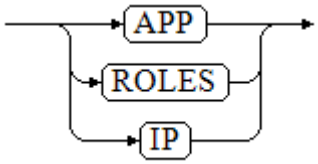
- policy_filter_clause:

  ```
  FILTER ON { FILTER_TYPE ( filter_value [, ...] ) } [, ...]
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



- FILTER_TYPE:
  { APP | ROLES | IP }



Parameters

- **policy_name**

  Specifies the name of a masking policy, which must be unique.

  Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

  NOTE

  - If a masking policy name contains more than 63 characters, the database truncates it and retains the first 63 characters.
  - If a masking policy name contains uppercase letters, the database automatically converts the uppercase letters into lowercase letters. To create a masking policy name that contains uppercase letters, enclose the masking policy name with double quotation marks ("").
  - The masking policy name must start with a letter or underscore (_) and can contain only lowercase letters (a to z), uppercase letters (A to Z), underscores (_), digits (0 to 9), and dollar signs ($).

- **label_name**

  Specifies the resource label name.

- **masking_clause**

  Specifies the identifier of the masking function used to mask specified database resources (**label_name**). You can specify the masking function by using schema.function.

- **policy_filter**

  Specifies the identifier of the user for whom the masking policy takes effect. If this parameter is left empty, the masking policy takes effect for all users.

- **FILTER_TYPE**

  Specifies the types of information to be filtered by the policy, including **IP**, **APP**, and **ROLES**.

- **filter_value**

  Specifies the detailed information to be filtered, such as the IP address, app name, and username.

- **ENABLE | DISABLE**

  Specifies whether to enable the masking policy. If **ENABLE** or **DISABLE** is not specified, **ENABLE** is used by default.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

- Create a masking policy that uses maskall to mask all string values as **x** and display default values for other data types.

```
-- Create table tb_for_masking.
gaussdb=# CREATE TABLE tb_for_masking(idx int, col1 text, col2 text, col3 text, col4 text, col5 text,
col6 text, col7 text,col8 text);

-- Insert data into the tb_for_masking table.
gaussdb=# INSERT INTO tb_for_masking VALUES(1, '9876543210', 'usr321usr', 'abc@huawei.com',
'abc@huawei.com', '1234-4567-7890-0123', 'abcdef 123456 ui 323 jsfd321 j3k2l3',
'4880-9898-4545-2525', 'this is a llt case');

-- View data.
gaussdb=# SELECT * FROM tb_for_masking;
 idx |    col1    |   col2    |      col3      |      col4      |       col5         |               col6
     |        col7         |        col8
-----+------------+-----------+----------------+----------------+--------------------
+-------------------------------
------+---------------------+--------------------
   1 | 9876543210 | usr321usr | abc@huawei.com | abc@huawei.com | 1234-4567-7890-0123 | abcdef
123456 ui 323 jsfd321 j
3k2l3 | 4880-9898-4545-2525 | this is a llt case
(1 row)

-- Create a resource label for sensitive column col1.
gaussdb=# CREATE RESOURCE LABEL mask_lb1 ADD COLUMN(tb_for_masking.col1);

-- Create a masking policy named maskpol1.
gaussdb=# CREATE MASKING POLICY maskpol1 maskall ON LABEL(mask_lb1);

-- Access the tb_for_masking table. The masking policy is triggered for the col1 column.
gaussdb=# SELECT col1 FROM tb_for_masking;
    col1
------------
 xxxxxxxxxx
(1 row)
```

- Create a masking policy that uses randommasking to mask values of the character string type to random numbers. The masked values are random each time.

```
-- Create a resource label for sensitive column col2.
gaussdb=# CREATE RESOURCE LABEL mask_lb2 ADD COLUMN(tb_for_masking.col2);

-- Create a masking policy named maskpol2.
gaussdb=# CREATE MASKING POLICY maskpol2 randommasking ON LABEL(mask_lb2);

-- Access the tb_for_masking table. The masking policy is triggered for the col2 column.
gaussdb=# SELECT col2 FROM tb_for_masking;
   col2
-----------
 0e8612d9a
(1 row)
```

- Create a masking policy that uses basicemailmasking to set all data content before the at sign (@) in the mailbox format value of the character string type to **x**.

```
-- Create a resource label for sensitive column col3.
gaussdb=# CREATE RESOURCE LABEL mask_lb3 ADD COLUMN(tb_for_masking.col3);

-- Create a masking policy named maskpol3.
gaussdb=# CREATE MASKING POLICY maskpol3 basicemailmasking ON LABEL(mask_lb3);

-- Access the tb_for_masking table. The masking policy is triggered for the col3 column.
gaussdb=# SELECT col3 FROM tb_for_masking;
     col3
----------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 xxx@huawei.com
(1 row)
```

- Create a masking policy that uses fullemailmasking to retain only the at sign (@) and **.com** in the email address format of the character string type, and set other parameters to **x**.

```
-- Create a resource label for sensitive column col4.
gaussdb=# CREATE RESOURCE LABEL mask_lb4 ADD COLUMN(tb_for_masking.col4);

-- Create a masking policy named maskpol4.
gaussdb=# CREATE MASKING POLICY maskpol4 fullemailmasking ON LABEL(mask_lb4);

-- Access the tb_for_masking table. The masking policy is triggered for the col4 column.
gaussdb=# SELECT col4 FROM tb_for_masking;
    col4
----------------
 xxx@xxxxxx.com
(1 row)
```

- Create a masking policy that uses creditcardmasking to retain the hyphen (-) and the last four digits for values of the string type, and set the rest to **x**.

```
-- Create a resource label for sensitive column col5.
gaussdb=# CREATE RESOURCE LABEL mask_lb5 ADD COLUMN(tb_for_masking.col5);

-- Create a masking policy named maskpol5.
gaussdb=# CREATE MASKING POLICY maskpol5 creditcardmasking ON LABEL(mask_lb5);

-- Access the tb_for_masking table. The masking policy is triggered for the col5 column.
gaussdb=# SELECT col5 FROM tb_for_masking;
       col5
--------------------
 xxxx-xxxx-xxxx-0123
(1 row)
```

- Create a masking policy that uses shufflemasking to shuffle the positions of characters in the value of the character string type.

```
-- Create a resource label for sensitive column col6.
gaussdb=# CREATE RESOURCE LABEL mask_lb6 ADD COLUMN(tb_for_masking.col6);

-- Create a masking policy named maskpol6.
gaussdb=#  CREATE MASKING POLICY maskpol6 shufflemasking ON LABEL(mask_lb6);

-- Access the tb_for_masking table. The masking policy is triggered for the col6 column.
gaussdb=# SELECT col6 FROM tb_for_masking;
              col6
-------------------------------------
 2 b6jusfd54c3312 13d23lk3jf3  2eai
(1 row)
```

- Create a masking policy that uses regexpmasking to mask values of the string type using regular expressions.

```
-- Create a resource label for sensitive column col7.
gaussdb=# CREATE RESOURCE LABEL mask_lb7 ADD COLUMN(tb_for_masking.col7);

-- Create a masking policy named maskpol7.
gaussdb=# CREATE MASKING POLICY maskpol7 regexpmasking('[\d+]','*',2, 9) ON LABEL(mask_lb7);

-- Access the tb_for_masking table. The masking policy is triggered for the col7 column.
gaussdb=# SELECT col7 FROM tb_for_masking;
       col7
--------------------
 48**-****-*545-2525
(1 row)
```

- Create a masking policy that takes effect only for scenarios where users are **dev_mask** and **bob_mask**, the client tool is gsql, and IP addresses are **172.31.17.160** and **127.0.0.0/24**.

```
-- Create users dev_mask and bob_mask.
gaussdb=# CREATE USER dev_mask PASSWORD '********';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CREATE USER bob_mask PASSWORD '*********';

-- Create a resource label for sensitive column col8.
gaussdb=# CREATE RESOURCE LABEL mask_lb8 ADD COLUMN(tb_for_masking.col8);

-- Create a masking policy named maskpol8.
gaussdb=# CREATE MASKING POLICY maskpol8 randommasking ON LABEL(mask_lb8) FILTER ON
ROLES(dev_mask, bob_mask), APP(gsql), IP('172.31.17.160', '127.0.0.0/24');

-- Access the tb_for_masking table. The masking policy is triggered for the col8 column.
gaussdb=# SELECT * FROM tb_for_masking;

-- Use the gsql tool, set the IP address to 172.31.17.160, and view tb_for_masking as user dev_mask.
gaussdb=# GRANT ALL PRIVILEGES TO dev_mask;

-- Use maskpol8 for data masking. The result is random and different each time.
gaussdb=# SELECT col8 FROM tb_for_masking;
       col8
--------------------
 cf32a9aa427f219ab0
(1 row)

gaussdb=# SELECT col8 FROM tb_for_masking;
       col8
--------------------
 13efa056dda1e1a474
(1 row)
```

- Drop data.

```
-- Drop masking policies.
gaussdb=# DROP MASKING POLICY maskpol1, maskpol2, maskpol3, maskpol4, maskpol5, maskpol6,
maskpol7, maskpol8;

-- Drop resource labels.
gaussdb=# DROP RESOURCE LABEL mask_lb1, mask_lb2, mask_lb3, mask_lb4, mask_lb5, mask_lb6,
mask_lb7, mask_lb8;

-- Drop the tb_for_masking table.
gaussdb=# DROP TABLE tb_for_masking;

-- Drop the dev_mask and bob_mask users.
gaussdb=# DROP USER dev_mask, bob_mask;
```

Helpful Links

ALTER MASKING POLICY and **DROP MASKING POLICY**

1.13.9.29 CREATE MATERIALIZED VIEW

Creates a complete-refresh materialized view. You can use REFRESH MATERIALIZED VIEW (complete-refresh) to refresh the data in the materialized view.

The CREATE MATERIALIZED VIEW statement has similar functions as the CREATE TABLE AS statement. However, the CREATE MATERIALIZED VIEW statement persistently stores the initialization query statement to support subsequent data refresh operations. Materialized views have the same persistence property as ordinary tables. However, unlike temporary tables, the system does not support the creation of temporary materialized views.

NOTE

For details about the application scenarios, see "Basic Concepts > Materialized Views" in *Developer Guide*.

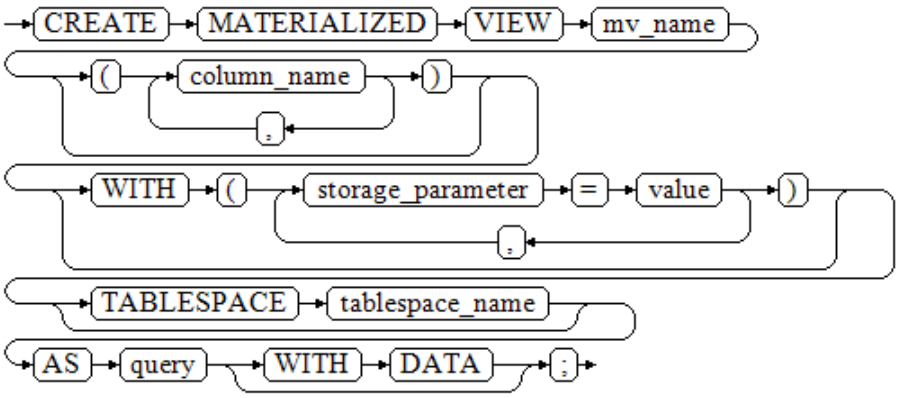
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

- Complete-refresh materialized view cannot be created on temporary tables or global temporary tables.
- Complete-refresh materialized views do not support node groups.
- After a complete-refresh materialized view is created, some DDL operations in the base table are no longer supported.
- IUD operations cannot be performed on complete-refresh materialized views.
- After a complete-refresh materialized view is created, if the base table data changes, you need to run the refresh command to synchronize the materialized view with the base table.
- Ustore does not support the creation and use of materialized views.
- The segment-page does not support the creation and use of materialized views.

Syntax

```
CREATE MATERIALIZED VIEW mv_name
   [ (column_name [, ...] ) ]
   [ WITH ( {storage_parameter = value} [, ... ] ) ]
   [ TABLESPACE tablespace_name ]
   AS query
   [ WITH DATA ];
```



Parameters

- **mv_name**

  Name of the created materialized view, which can be specified using a schema.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **column_name**

  Name of a column in the new materialized view. The materialized view supports specified columns. The number of specified columns must be the same as that of columns in the result of the subsequent query statement. If no column name is provided, the column name is obtained from the output column name of the query.

  Value range: character strings complying with **Identifier Naming Conventions**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **WITH (storage_parameter [= value] [, ...])**

  Specifies an optional storage parameter for a table or index. For details, see **CREATE TABLE**.

- **TABLESPACE tablespace_name**

  (Optional) Tablespace to which the new materialized view belongs. If it is not specified, the default tablespace is used.

- **AS query**

  A **SELECT**, **TABLE**, or **VALUES** command. This query will be run in a security-constrained operation.

Examples

```
-- Create a tablespace.
gaussdb=# CREATE TABLESPACE tbs_data1 RELATIVE LOCATION 'tablespace1/tbs_data1';

-- Create an ordinary table.
gaussdb=# CREATE TABLE my_table (c1 int, c2 int);

-- Create a complete-refresh materialized view.
gaussdb=# CREATE MATERIALIZED VIEW my_mv TABLESPACE tbs_data1 AS SELECT * FROM my_table;

-- Write data to the base table.
gaussdb=# INSERT INTO my_table VALUES(1,1),(2,2);

-- Query the complete-refresh materialized view.
gaussdb=# SELECT * FROM my_mv;
 c1 | c2
----+----
(0 rows)

-- Completely refresh the complete-refresh materialized view my_mv.
gaussdb=# REFRESH MATERIALIZED VIEW my_mv;

-- Query the complete-refresh materialized view.
gaussdb=# SELECT * FROM my_mv;
 c1 | c2
----+----
  1 |  1
  2 |  2
(2 rows)

-- Delete the complete-refresh materialized view.
gaussdb=# DROP MATERIALIZED VIEW my_mv;

-- Delete the ordinary table my_table.
gaussdb=# DROP TABLE my_table;

-- Delete the tablespace.
gaussdb=# DROP TABLESPACE tbs_data1;
```

Helpful Links

ALTER MATERIALIZED VIEW, **CREATE INCREMENTAL MATERIALIZED VIEW**, **CREATE TABLE**, **DROP MATERIALIZED VIEW**, **REFRESH INCREMENTAL MATERIALIZED VIEW**, and **REFRESH MATERIALIZED VIEW**

1.13.9.30 CREATE MODEL

Description

Trains a machine learning model and saves it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

This syntax is not supported currently.

1.13.9.31 CREATE NODE

Description

Creates a cluster node.

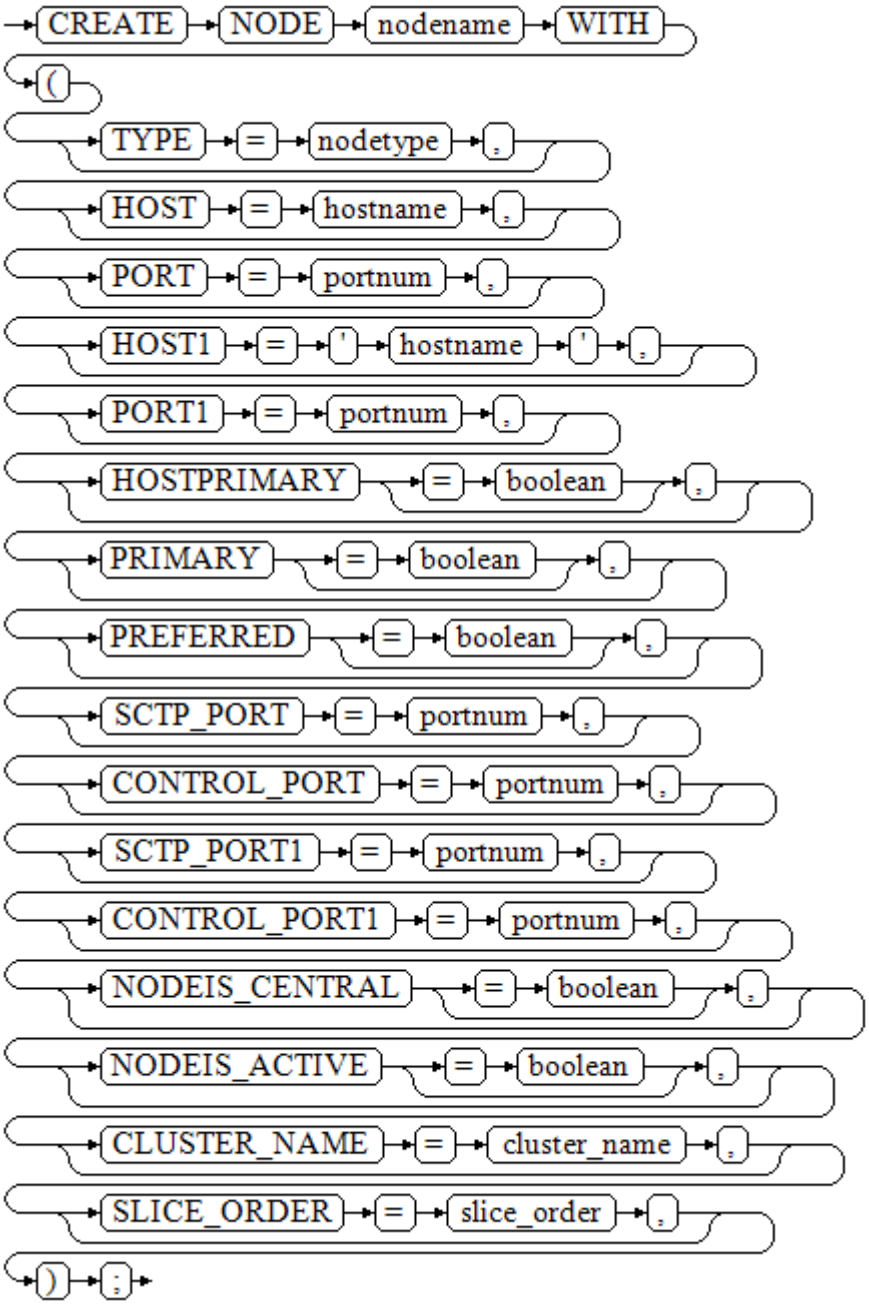
Precautions

- **CREATE NODE** is an API of the cluster management tool and is used to manage clusters. Only administrators have the permission to use this API.
- Do not use this API directly. Otherwise, the cluster status may be affected and the database may be unavailable.

Syntax

```
CREATE NODE nodename WITH
 (
   [ TYPE = nodetype,]
   [ HOST = hostname,]
   [ PORT = portnum,]
   [ HOST1 = 'hostname',]
   [ PORT1 = portnum,]
   [ HOSTPRIMARY [ = boolean ],]
   [ PRIMARY [ = boolean ],]
   [ PREFERRED [ = boolean ],]
   [ SCTP_PORT = portnum,]
   [ CONTROL_PORT = portnum,]
   [ SCTP_PORT1 = portnum,]
   [ CONTROL_PORT1 = portnum, ]
   [ NODEIS_CENTRAL [= boolean], ]
   [ NODEIS_ACTIVE [= boolean],]
   [ CLUSTER_NAME = cluster_name, ]
   [ SLICE_ORDER = slice_order, ]
 );
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



NOTE

The port whose number is specified by **PORT** is used for internal communications between nodes. Unlike the port connecting to an external client, it can be queried in the **pgxc_node** table.

Parameters

- **nodename**

  Specifies the node name.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **TYPE = nodetype**

Specifies the type of a node.

Value range:

- 'coordinator'
- 'datanode'
- 'coordinator_standbycluster'
- 'datanode_standbycluster_main'
- 'datanode_standbycluster_other'

- **HOST = hostname**

  Specifies the name or IP address of the primary node corresponding to a node.

- **PORT = portnum**

  Specifies the port number of the primary node to which a node is bound.

- **HOST1 = hostname**

  Specifies the name or IP address of the standby node corresponding to a node.

- **PORT1 = portnum**

  Specifies the port number of the standby node to which a node is bound.

- **HOSTPRIMARY**

  Specifies whether the host name or IP address is the primary one.

  Value range:

  - **true**
  - **false** (default value)

- **PRIMARY = boolean**

  Specifies whether the node is a primary node or not. A primary node allows read/write operations. A non-primary node allows only read operations.

  Value range:

  - **true**
  - **false** (default value)

- **PREFERRED = boolean**

  Specifies whether the node is a preferred node for read operations.

  Value range:

  - **true**
  - **false** (default value)

- **SCTP_PORT = portnum**

  Specifies the port used by the TCP proxy communications library of the primary node to listen on the data transmission channel. TCP is used to listen on connections.

- **CONTROL_PORT = portnum**

  Specifies the port used by the TCP proxy communications library of the primary node to listen on the control transmission channel. The TCP protocol is used to listen on connections.

- **SCTP_PORT1 = portnum**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the port used by the TCP proxy communications library of the standby node to listen on the data transmission channel. TCP is used to listen on connections.

- **CONTROL_PORT 1= portnum**

  Specifies the port used by the TCP proxy communications library of the standby node to listen on the control transmission channel. The TCP protocol is used to listen on connections.

- **NODEIS_CENTRAL**

  Specifies whether the current node is a central control node. It is used only for CNs and is invalid for DNs.

  - **true**
  - **false** (default value)

- **NODEIS_ACTIVE**

  Specifies whether the current node is normal. It is used to mark whether the CN is removed and is invalid for DNs.

  - **true**
  - **false** (default value)

- **CLUSTER_NAME**

  Specifies the cluster name.

- **SLICE_ORDER**

  Specifies the sequence number of the slice to which a node belongs.

  NOTE

  In multi-CN environments, when you create a cluster node, it is only generated on the CN that is currently connected and is not synchronized with other CNs. This requires you to repeat the creation process on all CNs. However, to drop the cluster node created, you only need to connect to one CN. If a cluster node is created on only one CN, you need to define the current CN name and drop the cluster node on the current CN. For details, see **DROP NODE**.

Examples

```
-- Create cluster nodes on all CNs.
gaussdb=# CREATE NODE datanode1 WITH(
   TYPE = datanode,
   PREFERRED = false
);
gaussdb=# CREATE NODE datanode2 WITH(
   TYPE = datanode,
   PREFERRED = false
);

-- Query the initial cluster DN status.
gaussdb=# SELECT node_name, nodeis_preferred FROM pgxc_node WHERE node_type = 'D' ORDER BY 1;
 node_name | nodeis_preferred
-----------+------------------
 datanode1 | f
 datanode2 | f
(2 rows)

-- Set datanode1 as the preferred DN.
gaussdb=# ALTER NODE datanode1 WITH(preferred = true);
-- Query the cluster DN status after the change.
gaussdb=# SELECT node_name, nodeis_preferred FROM pgxc_node WHERE node_type = 'D' ORDER BY 1;
 node_name | nodeis_preferred
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-----------+------------------
 datanode1 | t
 datanode2 | f
(2 rows)

-- Connect to only one CN to drop cluster nodes.
gaussdb=# DROP NODE datanode1;
gaussdb=# DROP NODE datanode2;
```

Helpful Links

ALTER NODE and **DROP NODE**

1.13.9.32 CREATE NODE GROUP

Description

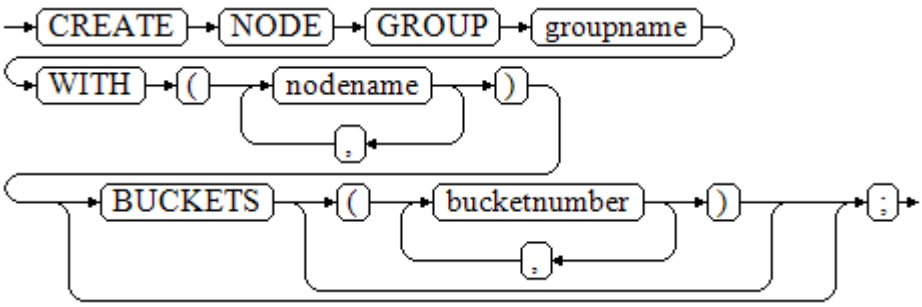
Creates a cluster node group.

Precautions

- CREATE NODE GROUP is an API of the cluster management tool.
- Only a system administrator has the permission.

Syntax

```
CREATE NODE GROUP groupname
   WITH ( nodename [, ... ] )
   [ BUCKETS [ ( bucketnumber [, ... ] ) ] ];
```



Parameters

- **groupname**

  Specifies the name of a node group.

  Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

  NOTE

  A node group name supports all ASCII characters, but you are advised to name a node group according to the **Identifier Naming Conventions**.

- **nodename**

  Specifies the node name.

  Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **BUCKETS [(bucketnumber [, ...])]**

  This is an internal use of the cluster management tool and can be called only during scale-out.

Examples

```
-- Query the initial cluster DN status.
gaussdb=# SELECT node_name, nodeis_preferred FROM pgxc_node WHERE node_type = 'D' ORDER BY 1;
 node_name | nodeis_preferred
-----------+------------------
 dn_6001_6002_6003 | f
 dn_6004_6005_6006 | f
 dn_6007_6008_6009 | f
(3 rows)

-- Create a node group and replace dn_6001_6002_6003 with the actual node name obtained in the previous step.
gaussdb=# CREATE NODE GROUP test_group WITH ( dn_6001_6002_6003 );

-- Query the node group.
gaussdb=# SELECT group_name, group_members FROM pgxc_group;
   group_name   |   group_members
----------------+-------------------
 group_version1 | 16384 16388 16394
 test_group     | 16384
(2 rows)

-- Delete the node group.
gaussdb=# DROP NODE GROUP test_group;
```

Helpful Links

ALTER NODE GROUP and **DROP NODE GROUP**

1.13.9.33 CREATE PACKAGE

Description

Creates a package.

Precautions

- Packages can be used only in ORA-compatible databases.
- Functions or stored procedures declared in the package specification must be clearly defined in the package body.
- When a stored procedure is created, a write lock is added only to the stored procedure or package, and a read lock is added only to the functions and packages on which the functions depend during compilation and execution.
- During instantiation, stored procedures with COMMIT or ROLLBACK cannot be called.
- The PACKAGE function cannot be called in a trigger.
- Variables in a package cannot be directly used in external SQL statements.
- Private variables and stored procedures in a package cannot be called outside the package.
- If a usage is not supported in a stored procedure, it is not supported in a package. For example, if COMMIT/ROLLBACK cannot be called in a function, COMMIT/ROLLBACK cannot be called in a function of a package.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The name of a schema cannot be the same as that of a package.
- Only stored procedures and function definitions compatible with Oracle Database are supported.
- Variables with the same name in a package, including parameters with the same name in a package, are not supported.
- The global variables in a package are at the session level. The variables in packages cannot be shared in different sessions.
- When a function of an autonomous transaction is called in a package, cursor variables in the package and functions that use cursor variables in the package cannot be used recursively.
- A package does not declare REF CURSOR variables.
- The default permission on a package is SECURITY INVOKER. To change the default permission to SECURITY DEFINER, set the GUC parameter **behavior_compat_options** to **'plsql_security_definer'**.
- A user with the CREATE ANY PACKAGE permission can create packages in the public and user schemas.
- If the name of a package to be created contains special characters, the special characters cannot contain spaces. You are advised to set the GUC parameter **behavior_compat_options** to **"skip_insert_gs_source"**. Otherwise, an error may occur.
- When you create a package function, the default parameter value can contain variables in the package but cannot contain cross-package variables.
- When the complex function is called in a package, for example, func(x).a, a composite type is returned. Cross-schema call is supported, but the database.schema.package.func(x).b call is not supported.
- When a stored procedure in a package is created, if the stored procedure name is in the *schema*.*func* or *package*.*func* format, only the value of *func* is obtained, and the declaration of *schema* or *package* is not obtained. To disable this behavior, set the GUC parameter **behavior_compat_options** to **'forbid_package_function_with_prefix'**.
- **view%RowType** cannot be used as a function parameter.
- Composite records cannot be used in INSERT VALUES in stored procedures or functions.
- SAVE EXCEPTIONS (used in called stored procedures) is not supported currently.
- Expressions do not support COMMIT/ROLLBACK.
- For a function without parameters, parentheses cannot be omitted.
- User-defined variables cannot be pushed down.
- When a stored procedure in a package is created, if the input parameters of the stored procedure contain pseudo-type parameters (for example, **anyelement**), variable-length parameters cannot be specified during a call.
- Cursors created in a package or cursors in functions with parameters can be referenced (**%RowType**) and opened across packages. The restrictions are as follows:
  - Currently, cursors cannot be called in database.schema.package.cursor mode.

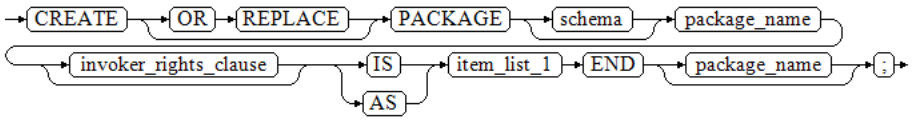
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Cursors cannot be defined as function parameters.
  - **cursor %RowType** is not supported when a cursor is referenced.
- If the PRAGMA AUTONOMOUS_TRANSACTION syntax is declared in PACKAGE SPECIFICATION or PACKAGE BODY SPECIFICATION, a package can be created, but the autonomous transaction does not take effect.
- A non-initial user cannot perform the REPLACE operation on packages of the initial user.
- When a package is created, only the validity of the default values of the variables in the package specification and package body is checked (whether the function on which the default values depend exists and whether the function is valid). Related statements are not executed, the initialization of package header and package body variables and instantiation of the package body (PACKAGE BODY ANONYMOUS BLOCK) are executed only once when the API is called for the first time.
- During package creation, recompilation, and nested compilation, package instantiation (including packet header, package body variable initialization, and package body instantiation) is not executed.

Syntax

- CREATE PACKAGE SPECIFICATION syntax format.

```
CREATE [ OR REPLACE ] PACKAGE [ schema ] package_name
  [ invoker_rights_clause ] { IS | AS } item_list_1 END [ package_name ];
```



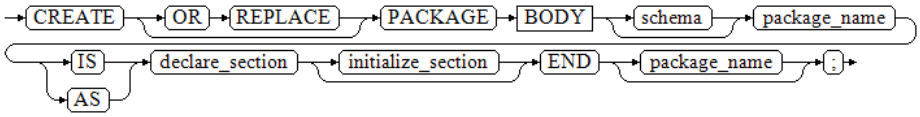
  - **invoker_rights_clause**: You can use **AUTHID DEFINER** or **AUTHID CURRENT_USER** to declare the caller's permission in the package header. **AUTHID DEFINER** indicates the creator's permissions, and **AUTHID CURRENT_USER** indicates the caller's permissions. Variables, functions, or stored procedures in the package header are executed with the permission specified during declaration (**AUTHID DEFINER** or **AUTHID CURRENT_USER**), while cursors in the package header are executed with the caller's permission (**AUTHID CURRENT_USER**).
  - **item_list_1** can be a declared variable, stored procedure, or function.
  - The **package_name** after END can be omitted. (For the SQL file generated by gs_dump, the package name after END is automatically added.) If **package_name** is not specified, the end character (/) must be added to the end of the CREATE PACKAGE SPECIFICATION syntax.

  NOTE

  PACKAGE SPECIFICATION (header) declares public variables, functions, and exceptions in a package, which can be called by external functions or stored procedures. It can only declare stored procedures and functions but cannot define them.

- CREATE PACKAGE BODY syntax format.

```
CREATE [ OR REPLACE ] PACKAGE BODY [ schema ] package_name
  { IS | AS } declare_section [ initialize_section ] END [ package_name ];
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The **package_name** after END can be omitted. (For the SQL file generated by gs_dump, the package name after END is automatically added.) If **package_name** is not specified, the end character (/) must be added to the end of the CREATE PACKAGE BODY syntax.

NOTE

- The package body defines private variables and functions of the package. If a variable or function is not declared by the package specification, it is a private variable or private function.
- The package body also has an instantiation part to initialize the package.

Examples

- CREATE PACKAGE

```
-- Create a database.
gaussdb=# CREATE DATABASE test DBCOMPATIBILITY 'ORA';
gaussdb=# \c test

-- Create a test table tbl_test.
test=# CREATE TABLE tbl_test(uid varchar(6) PRIMARY KEY, area_id varchar(5), level int);

-- Create a package header.
test=# CREATE OR REPLACE PACKAGE pkg_test AS
    -- Public stored procedures can be called by external systems.
    PROCEDURE proc_ist_tbl_test(v_uid in varchar, v_area_id varchar, v_level int);
    PROCEDURE proc_del_tbl_test(v_uid in varchar);
    PROCEDURE proc_upd_tbl_test(v_uid in varchar, v_area_id varchar, v_level int);
END pkg_test;
/

-- Create a package body.
test=# CREATE OR REPLACE PACKAGE BODY pkg_test AS
    PROCEDURE proc_ist_tbl_test(v_uid in varchar, v_area_id varchar, v_level int) AS
    BEGIN
        INSERT INTO tbl_test VALUES (v_uid, v_area_id, v_level);
    END;
    PROCEDURE proc_del_tbl_test(v_uid in varchar) AS
    BEGIN
        DELETE FROM tbl_test WHERE uid = v_uid;
    END;
    PROCEDURE proc_upd_tbl_test(v_uid in varchar, v_area_id varchar, v_level int) AS
    BEGIN
        UPDATE tbl_test SET area_id = v_area_id, level = v_level WHERE uid = v_uid;
    END;
    var4 int := 10;
    -- The instantiation starts.
BEGIN
    var4 := 10;
    dbe_output.print_line(var4);
END pkg_test;
/
```

- Calling a package

```
-- Use CALL to call the stored procedure of a package.
test=# CALL pkg_test.proc_ist_tbl_test('0A00B1','01001',24);
test=# SELECT * FROM tbl_test;
  uid   | area_id | level
--------+---------+---------
 0A00B1 | 01001   |     24
(1 row)

-- Use SELECT to call the stored procedure of a package.
test=# SELECT pkg_test.proc_upd_tbl_test('0A00B1','01001','26');
test=# SELECT * FROM tbl_test;
  uid   | area_id | level
--------+---------+-------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 0A00B1 | 01001   |    26
(1 row)

-- Call a stored procedure of a package by using an anonymous block.
test=# BEGIN
pkg_test.proc_del_tbl_test('0A00B1');
END;
/
test=# SELECT * FROM tbl_test;
 uid | area_id | level
-----+---------+-------
(0 rows)

-- Drop.
test=# DROP TABLE tbl_test;
test=# DROP PACKAGE pkg_test;
-- Switch to the default database and drop the test database. Change the database name as required.
test=# \c postgres
gaussdb=# DROP DATABASE test;
```

Helpful Links

ALTER PACKAGE and **DROP PACKAGE**

1.13.9.34 CREATE PROCEDURE

Description

Creates a stored procedure.

Precautions

- When a stored procedure is created, if the parameter or return value has precision, the precision is not checked.
- When creating a stored procedure, you are advised to specify the schemas of all operations on table objects in the stored procedure definition. Otherwise, the stored procedure may fail to be executed.
- When a stored procedure is created using CREATE, a write lock is added only to the stored procedure or package, and a read lock is added only to the functions and packages on which the functions depend during compilation and execution.
- **current_schema** and **search_path** specified by SET during stored procedure creation are invalid. **search_path** and **current_schema** before and after function execution should be the same.
- When the function is called by SELECT or CALL, an argument must be provided in the output parameter. The argument does not take effect.
- A stored procedure with the PACKAGE attribute can use overloaded functions.
- Do not create an overloaded stored procedure with different parameter names but same stored procedure name and parameter list type.
- Do not create a stored procedure that has the same name and parameter list as the function.
- Do not overload stored procedures with different default values.
- Only the IN, OUT, and INOUT parameters of the stored procedure cannot be reloaded after the GUC parameter **behavior_compat_options** is set to

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

'proc_outparam_override'. Overloading is supported after the GUC parameter is disabled.

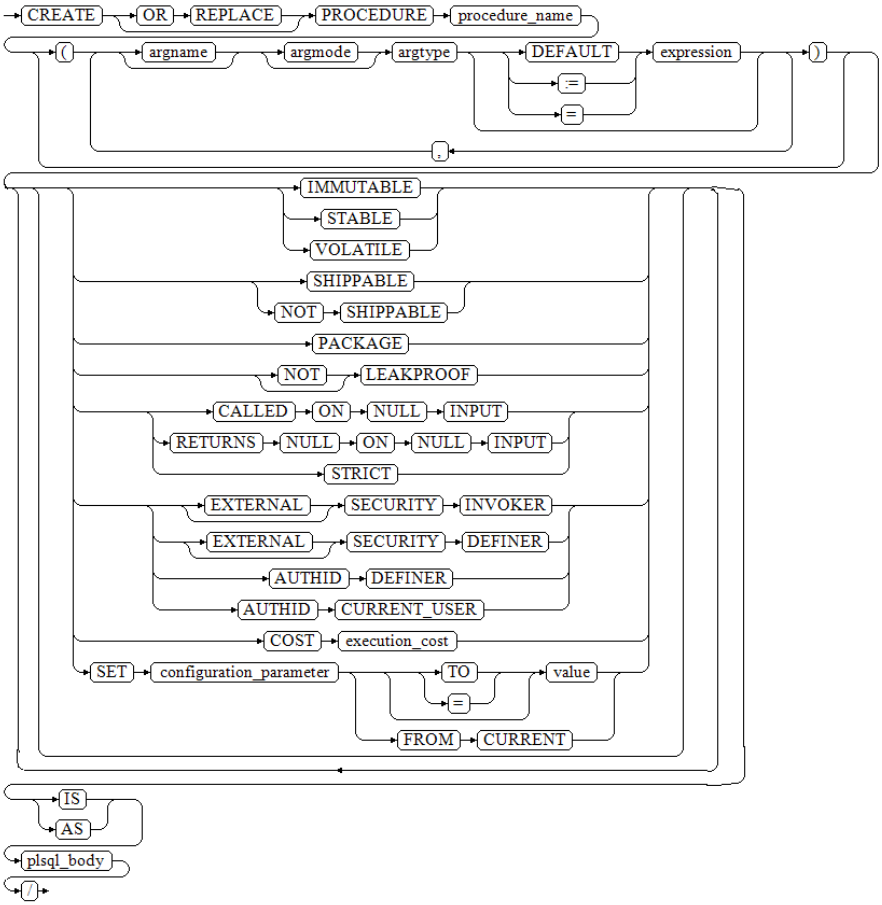
- Oracle-compatible functions are created for ORA-compatible databases and PostgreSQL-compatible functions are created for PG-compatible databases. Hybrid creation is not recommended.
- If a function supports overloading, you need to specify the keyword PACKAGE.
- When an overloaded stored procedure is called, the variable type must be specified.
- If an undeclared variable is used in a stored procedure, an error is reported when the stored procedure is called.
- When you create a procedure, you cannot nest average functions in aggregate or other functions.
- If the function is defined as the IMMUTABLE or SHIPPABLE type, avoid having INSERT, UPDATE, DELETE, MERGE, and DDL operations in the function. These operations require the CN to determine the execution node, and the execution result may be incorrect.
- The stored procedure does not support operations that will return a set.
- When stored procedures without parameters are called in another stored procedure, you can omit brackets and call stored procedures using their names directly.
- When other functions with output parameters are called in a stored procedure and an assignment expression, set the GUC parameter **behavior_compat_options** to **'proc_outparam_override'**, define variables of the same type as the output parameters, and use the variables as output parameters to call other functions with output parameters for the output parameters to take effect. Otherwise, the output parameters of the called functions will be ignored.
- The **OUT** parameter does not take effect in the following scenarios when it is used in an expression: The EXECUTE IMMEDIATE sqlv USING func syntax is used to execute a function. The SELECT func INTO syntax is used to execute a function. DML statements such as INSERT and UPDATE are used to execute a function. The SELECT WHERE a=func() method is used. A function with an **OUT** output parameter is used as an input parameter (that is, **fun(func(OUT b), a)**) and the **OUT b** parameter does not take effect. A function with an **OUT** output parameter is used to assign a default value to a variable.
- The stored procedure supports viewing, exporting, and importing parameter comments.
- The stored procedure supports viewing, exporting, and importing parameter comments between IS/AS and plsql_body.
- Users with the CREATE ANY FUNCTION permission can create or replace stored procedures in the user schemas.
- An **OUT** or **INOUT** parameter must be set to a variable but not a constant.
- The default permission on a stored procedure is SECURITY INVOKER. To change the default permission to SECURITY DEFINER, set the GUC parameter **behavior_compat_options** to **'plsql_security_definer'**.
- If a stored procedure with the definer specified is created in a schema of another user, the stored procedure will be executed by another user, which may cause unauthorized operations. Therefore, exercise caution when performing this operation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- When the complex function is called in a stored procedure (for example, func(x).a), a composite type is returned. Cross-schema call is supported, but the database.schema.package.func(x).b call is not supported.
- When a stored procedure with the **OUT** parameter is called, you can set the GUC parameter **behavior_compat_options** to **'proc_outparam_transfer_length'** to pass the parameter length. The specifications and limitations are as follows:
  - The following types are supported: CHAR(n), CHARACTER(n), NCHAR(n), VARCHAR(n), VARYING(n), VARCHAR2(n), and NVARCHAR2(n).
  - If the **OUT** parameter does not take effect (for example, **PERFORM**), the length does not need to be passed.
  - Precisions cannot be specified for the following types: NUMERIC, DECIMAL, NUMBER, FLOAT, DEC, INTEGER, TIME, TIMESTAMP, INTERVAL, TIME WITH TIME ZONE, TIMESTAMP WITH TIME ZONE, TIME WITHOUT TIME ZONE, and TIMESTAMP WITHOUT TIME ZONE.
  - The parameter length can be passed regardless of whether the GUC parameter **behavior_compat_options** is set to **proc_outparam_override**.
  - To pass the length of elements of the collection type and the length of elements of the array type nested by the collection type, you need to set the GUC parameter **behavior_compat_options** to **tableof_elem_constraints**.
- Functions contain syntax and functions that use GUC parameters to control features. If GUC parameters are modified in a session, the function may retain the behavior before the modification. Therefore, exercise caution when modifying GUC parameters.
- Non-initial users do not have the permission to perform the REPLACE operation on the stored procedures of the initial user.
- When you perform operations related to passwords or keys, such as user creation, password change, and encryption/decryption, in a stored procedure body, the system catalogs and logs record plaintext information about the passwords or keys. To prevent sensitive information leakage, you are advised not to perform password-related operations in a stored procedure body.

Syntax

```
CREATE [ OR REPLACE ] PROCEDURE procedure_name
   [ ( {[ argname ] [ argmode ] argtype [ { DEFAULT | := | = } expression ]}[,...]) ]
   [
     { IMMUTABLE | STABLE | VOLATILE }
     | { SHIPPABLE | NOT SHIPPABLE }
     | {PACKAGE}
     | [ NOT ] LEAKPROOF
     | { CALLED ON NULL INPUT | RETURNS NULL ON NULL INPUT | STRICT }
     | {[ EXTERNAL ] SECURITY INVOKER | [ EXTERNAL ] SECURITY DEFINER | AUTHID DEFINER | AUTHID
CURRENT_USER}
     | COST execution_cost
     | SET configuration_parameter { [ TO | = ] value | FROM CURRENT }
   ][ ... ]
 { IS | AS }
plsql_body
/
```



Parameters

- **OR REPLACE**

  Replaces the original definition when two stored procedures are with the same name.

- **procedure_name**

  Specifies the name (optionally schema-qualified) of the stored procedure to be created.

  Value range: character strings complying with **Identifier Naming Conventions**.

  NOTE

  You are advised not to create a function with the same name as a system function. Otherwise, you need to specify the schema of the function when calling the function.

- **argname**

  Specifies the argument name.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **argmode**

  Specifies the mode of an argument.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: **IN**, **OUT**, **INOUT**, or **VARIADIC**. The default value is **IN**. **VARIADIC** can be added only to parameters in OUT mode, and parameters in OUT and INOUT modes cannot be used in the RETURNS TABLE procedure definition.

NOTE

VARIADIC specifies parameters of the array type.

- **argtype**

  Specifies the type of an argument. You can use **%RowType** to indirectly reference the type of a table, or **%Type** to indirectly reference the type of a column in a table or composite type.

  Value range: a valid data type.

  NOTE

  No specific order is applied to **argname** and **argname**. The following order is advised: **argname**, **argmode**, and **argtype**.

- **expression**

  Specifies the default expression of a parameter.

  NOTE

  - If **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s2**, the default expression is not supported when the parameter is in INOUT mode.
  - It is recommended that you define all default parameters after all non-default parameters.
  - If a function with default parameters is called, input parameters are added to the function from left to right. If inputs of non-default parameters are missing, an error is reported.
  - If **proc_uncheck_default_param** is enabled and a function with default parameters is called, input parameters are added to the function from left to right. The default input parameters can be left empty. If any non-default parameter is left empty, it will be padded using a misplaced default value.
  - When **a_format_version** is set to **10c**, **a_format_dev_version** is set to **s1**, **proc_outparam_override** is disabled, and the function parameters include the output parameter **OUT** and **DEFAULT**, the default value cannot be omitted.

- **IMMUTABLE | STABLE**

  Specifies a constraint.

  The function of each parameter is similar to that of CREATE FUNCTION. For details, see **CREATE FUNCTION**.

- **plsql_body**

  Specifies the PL/SQL stored procedure body.

  When a stored procedure is created, the **PLSQL_BODY** can end with "END;" or "END procedure_name;".

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

Constraints for stored procedures ending with "END procedure_name;":

- It is supported only in ORA-compatible mode (database level).
- Only the syntax for creating Oracle-compatible databases is supported.
- Only the scenario where the stored procedure name is set after END is supported.
- In the **DBE_PLDEVELOPER.gs_source**, **my_source**, **db_source** and **adm_source** views, the definition of the stored procedure stays consistent (if the created procedure is "END+name", it is displayed as "END+name"). In the **pg_proc** and **\sf** views, the outermost END is not followed by a name in any case. The innermost END is displayed as defined during creation (if the created procedure is "END +name", it is displayed as "END+name").
- The stored procedure cannot retain the name following the outermost END in the SQL file generated by gs_dump.
- If a stored procedure is named IF/LOOP or a subprogram is nested, the stored procedure cannot be ended with END IF/LOOP.

Examples

- Create a stored procedure.

```
-- Create a stored procedure and return the sum of the input parameters.
gaussdb=# CREATE OR REPLACE PROCEDURE proc_add(i int, j int)
AS
BEGIN
    dbe_output.print_line('result is: '|| i+j);
END;
/

-- Use CALL to call a stored procedure.
gaussdb=# CALL proc_add(16,17);

-- Use a program block to call a stored procedure.
gaussdb=# BEGIN
   proc_add(16,17);
END;
/

-- Drop.
gaussdb=# DROP PROCEDURE proc_add;
```

- Create a stored procedure ending with END in an ORA-compatible database.

```
-- Create a stored procedure and print the sum of the input parameters.
gaussdb=# CREATE OR REPLACE PROCEDURE proc_add(i int, j int)
AS
BEGIN
    dbe_output.print_line('result is: '|| i+j);
END proc_add;
/

-- Use CALL to call a stored procedure.
gaussdb=# CALL proc_add(16,17);

-- Use a program block to call a stored procedure.
gaussdb=# BEGIN
   proc_add(16,17);
END;
/

-- Drop.
gaussdb=# DROP PROCEDURE proc_add;
```

- Create a stored procedure whose parameter model is **VARIADIC**.

```
-- Create the stored procedure pro_variadic.
gaussdb=# CREATE OR REPLACE PROCEDURE pro_variadic (var1 VARCHAR2(10) DEFAULT 'hello!',var4
VARIADIC int4[])
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
AS
BEGIN
    dbe_output.print_line(var1);
    for i in 1..array_length(var4,1) loop
        dbe_output.print_line(var4[i]);
    end loop;
END;
/

-- Execute the stored procedure.
gaussdb=# SELECT pro_variadic(var1=>'hello', VARIADIC var4=> array[3,5,11,2]);

-- Drop.
gaussdb=# DROP PROCEDURE pro_variadic;
```

- Parameter models: **IN** and **OUT**
  - **IN** (default) indicates that the parameter is an input parameter.
  - **OUT** indicates that the parameter is an output parameter.
  - **IN OUT** indicates that the parameter is an INOUT parameter.

```
-- Create the stored procedure proc_add1. num1 and num2 are input parameters, and num3 is an output parameter.
gaussdb=# CREATE PROCEDURE proc_add1 (num1 in int, num2 in int, num3 out int)
AS
BEGIN
    num3 := num1 + num2;
END;
/

-- Use a program block to call the stored procedure and use variable c to receive the parameters sent by the stored procedure.
gaussdb=# DECLARE
    a int := 20;
    b int := 32;
    c int := 0;
BEGIN
    proc_add1(a,b,c);
    dbe_output.put_line(c);
END;
/

-- Drop.
gaussdb=# DROP PROCEDURE proc_add1;
```

Helpful Links

ALTER PROCEDURE and **DROP PROCEDURE**

1.13.9.35 CREATE RESOURCE LABEL

Description

Creates a resource label.

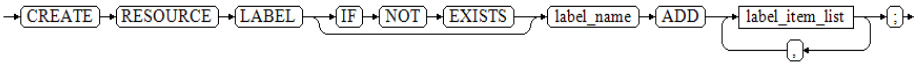
Precautions

Only users with the POLADMIN or SYSADMIN permission, or the initial user can perform this operation.

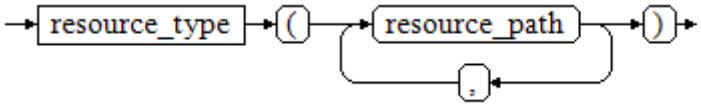
Syntax

```
CREATE RESOURCE LABEL [IF NOT EXISTS] label_name ADD label_item_list[, ...];
```

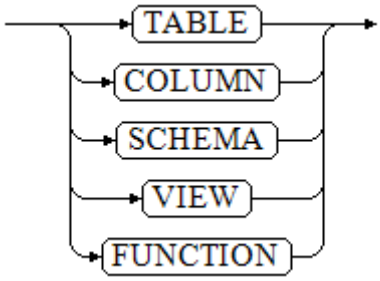
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



- label_item_list
  resource_type(resource_path[, ...])



- resource_type
  { TABLE | COLUMN | SCHEMA | VIEW | FUNCTION }



Parameters

- **IF NOT EXISTS**

  Checks whether a resource label with the same name already exists. If it exists, a prompt indicating that such resource label already exists will be displayed but no error will be generated.

- **label_name**

  Resource label name, which must be unique.

  Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

  NOTE

  - If the value contains more than 63 characters, the database truncates it and retains the first 63 characters as the resource label name.
  - If a resource label name contains uppercase letters, the database automatically converts the uppercase letters into lowercase letters. To create a resource label name that contains uppercase letters, enclose the resource label name with double quotation marks ("").
  - The audit policy name must start with a letter or underscore (_) and can contain only lowercase letters (a to z), uppercase letters (A to Z), underscores (_), digits (0 to 9), and dollar signs ($).

- **resource_type**

  Specifies the type of the database resource to be labeled.

  Value range: **TABLE**, **COLUMN**, **Schema**, **VIEW**, and **FUNCTION**.

- **resource_path**

  Specifies the path of the database resource.

Examples

```
-- Create table tb_for_label.
gaussdb=# CREATE TABLE tb_for_label(col1 text, col2 text, col3 text);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a resource label based on a table.
gaussdb=# CREATE RESOURCE LABEL IF NOT EXISTS table_label add TABLE(public.tb_for_label);

-- Create an existing table resource label again and compare the differences between adding IF NOT EXISTS and not adding IF NOT EXISTS.
gaussdb=# CREATE RESOURCE LABEL IF NOT EXISTS table_label add TABLE(public.tb_for_label);
NOTICE:  table_label label already defined, skipping
CREATE RESOURCE LABEL
gaussdb=# CREATE RESOURCE LABEL table_label add TABLE(public.tb_for_label);
ERROR:  table_label label already defined

-- Create a resource label based on columns.
gaussdb=# CREATE RESOURCE LABEL IF NOT EXISTS column_label add COLUMN(public.tb_for_label.col1);

-- Create schema schema_for_label.
gaussdb=# CREATE SCHEMA schema_for_label;

-- Create a resource label based on a schema.
gaussdb=# CREATE RESOURCE LABEL IF NOT EXISTS schema_label add SCHEMA(schema_for_label);

-- Create view view_for_label.
gaussdb=# CREATE VIEW view_for_label AS SELECT 1;

-- Create a resource label based on a view.
gaussdb=# CREATE RESOURCE LABEL IF NOT EXISTS view_label add VIEW(view_for_label);

-- Create function func_for_label.
gaussdb=# CREATE FUNCTION func_for_label RETURNS TEXT AS $$ SELECT col1 FROM tb_for_label; $$ LANGUAGE SQL;

-- Create a resource label based on a function.
gaussdb=# CREATE RESOURCE LABEL IF NOT EXISTS func_label add FUNCTION(func_for_label);

-- Drop the table resource label table_label.
gaussdb=# DROP RESOURCE LABEL IF EXISTS table_label;

-- Drop the column resource label column_label.
gaussdb=# DROP RESOURCE LABEL IF EXISTS column_label;

-- Drop the function resource label func_for_label.
gaussdb=# DROP FUNCTION func_for_label;

-- Drop the view resource label view_for_label.
gaussdb=# DROP VIEW view_for_label;

-- Drop the schema resource label schema_for_label.
gaussdb=# DROP SCHEMA schema_for_label;

-- Drop the tb_for_label table.
gaussdb=# DROP TABLE tb_for_label;
```

Helpful Links

ALTER RESOURCE LABEL and **DROP RESOURCE LABEL**

1.13.9.36 CREATE RESOURCE POOL

Description

Creates a resource pool and specifies the Cgroup of the resource pool.

Precautions

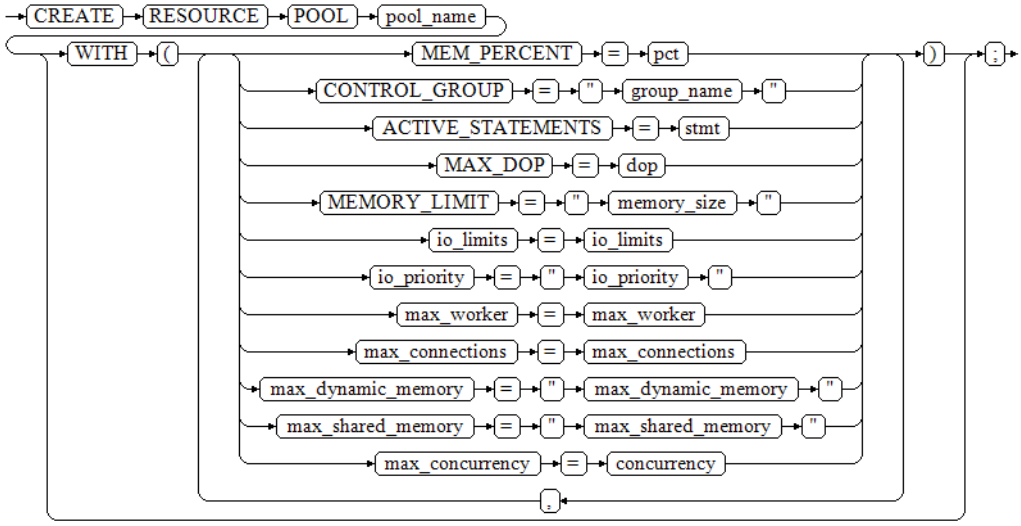
- Only users with the SYSADMIN permission or the initial user can perform this operation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- This syntax is not supported during upgrade.

Syntax

```
CREATE RESOURCE POOL pool_name
    [WITH ({MEM_PERCENT=pct | CONTROL_GROUP="group_name" | ACTIVE_STATEMENTS=stmt |
MAX_DOP = dop | MEMORY_LIMIT="memory_size" | io_limits=io_limits | io_priority="io_priority" |
max_worker=max_worker | max_connections=max_connections |
max_dynamic_memory="max_dynamic_memory" | max_shared_memory="max_shared_memory" |
max_concurrency=concurrency}[, ... ])];
```



Parameters

- **pool_name**

  The resource pool name must be unique and cannot be the same as an existing resource pool name in the current database.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **group_name**

  Specifies the name of a Cgroup.

  Value range: a string that complies with the following rules:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- You can use either double quotation marks ("") or single quotation marks (') in the syntax when setting the name of a Cgroup.
- The Cgroup name is case-sensitive.
- If **group_name** is not specified, the string "Medium" will be used by default in the syntax, indicating the **Medium** Timeshare Cgroup under **DefaultClass**.
- If an administrator specifies a Workload Cgroup under Class, for example, **control_group** set to **class1:workload1**, the resource pool will be associated with the **workload1** Cgroup under **class1**. The level of **Workload** can also be specified. For example, **control_group** is set to **class1:workload1:1**.
- If a database user specifies the Timeshare string (**Rush**, **High**, **Medium**, or **Low**) in the syntax, for example, **control_group** is set to **High**, the resource pool will be associated with the **High** Timeshare Cgroup under **DefaultClass**.
- In multi-tenant scenarios, the Cgroup associated with a group resource pool is a Class Cgroup, and that associated with a service resource pool is a Workload Cgroup. Additionally, switching Cgroups between different resource pools is not allowed.

- **stmt**

  Specifies the maximum number of statements that can be concurrently executed in a resource pool.

  Value range: an integer ranging from –1 to 2147483647. **–1** indicates no restriction and **0** indicates that no statement can be executed.

- **dop**

  Specifies the maximum statement concurrency degree for a resource pool, equivalent to the number of threads that can be created for executing a statement.

  Value range: a positive integer ranging from 1 to 2147483647.

- **memory_size**

  Specifies the maximum memory size of a resource pool.

  Value range: character strings occupying 1 KB to 2047 GB.

- **mem_percent**

  Specifies the proportion of available resource pool memory to the total memory or group user memory.

  – In multi-tenant scenarios, **mem_percent** of group users or service users ranges from 1 to 100. The default value is **20**.

  – In common scenarios, **mem_percent** of common users ranges from 0 to 100. The default value is **0**.

  NOTE

  If both **mem_percent** and **memory_limit** are specified, only **mem_percent** takes effect.

- **io_limits**

  Specifies the upper limit of IOPS in a resource pool. The unit is 10,000.

- **io_priority**

  I/O priority set for jobs that consume many I/O resources. It takes effect when the I/O usage reaches 50%. The options are **Low**, **Medium**, and **High**.

  If you do not want to control I/O resources, use the default value **None**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> The settings of **io_limits** and **io_priority** are valid only for complex jobs, such as batch import (using **INSERT INTO SELECT**, **COPY FROM**, or **CREATE TABLE AS**), complex queries involving over 500 MB data on each DN, and **VACUUM FULL**.

- **max_worker**

  Concurrency in a table during data redistribution. This column is used only for scaling.

- **max_connections**

  Maximum number of connections that can be used by a resource pool.

  > **NOTE**
  >
  > The total number of connections in all resource pools cannot exceed the value of the GUC parameter **max_connections** set for the entire database process.

- **max_dynamic_memory**

  Specifies the maximum dynamic memory that can be used by a resource pool.

- **max_shared_memory**

  Specifies the maximum shared memory that can be used by a resource pool.

- **max_concurrency**

  Specifies the maximum concurrent requests that can be used by a resource pool.

Examples

This example assumes that Cgroups have been created by users in advance. Contact an administrator to create a Cgroup.

```
-- Create a Cgroup (by running commands).
gs_ssh -c "gs_cgroup -c -S class1 -s 40"
gs_ssh -c "gs_cgroup -c -S class1 -G wg1 -g 20"
gs_ssh -c "gs_cgroup -c -S class1 -G wg2 -g 20"

-- Use gsql to log in to the database as an administrator.
-- Create a default resource pool, and associate it with the Medium Timeshare Cgroup under Workload under DefaultClass.
gaussdb=# CREATE RESOURCE POOL pool1;

-- Create a resource pool and specify the High Timeshare Workload Cgroup under the DefaultClass Cgroup.
gaussdb=# CREATE RESOURCE POOL pool2 WITH (CONTROL_GROUP="High");

-- Create a resource pool, and associate it with the Low Timeshare Cgroup under Workload under class1.
gaussdb=# CREATE RESOURCE POOL pool3 WITH (CONTROL_GROUP="class1:Low");

-- Create a resource pool, and associate it with the wg1 Workload Cgroup under class1.
gaussdb=# CREATE RESOURCE POOL pool4 WITH (CONTROL_GROUP="class1:wg1");

-- Create a resource pool, and associate it with the wg2 Workload Cgroup under class1.
gaussdb=# CREATE RESOURCE POOL pool5 WITH (CONTROL_GROUP="class1:wg2:3");

-- Drop resource pools.
gaussdb=# DROP RESOURCE POOL pool1;
gaussdb=# DROP RESOURCE POOL pool2;
gaussdb=# DROP RESOURCE POOL pool3;
gaussdb=# DROP RESOURCE POOL pool4;
gaussdb=# DROP RESOURCE POOL pool5;

-- Drop the Cgroup (by running commands).
gs_ssh -c "gs_cgroup -d -S class1"
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

ALTER RESOURCE POOL and **DROP RESOURCE POOL**

1.13.9.37 CREATE ROLE

Description

Creates a role.

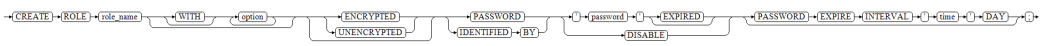
A role is an entity that has objects and permissions in GaussDB. In different environments, a role can represent a user, a group, or both.

Precautions

- By default, new roles added to the database do not have the **LOGIN** attribute. Only roles with the **LOGIN** attribute can log in to the database.
- Only the user who has the CREATE ROLE permission or a system administrator is allowed to create roles.

Syntax

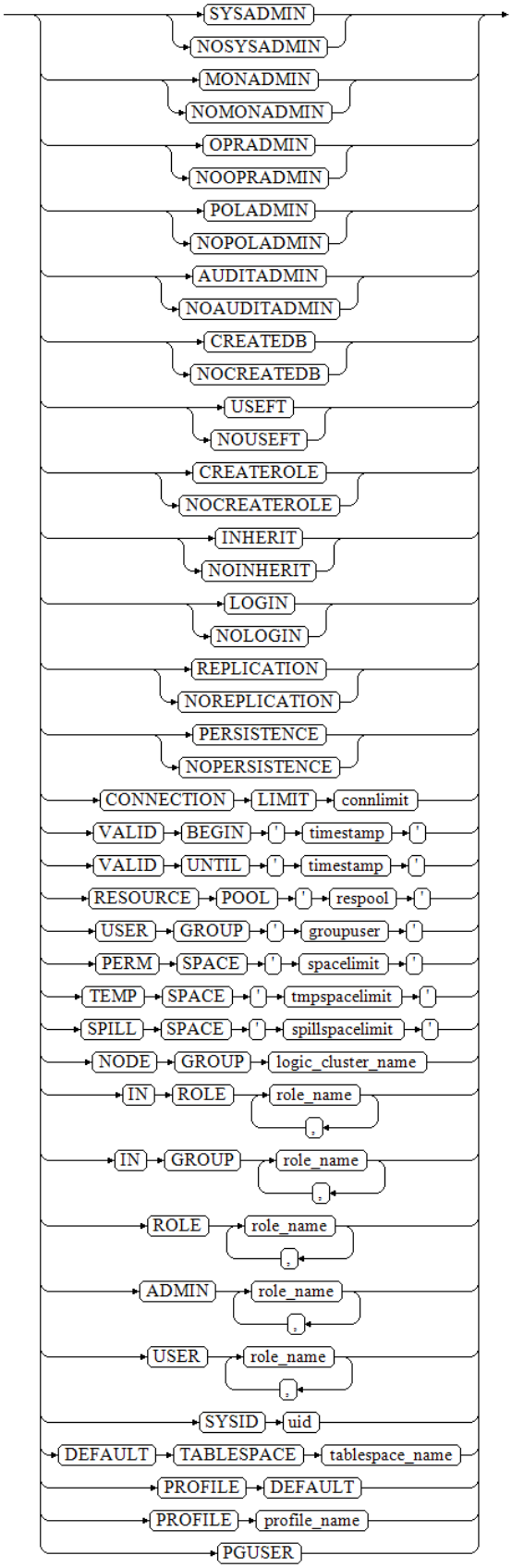
```
CREATE ROLE role_name [ [ WITH ] option [ ... ] ] [ ENCRYPTED | UNENCRYPTED ] { PASSWORD |
IDENTIFIED BY } { 'password' [EXPIRED] | DISABLE }[ PASSWORD EXPIRE INTERVAL 'time' DAY ];
```



The syntax of role information configuration clause **option** is as follows:

```
{
   {SYSADMIN | NOSYSADMIN}
   | {MONADMIN | NOMONADMIN}
   | {OPRADMIN | NOOPRADMIN}
   | {POLADMIN | NOPOLADMIN}
   | {AUDITADMIN | NOAUDITADMIN}
   | {CREATEDB | NOCREATEDB}
   | {USEFT | NOUSEFT}
   | {CREATEROLE | NOCREATEROLE}
   | {INHERIT | NOINHERIT}
   | {LOGIN | NOLOGIN}
   | {REPLICATION | NOREPLICATION}
   | {PERSISTENCE | NOPERSISTENCE}
   | CONNECTION LIMIT connlimit
   | VALID BEGIN 'timestamp'
   | VALID UNTIL 'timestamp'
   | RESOURCE POOL 'respool'
   | USER GROUP 'groupuser'
   | PERM SPACE 'spacelimit'
   | TEMP SPACE 'tmpspacelimit'
   | SPILL SPACE 'spillspacelimit'
   | NODE GROUP logic_cluster_name
   | IN ROLE role_name [, ...]
   | IN GROUP role_name [, ...]
   | ROLE role_name [, ...]
   | ADMIN role_name [, ...]
   | USER role_name [, ...]
   | SYSID uid
   | DEFAULT TABLESPACE tablespace_name
   | PROFILE DEFAULT
   | PROFILE profile_name
   | PGUSER
}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **role_name**

  Name of a role.

  Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

  NOTE

  - If a role name contains more than 63 characters, the database truncates it and retains the first 63 characters as the role name.
  - If a role name contains uppercase letters, the database automatically converts the uppercase letters into lowercase letters. To create a role name that contains uppercase letters, enclose the role name with double quotation marks ("").
  - The role name must start with a letter or underscore (_) and can contain only lowercase letters (a to z), uppercase letters (A to Z), underscores (_), digits (0 to 9), and dollar signs ($).
  - The redisuser is an object used for resource control during redistribution. Currently, a role whose **role_name** is **redisuser** cannot be created.

- **password**

  Login password of a role.

  Value range: a character string that cannot be empty.

  A new password must:

  - Contain at least eight characters. This is the default length.
  - Differ from the role name or the role name spelled backward.
  - Contain at least three types of the following characters: uppercase letters (A to Z), lowercase letters (a to z), digits (0 to 9), and special characters (limited to ~ ! @ # $ % ^ & * ( ) - _ = + \ | [{ }] ; : , < . > / ?).
  - The password can also be a ciphertext character string that meets the format requirements. It is mainly used to import user data. You are advised not to use it directly. The database does not verify the complexity of the ciphertext password. Using the ciphertext password may cause data security risks. If you use the ciphertext password, you need to know the plaintext corresponding to the ciphertext password and ensure that the password meets the complexity requirements.
  - When creating a role, use single or double quotation marks to enclose the user password.

- **EXPIRED**

  You can specify **EXPIRED** to create a role with an expired password.

  After **EXPIRED** is specified, the role cannot perform simple and extended queries. The role can perform statements only after its password is changed.

- **PASSWORD EXPIRE INTERVAL 'time' DAY**

  When creating a role, you can set the password validity period. If the password validity period expires, the role is not allowed to log in to the system. You need to change the password before logging in to the system.

  The value of **time** is an integer ranging from 0 to 999. If the validity period is not specified, the default value **0** is used, indicating that the password is valid permanently.

- **DISABLE**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

By default, a role can change its own password. If **DISABLE** is specified, the role cannot change its own password, that is, disabling password changes for that role.

After **DISABLE** is specified, the password of the role is deleted from the system. The role cannot connect to the database using its own password. The role can connect to the database only through external authentication, such as Kerberos authentication.

Only the administrator can enable or disable the password. Common users cannot disable the password of an initial user. To enable a password, run **ALTER USER** and specify the password.

- **ENCRYPTED | UNENCRYPTED**

  Controls whether the password is stored encrypted in the system catalogs.

  According to product security requirements, passwords must be encrypted for storage. **UNENCRYPTED** cannot be used in GaussDB.

  Since the system cannot decrypt the specified encrypted password string, when the password string is already encrypted using SHA-256, it will be stored in that format regardless of whether **ENCRYPTED** or **UNENCRYPTED** is specified. The encrypted password will be reloaded during the dump/restore operation.

- **SYSADMIN | NOSYSADMIN**

  Specifies whether the role is a system administrator. The default value is **NOSYSADMIN**. After **SYSADMIN** is specified, the role has the highest permission in the system.

  – When separation of duties is disabled, the role with the SYSADMIN permission can create users with the SYSADMIN, REPLICATION, CREATEROLE, AUDITADMIN, MONADMIN, POLADMIN, or CREATEDB permission and common users.

  – When separation of duties is enabled, users with the SYSADMIN permission do not have the permission to create users.

- **MONADMIN | NOMONADMIN**

  Specifies whether the role is a monitor administrator. The default value is **NOMONADMIN**.

- **OPRADMIN | NOOPRADMIN**

  Specifies whether the role is an O&M administrator. The default value is **NOOPRADMIN**.

- **POLADMIN | NOPOLADMIN**

  Specifies whether the role is a security policy administrator. The default value is **NOPOLADMIN**.

- **AUDITADMIN | NOAUDITADMIN**

  Specifies whether a role has the audit and management attributes. The default value is **NOAUDITADMIN**.

- **CREATEDB | NOCREATEDB**

  Specifies whether a role can create databases. The default value is **NOCREATEDB**.

- **USEFT | NOUSEFT**

  This parameter is reserved and not used in this version.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **CREATEROLE | NOCREATEROLE**

  Specifies whether a role can create new roles, that is, whether the role can perform the CREATE ROLE and CREATE USER operations. The default value is **NOCREATEROLE**. After **CREATEROLE** is specified, the role can modify and delete other roles.

  - When separation of duties is disabled, users with the CREATEROLE permission can create users with the CREATEROLE, AUDITADMIN, MONADMIN, POLADMIN, or CREATEDB permission and common users.
  - When separation of duties is enabled, users with the CREATEROLE permission can create users with the CREATEROLE, MONADMIN, POLADMIN, or CREATEDB permission and common users.

- **INHERIT | NOINHERIT**

  Specifies whether a role inherits the permissions of roles in the same group. It is not recommended.

- **LOGIN | NOLOGIN**

  Specifies whether a role can log in to the database. Only roles with the **LOGIN** attribute can log in to the database. The default value is **NOLOGIN**.

  A role having the **LOGIN** attribute can be considered as a user.

- **REPLICATION | NOREPLICATION**

  Specifies whether a role is allowed to initiate streaming replication or put the system in and out of backup mode. A role having the **REPLICATION** attribute is specific to replication.

  The default value is **NOREPLICATION**.

- **PERSISTENCE | NOPERSISTENCE**

  Defines a permanent user. Only the initial user is allowed to create, modify, and delete permanent users with the PERSISTENCE attribute.

- **CONNECTION LIMIT connlimit**

  Specifies how many concurrent connections the role can make.

  Value range: an integer in the range [–1,2^{31} – 1]. The default value is **-1**, indicating that there is no limit.

  NOTICE

  - System administrators are not restricted by this parameter.
  - **connlimit** of each CN is calculated separately. The number of connections in a cluster is calculated using the following formula: Number of connections in a cluster = **connlimit** x Number of normal CNs.

- **VALID BEGIN 'timestamp'**

  Sets the effective time of a role. If the effective time is not specified, the role has no restriction on the valid start time.

  timestamp indicates the effective time in the format of 'YYYY-MM-DD HH:mm:ss'.

- **VALID UNTIL 'timestamp'**

  Sets the timestamp when a role expires. If the expiration time is not specified, there is no restriction on the expiration time of the role.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

timestamp indicates the expiration time in the format of 'YYYY-MM-DD HH:mm:ss'.

- **RESOURCE POOL 'respool'**

  Sets the name of resource pool used by the role. The name belongs to the system catalog pg_resource_pool.

- **USER GROUP 'groupuser'**

  Creates a sub-user.

- **PERM SPACE 'SPACELIMIT'**

  Sets the space available for a role.

- **TEMP SPACE 'tmpspacelimit'**

  Sets the space allocated to the temporary table of a role.

- **SPILL SPACE 'spillspacelimit'**

  Sets the operator disk flushing space of a role.

- **IN ROLE role_name**

  Lists one or more existing roles whose permissions will be inherited by a new role. It is not recommended.

- **IN GROUP role_name**

  IN GROUP is the outdated syntax of IN ROLE. It is not recommended.

- **ROLE role_name**

  Lists one or more existing roles which are automatically added as members of the new role.

- **ADMIN role_name**

  Similar to ROLE. However, ADMIN grants permissions of new roles to other roles.

- **USER role_name**

  Specifies an obsolete spelling of the ROLE clause.

- **SYSID uid**

  The SYSID clause is ignored.

- **DEFAULT TABLESPACE tablespace_name**

  The DEFAULT TABLESPACE clause is ignored.

- **PROFILE profile_name**

  The PROFILE clause is ignored.

- **PGUSER**

  In the current version, this attribute is reserved only for compatibility with earlier versions.

Examples

- Differences between CREATE ROLE and CREATE USER

```
-- Run CREATE ROLE to create role test_role.
gaussdb=# CREATE ROLE test_role PASSWORD '********';
-- Run CREATE USER to create user test_user.
gaussdb=# CREATE USER test_user PASSWORD '********';

-- View the information. Roles created by CREATE ROLE are not allowed to log in to the database by default.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# \du test*
           List of roles
 Role name |  Attributes  | Member of
-----------+--------------+-----------
 test_role | Cannot login | {}
 test_user |              | {}

-- Enable the test_role role to log in to the database.
gaussdb=# ALTER ROLE test_role WITH LOGIN;

gaussdb=# \du test*
           List of roles
 Role name | Attributes | Member of
-----------+------------+-----------
 test_role |            | {}
 test_user |            | {}

-- View schema information. When CREATE USER is executed to create a user, a schema with the
same name is automatically created.
gaussdb=# \dn test*
    List of schemas
   Name    |   Owner
-----------+-----------
 test_user | test_user
(1 row)

-- Drop.
gaussdb=# DROP ROLE test_role;
gaussdb=# DROP USER test_user;
```

- Creating roles with permissions for the separation of duties

```
-- Create user lisa with the OPRADMIN permission and user bob with the SYSADMIN permission.
gaussdb=# CREATE ROLE lisa WITH OPRADMIN PASSWORD '********';
gaussdb=# CREATE ROLE bob WITH SYSADMIN PASSWORD '********';

-- Query the permissions.
gaussdb=# \du lisa;
                  List of roles
 Role name |         Attributes          | Member of
-----------+-----------------------------+-----------
 lisa      | Cannot login, Operatoradmin | {}

gaussdb=# \du bob
                List of roles
 Role name |       Attributes       | Member of
-----------+------------------------+-----------
 bob       | Cannot login, Sysadmin | {}

-- Create role role11 with the CREATEDB permission.
gaussdb=# CREATE ROLE role11 CREATEDB PASSWORD '********';
CREATE ROLE

-- Query the permissions.
gaussdb=# \du role11
                 List of roles
 Role name |       Attributes        | Member of
-----------+-------------------------+-----------
 role11    | Create DB, Cannot login | {}

-- Create role role12 using the ADMIN clause.
gaussdb=# CREATE ROLE role12 WITH  CREATEDB  ADMIN role11  PASSWORD '********';
CREATE ROLE
-- Query the permissions.
gaussdb=# \du role11
                 List of roles
 Role name |       Attributes        | Member of
-----------+-------------------------+-----------
 role11    | Create DB, Cannot login | {role12}

gaussdb=# \du role12
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
                List of roles
 Role name |       Attributes        | Member of
-----------+-------------------------+-----------
 role12    | Create DB, Cannot login | {}

-- Drop roles.
gaussdb=# DROP ROLE role11;
gaussdb=# DROP ROLE role12;
gaussdb=# DROP ROLE lisa;
gaussdb=# DROP ROLE bob;

-- Separation-of-duties configuration: If enableSeparationOfDuty is set to on, the separation of duties is enabled to restrict the permissions of system administrators. System administrators are not allowed to create or modify user configurations.

-- Check whether separation of duties is enabled.
gaussdb=# SHOW enableSeparationOfDuty;
 enableSeparationOfDuty
------------------------
 off
(1 row)
-- Run the following statement to modify parameters:
gs_guc set -Z coordinator -Z datanode -N all -I all -c "enableSeparationOfDuty = on"

-- After the setting is successful, the setting takes effect after a restart.
gs_om -t stop
gs_om -t start

-- Check the separation of duties again.
gaussdb=# SHOW enableSeparationOfDuty;
 enableSeparationOfDuty
------------------------
 on
(1 row)
```

- Creating a role with an expired password

```
-- Create the role test_role2 with an expired password.
gaussdb=# CREATE ROLE test_role2 PASSWORD '********' EXPIRED;
gaussdb=# ALTER ROLE test_role2 WITH LOGIN;

-- test_role2 cannot perform any operation after logging in to the database. You can perform operations only after changing the password as prompted.
gaussdb=# SET ROLE test_role2 PASSWORD '********';
gaussdb=> \d
ERROR:  Please use "ALTER ROLE user_name IDENTIFIED BY 'password' REPLACE 'old password';" to modify the expired password of user test_role2 before operation!

-- Change the password of role test_role2.
gaussdb=> ALTER ROLE test_role2 IDENTIFIED BY '********' REPLACE '********';

-- Drop.
gaussdb=> RESET ROLE;
gaussdb=# DROP ROLE test_role2;
```

- Creating a role and specifying the effective date and expiration date

```
-- Create a role with its validity from January 1, 2015 to January 1, 2026.
gaussdb=# CREATE ROLE test_role3 WITH LOGIN PASSWORD '********' VALID BEGIN '2015-01-01' VALID UNTIL '2026-01-01';
-- Drop.
gaussdb=# DROP ROLE test_role3;
```

- Creating a role and setting the password validity period

```
-- Create a role with a validity period of 30 days.
gaussdb=# CREATE ROLE role4 PASSWORD '********' PASSWORD EXPIRE INTERVAL 30 DAY;
```

Helpful Links

SET ROLE, **ALTER ROLE**, **DROP ROLE**, **GRANT**, and **REVOKE**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.9.38 CREATE ROW LEVEL SECURITY POLICY

Description

Creates a row-level security policy for a table.

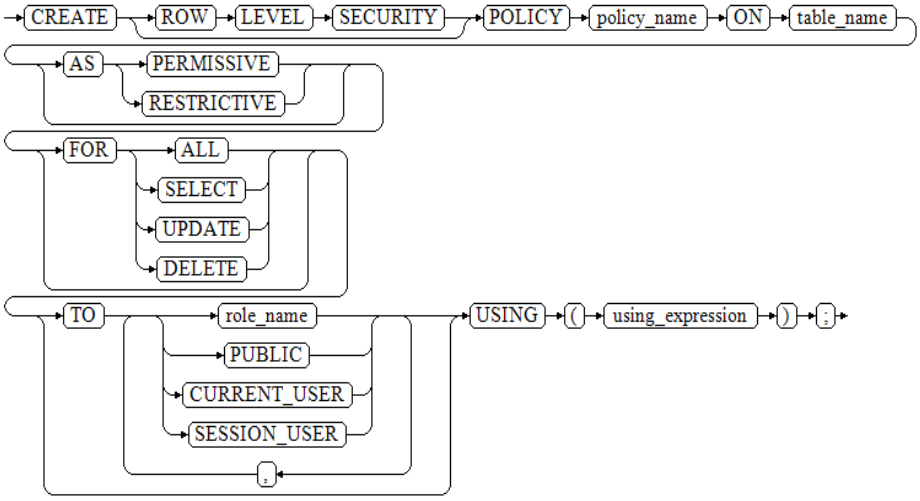
Precautions

- Row-level security policies can be created for row-store tables, row-store partitioned tables, replication tables, unlogged tables, and hash tables.
- Row-level security policies cannot be created for foreign tables and temporary tables.
- Row-level security policies cannot be created for views.
- A maximum of 100 row-level security policies can be defined for a table.
- System administrators are not affected by row-level security policies and can view all data in a table.
- Tables queried by using SQL statements, views, functions, and stored procedures are affected by row-level security policies.
- The data type of a table column to which a row-level security policy has been added cannot be changed.
- The policy takes effect only after row-level security is enabled (by running **ALTER TABLE ... ENABLE ROW LEVEL SECURITY**).
- Currently, row-level security affects the read (SELECT, UPDATE, and DELETE) of data tables and does not affect the write (INSERT and MERGE INTO) of data tables.
- The table owner or system administrators can create an expression in the USING clause. When the client reads the data table, the database server combines the expressions that meet the condition and applies it to the execution plan in the statement rewriting phase of a query. For each tuple in a data table, if the USING expression returns **TRUE**, the tuple is visible to the current user; if the USING expression returns **FALSE** or **NULL**, the tuple is invisible to the current user.
- A row-level security policy name is specific to a table. A data table cannot have row-level security policies with the same name. Different data tables can have the same row-level security policy.

Syntax

```
CREATE [ ROW LEVEL SECURITY ] POLICY policy_name ON table_name
    [ AS { PERMISSIVE | RESTRICTIVE } ]
    [ FOR { ALL | SELECT | UPDATE | DELETE } ]
    [ TO { role_name | PUBLIC | CURRENT_USER | SESSION_USER } [, ...] ]
    USING ( using_expression );
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- **policy_name**

  Specifies the name of a row-level security policy to be created. The names of row-level security policies for a table must be unique.

- **table_name**

  Specifies the name of a table to which a row-level security policy is applied.

- **PERMISSIVE | RESTRICTIVE**
  - **PERMISSIVE** enables the permissive policy for row-level security. The conditions of the permissive policy are joined through the OR expression.
  - **RESTRICTIVE** enables the restrictive policy for row-level security. The conditions of the restrictive policy are joined through the AND expression.

  The join method is as follows:

  ```
  (using_expression_permissive_1 OR using_expression_permissive_2 ...) AND
  (using_expression_restrictive_1 AND using_expression_restrictive_2 ...)
  ```

  If not specified, the default value is **PERMISSIVE**.

- **command**

  Specifies the SQL operations affected by a row-level security policy, including **ALL**, **SELECT**, **UPDATE**, and **DELETE**. If not specified, the default value is **ALL**.
  - **ALL** indicates that all SELECT, UPDATE, and DELETE operations are affected.
  - **SELECT** indicates that SELECT operations are affected by row-level security policies. Only tuple data that meets the condition (the return value of **using_expression** is **true**) can be viewed. The affected operations include SELECT, SELECT FOR UPDATE/SHARE, UPDATE ... RETURNING, and DELETE ... RETURNING.
  - **UPDATE** indicates that UPDATE operations are affected by row-level security policies. Only tuple data that meets the condition (the return value of **using_expression** is **true**) can be updated. The affected operations include UPDATE, UPDATE ... RETURNING, and SELECT ... FOR UPDATE/SHARE.
  - **DELETE** indicates that DELETE operations are affected by row-level security policies. Only tuple data that meets the condition (the return

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

value of **using_expression** is **true**) can be deleted. The affected operations include DELETE and DELETE ... RETURNING.

For the relationship between row-level security policies and SQL statements, see **Table 1-397**.

Table 1-397 Relationship between the ROW LEVEL SECURITY policy and SQL statements

Command	SELECT/ALL Policy	UPDATE/ALL Policy	DELETE/ALL Policy
SELECT	Existing row	No	No
SELECT FOR UPDATE/SHARE	Existing row	Existing row	No
UPDATE	No	Existing row	No
UPDATE RETURNING	Existing row	Existing row	No
DELETE	No	No	Existing row
DELETE RETURNING	Existing row	No	Existing row

- **role_name | PUBLIC | CURRENT_USER | SESSION_USER**

  Specifies database users/roles affected by a row-level security policy. Row-level security policies can be applied to a specified user /role or to all users (**PUBLIC**). The default value **PUBLIC** will be used if you do not specify the user who will be affected.

  - **role_name**: the specified database user.
  - **CURRENT_USER**: username of the current operating environment.
  - **SESSION_USER**: session username.
  - **PUBLIC**: All database users are affected. You can specify multiple affected database users.

  NOTICE

  System administrators are not affected by row-level security.

- **using_expression**

  Specifies an expression defined for a row-level security policy (return type: Boolean).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The expression cannot contain aggregate functions or window functions.
- In the statement rewriting phase of a query, if row-level security for a data table is enabled, the expressions that meet the specified conditions will be added to the plan tree.
- The expression is calculated for each tuple in the data table. For SELECT, UPDATE, and DELETE, row data is visible to the current user only when the return value of the expression is **TRUE**. If the expression returns **FALSE**, the tuple is invisible to the current user. In this case, the user cannot view the tuple through the SELECT statement, update the tuple through the UPDATE statement, or delete the tuple through the DELETE statement.

Examples

```
-- Create user alice.
gaussdb=# CREATE USER alice PASSWORD '*********';

-- Create user bob.
gaussdb=# CREATE USER bob PASSWORD '*********';

-- Create data table all_data.
gaussdb=# CREATE TABLE public.all_data(id int, role varchar(100), data varchar(100));

-- Insert data into the data table.
gaussdb=# INSERT INTO all_data VALUES(1, 'alice', 'alice data');
gaussdb=# INSERT INTO all_data VALUES(2, 'bob', 'bob data');
gaussdb=# INSERT INTO all_data VALUES(3, 'peter', 'peter data');

-- Grant the read permission on the all_data table to users alice and bob.
gaussdb=# GRANT SELECT ON all_data TO alice, bob;

-- Enable the row-level security policy.
gaussdb=# ALTER TABLE all_data ENABLE ROW LEVEL SECURITY;

-- Create a row-level security policy to specify that the current user can view only their own data.
gaussdb=# CREATE ROW LEVEL SECURITY POLICY all_data_rls ON all_data USING(role = CURRENT_USER);

-- View information about the all_data table.
gaussdb=# \d+ all_data
                            Table "public.all_data"
 Column |          Type          | Modifiers | Storage  | Stats target | Description
--------+------------------------+-----------+----------+--------------+-------------
 id     | integer                |           | plain    |              |
 role   | character varying(100) |           | extended |              |
 data   | character varying(100) |           | extended |              |
Row Level Security Policies:
    POLICY "all_data_rls" FOR ALL
      TO public
      USING (((role)::name = "current_user"()))
Has OIDs: no
Distribute By: HASH(id)
Location Nodes: ALL DATANODES
Options: orientation=row, compression=no, storage_type=USTORE, segment=off, enable_rowsecurity=true

-- Run SELECT.
gaussdb=# SELECT * FROM all_data;
 id | role  |    data
----+-------+------------
  1 | alice | alice data
  2 | bob   | bob data
  3 | peter | peter data
(3 rows)

gaussdb=# EXPLAIN(COSTS OFF) SELECT * FROM all_data;
 id |          operation
----+-----------------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 1 | ->  Streaming (type: GATHER)
 2 |    ->  Seq Scan on all_data
(2 rows)

-- Switch to user alice and run SELECT.
gaussdb=# SET ROLE alice PASSWORD '********';
gaussdb=> SELECT * FROM all_data;
 id | role  |    data
----+-------+------------
  1 | alice | alice data
(1 row)

gaussdb=> EXPLAIN(COSTS OFF) SELECT * FROM all_data;
 id |          operation
----+-----------------------------
  1 | ->  Streaming (type: GATHER)
  2 |    ->  Seq Scan on all_data
(2 rows)

        Predicate Information (identified by plan id)
-------------------------------------------------------------------
   2 --Seq Scan on all_data
        Filter: ((role)::name = "current_user"())
 Notice: This query is influenced by row level security feature
(3 rows)

-- Insert data. An error will be reported if the INSERT permission is not granted.
gaussdb=> INSERT INTO all_data VALUES(4, 'test', 'test data');
ERROR: The operation permission on relation all_data is denied.
DETAIL: N/A.

-- Grant the INSERT permission on the all_data table to user alice.
gaussdb=> RESET ROLE;
gaussdb=# GRANT INSERT ON all_data TO alice;

-- Switch to user alice and insert data.
gaussdb=# SET ROLE alice PASSWORD '********';
gaussdb=> INSERT INTO all_data VALUES(4, 'alice', 'test data');
INSERT 0 1

-- Query the all_data table.
gaussdb=> SELECT * FROM all_data;
 id | role  |    data
----+-------+------------
  1 | alice | alice data
  4 | alice | test data
(2 rows)

-- Delete a row-level security policy.
gaussdb=> RESET ROLE;
gaussdb=# DROP ROW LEVEL SECURITY POLICY all_data_rls ON all_data;

-- Delete the all_data table.
gaussdb=# DROP TABLE public.all_data;

-- Delete users alice and bob.
gaussdb=# DROP USER alice, bob;
```

Helpful Links

DROP ROW LEVEL SECURITY POLICY and **ALTER ROW LEVEL SECURITY POLICY**

1.13.9.39 CREATE SCHEMA

Description

Creates a schema.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

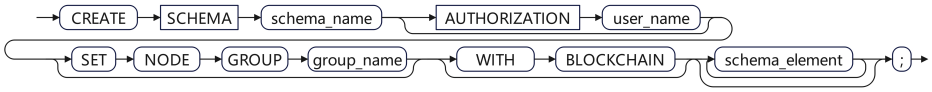
Precautions

- Users with the CREATE permission in the database can create schemas.
- The owner of an object created by a system administrator in a schema with the same name as a common user is the common user, not the system administrator.
- The schema name can be used as a prefix to access named objects. Without this prefix, the system will access named objects within the current schema. When creating a named object, you can also use the schema name as its prefix.
- CREATE SCHEMA can include a subcommand for creating an object in a new schema. Using this subcommand is the same as the executing the subcommand after a schema is created. If the AUTHORIZATION clause is used, all the created objects are owned by this user.

Syntax

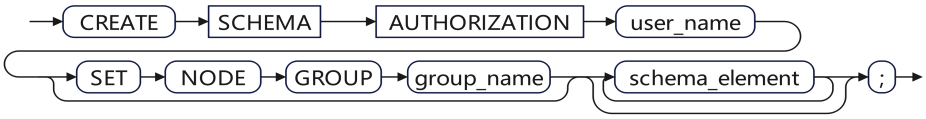
- Create a schema based on a specified schema name.

```
CREATE SCHEMA schema_name
   [ AUTHORIZATION user_name ] [SET NODE GROUP group_name] [WITH BLOCKCHAIN]
[ schema_element [ ... ] ];
```



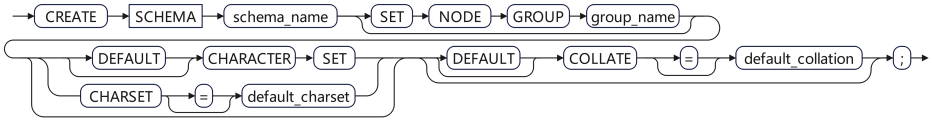
- Create a schema based on a username.

```
CREATE SCHEMA AUTHORIZATION user_name [SET NODE GROUP group_name] [ schema_element
[ ... ] ];
```



- Create a schema and specify the default character set and collation.

```
CREATE SCHEMA schema_name [SET NODE GROUP group_name]
   [ [DEFAULT] CHARACTER SET | CHARSET [ = ] default_charset ] [ [DEFAULT] COLLATE [ = ]
default_collation ];
```



Parameters

- **schema_name**

  Specifies the schema name.

  Value range: character strings complying with **Identifier Naming Conventions**.

  The rules for naming a schema are as follows:

  – The schema name must be unique in the current database.

  – The new name cannot be the same as the initial username of the current database.

  – Do not use a new name starting with pg_, gs_role_, or data_redis.

- **AUTHORIZATION user_name**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the owner of a schema. If **schema_name** is not specified (that is, **creating a schema based on a username**), **user_name** is used as the schema name. In this case, **user_name** can only be a role name.

Value range: an existing username or role name

- **SET NODE GROUP group_name**

  Sets the single-node attribute of a schema. **group_name** indicates the name of a single-node group. Only the single-node group whose type is 's' can be set. The prefix of the group name is **sys_single_group**.

  NOTE

  - This syntax is supported only in M-compatible databases (**sql_compatibility = 'M'**).
  - The CREATE SCHEMA syntax that includes the CREATE TABLE clause (referred to as a composite SQL statement) does not support the single-node table feature. This means it cannot be used to create single-node tables or set the single-node table attribute for schemas.

- **WITH BLOCKCHAIN**

  Specifies the tamper-proof attribute of a schema.

  NOTE

  - To change a normal schema to a tamper-proof schema, set the GUC parameter **enable_ledger** to **on**. The default value is **off**, and the level is SIGHUP.
  - Ordinary row-store tables in the tamper-proof mode are automatically extended to tamper-proof user tables.

- **schema_element**

  Specifies an SQL statement defining an object to be created within the schema.

  Currently, only the CREATE TABLE, CREATE VIEW, CREATE INDEX, CREATE TABLE PARTITION, CREATE SEQUENCE, CREATE TRIGGER, and GRANT clauses are supported.

  NOTE

  - Objects created by **schema_element** are owned by the **user_name** specified by **AUTHORIZATION user_name**.
  - If objects with the same name exist in the schemas in the current search path, you need to specify the schema where the referenced objects are located. You can run **SHOW search_path** to check the schemas on the current search path.

- **default_charset**

  Specifies the default character set of a schema. If this option is not specified, the schema will use the default character set corresponding to the specified collation.

  Value range: For details about the supported character sets, see **Table 1-398**.

  NOTE

  This syntax is supported only when **sql_compatibility** is set to **'MYSQL'** or **'M'**. The preceding information is displayed when **sql_compatibility** is set to **'MYSQL'**. For details about the information when **sql_compatibility** is set to **'M'**, see "SQL Reference > SQL Syntax > SQL Statements > C > CREATE SCHEMA" in *M Compatibility Developer Guide*.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **default_collation**

  Specifies the default collation of a schema. If this option is not specified, the schema will use the default collation corresponding to the specified character set.

  Value range: For details about the supported character sets, see **Table 1-398**.

  > NOTE
  >
  > This syntax is supported only when **sql_compatibility** is set to **'MYSQL'** or **'M'**. The preceding information is displayed when **sql_compatibility** is set to **'MYSQL'**. For details about the information when **sql_compatibility** is set to **'M'**, see "SQL Reference > SQL Syntax > SQL Statements > C > CREATE SCHEMA" in *M Compatibility Developer Guide*.

Examples

```
-- Create the role1 role.
gaussdb=# CREATE ROLE role1 IDENTIFIED BY '********';
CREATE ROLE

-- Create a schema named role1 for the role1 role. The owner of the films and winners tables created by the schema_element subcommand is role1.
gaussdb=# CREATE SCHEMA AUTHORIZATION role1
    CREATE TABLE films (title text, release date, awards text[])
    CREATE VIEW winners AS SELECT title, release FROM films WHERE awards IS NOT NULL;
CREATE SCHEMA

-- Create role2.
gaussdb=# CREATE ROLE role2 IDENTIFIED BY '********';
CREATE ROLE

-- Create schema1 based on the name and specify role2 as the owner.
gaussdb=# CREATE SCHEMA schema1 AUTHORIZATION role2;
CREATE SCHEMA

-- Create schema2 by name and explicitly set the single-node table attribute.
gaussdb=# CREATE SCHEMA schema2 SET NODE GROUP sys_single_group1;
CREATE SCHEMA

-- Modify the tamper-proof attribute.
-- Before modifying the tamper-proof attribute, you need to use gs_guc to set the GUC parameter enable_ledger to on. The default value is off.
gs_guc reload -Z coordinator -Z datanode -N all -I all -c "enable_ledger=on";

-- Check whether enable_ledger is on.
gaussdb=# SHOW enable_ledger;
 enable_ledger
---------------
 on
(1 row)

-- Create a schema and specify the tamper-proof attribute.
gaussdb=# CREATE SCHEMA ledgernsp WITH BLOCKCHAIN;
CREATE SCHEMA

-- Drop schemas.
gaussdb=# DROP SCHEMA role1 CASCADE;
gaussdb=# DROP SCHEMA schema1 CASCADE;
gaussdb=# DROP SCHEMA schema2 CASCADE;
gaussdb=# DROP SCHEMA ledgernsp CASCADE;

-- Drop users.
gaussdb=# DROP USER role1 CASCADE;
gaussdb=# DROP USER role2 CASCADE;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

ALTER SCHEMA and **DROP SCHEMA**

1.13.9.40 CREATE SECURITY LABEL

Description

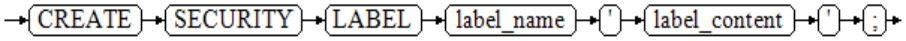
Creates a security label for the specified security policy in the current database.

Precautions

The initial user, users with the SYSADMIN permission, or users who inherit permissions of the built-in role gs_role_seclabel can create security labels.

Syntax

```
CREATE SECURITY LABEL label_name 'label_content';
```



Parameters

- **label_name**

  Security label name, which must be unique in the database.

  Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

  NOTE

  - If the value contains more than 63 characters, the database truncates it and retains the first 63 characters as the security label name.
  - If a security label name contains uppercase letters, the database automatically converts the uppercase letters into lowercase letters. To create a resource label name that contains uppercase letters, enclose the security label name with double quotation marks ("").
  - The security label name must start with a letter or underscore (_) and can contain only lowercase letters (a to z), uppercase letters (A to Z), underscores (_), digits (0 to 9), and dollar signs ($).

- **label_content**

  Security label content. The requirements are as follows:

  A security label consists of a level and a range, which are separated by a colon (:). The format is "level:range" with only one level and at least one range, for example, "L1:G2,G41,G6-G27".

  - There are 1024 levels named **L***i*, where 1 ≤ *i* ≤ 1024. The levels meet a partial order relationship (if *i* ≤ *j*, then **L***i* ≤ **L***j*). For example, L1 is lower than L3.
  - There are 1024 ranges named **G***i*, where 1 ≤ *i* ≤ 1024. Ranges cannot be compared, but set operations can be performed. Multiple ranges are separated by commas (,). A hyphen (-) indicates an interval. For example, {G2-G5} indicates {G2, G3, G4, G5}, and {G1} is a subset of the set {G1, G6}.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - The letters L and G must be capitalized and followed by at least one non-zero digit. Other characters are not allowed. In the {Gxxx-Gyyy} format, yyy must be greater than or equal to xxx.
  - If the input levels and ranges do not meet the requirements, the system reports an error.

```
-- Example
gaussdb=# CREATE SECURITY LABEL sec_label3 'L3:'; // The label must contain at least one content range category.
ERROR:  in label text "L3:", there at least have one level and one group
```

Examples

```
-- Create a security label sec_label.
gaussdb=# CREATE SECURITY LABEL sec_label 'L1:G4';

-- Create security label sec_label with the content of 'L1:G2,G4'.
gaussdb=# CREATE SECURITY LABEL sec_label 'L1:G2,G4';
ERROR:  security label "sec_label" already exists
-- Create security label sec_label1 with the content of 'L1:G2,G4'.
gaussdb=# CREATE SECURITY LABEL sec_label1 'L1:G2,G4';

-- Create security label sec_label2 with the content of 'L3:G1-G5'.
gaussdb=# CREATE SECURITY LABEL sec_label2 'L3:G1-G5';

-- View the security labels created in the system.
gaussdb=# SELECT * FROM gs_security_label;
 label_name | label_content
------------+---------------
 sec_label  | L1:G4
 sec_label1 | L1:G2,G4
 sec_label2 | L3:G1-G5
(3 rows)

-- Delete the existing security labels sec_label, sec_label1, and sec_label2.
gaussdb=# DROP SECURITY LABEL sec_label;
gaussdb=# DROP SECURITY LABEL sec_label1;
gaussdb=# DROP SECURITY LABEL sec_label2;

-- View the security labels created in the system again.
gaussdb=# SELECT * FROM gs_security_label;
 label_name | label_content
------------+---------------
(0 rows)
```

Helpful Links

DROP SECURITY LABEL and **SECURITY LABEL ON**

1.13.9.41 CREATE SEQUENCE

Description

Adds a sequence to the current database. A sequence is a special table that stores arithmetic progressions. It does not store regular data and is usually used to generate unique identifiers for rows or tables.

Precautions

- You can create sequences in the **PUBLIC** or user schemas only when you have the CREATE ANY SEQUENCE permission. The user who creates a sequence becomes its owner.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- After a sequence is created, when you use the nextval() and generate_series(1,*N*) functions of the sequence to insert data into a table, ensure that the number of times the nextval() function can be called is greater than or equal to *N* + 1. Otherwise, an error will be reported because the generate_series() function calls the nextval() function for *N* + 1 times.
- Sequences cannot be associated with the **rowid** and **rowno** system columns.

Syntax

```
CREATE [LARGE] SEQUENCE [ IF NOT EXISTS ] name [ INCREMENT [ BY ] increment ]
    [ MINVALUE minvalue | NO MINVALUE | NOMINVALUE ] [ MAXVALUE maxvalue | NO MAXVALUE |
NOMAXVALUE]
    [ START [ WITH ] start ] [ CACHE cache | NOCACHE ] [ [ NO ] CYCLE | NOCYCLE ] [ RESTART [ WITH
start ] ]
    [ OWNED BY { table_name.column_name | NONE } ];
```

Parameters

- **LARGE**

  If **LARGE** is specified, a 128-bit int sequence is created. If **LARGE** is not specified, a 64-bit int sequence is created by default.

  By default, the maximum value of **SEQUENCE** is $2^{63} - 1$. If a LARGE identifier is used, the maximum value can be $2^{127} - 1$.

- **IF NOT EXISTS**

  When the **IF NOT EXISTS** keyword is specified, the system checks the current schema for any sequence with the same name before you create a sequence. If it finds such a sequence, the sequence creation process will be halted, and a NOTICE message will be returned.

  When **IF NOT EXISTS** is not specified and a sequence with the same name exists in the schema, an ERROR is returned.

- **name**

  Specifies the name of the sequence to be created. The name can be schema-qualified. If a schema is qualified, the sequence is created in the specified schema; otherwise, it is created in the current schema. In the same schema, the sequence name must be unique and cannot be the same as that of other sequences, tables, indexes, views, or foreign tables in the schema.

  Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

- **increment**

  (Optional) Step of the sequence.

  – Positive number: An ascending sequence is created.

  – Negative number: A descending sequence is created.

  The default value is **1**.

  NOTE

  In MySQL-compatible mode, any floating-point number used as a step will be automatically converted to an integer. In other modes, this parameter cannot be set to a floating-point number.

- **MINVALUE minvalue | NO MINVALUE| NOMINVALUE**

  (Optional) Determines the minimum value that the sequence can generate. **NOMINVALUE** is equivalent to **NO MINVALUE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

If **minvalue** is not declared or **NO MINVALUE** is declared:

- If the keyword LARGE is not declared, the default value of an ascending sequence is **1**, and that of a descending sequence is **$-2^{63} + 1$**.
- If the keyword LARGE is declared, the default value of an ascending sequence is **1**, and that of a descending sequence is **-2^{127}**.

- **MAXVALUE maxvalue | NO MAXVALUE| NOMAXVALUE**

  (Optional) Determines the maximum value that the sequence can generate. **NOMAXVALUE** is equivalent to **NO MAXVALUE**.

  If **maxvalue** is not declared or **NO MAXVALUE** is declared:

  - If the keyword LARGE is not declared, the default value of an ascending sequence is **$2^{63} – 1$**, and that of a descending sequence is **-1**.
  - If the keyword LARGE is declared, the default value of an ascending sequence is **$2^{127} – 1$**, and that of a descending sequence is **-1**.

  NOTE

  When a sequence is bound to a table column, if the sequence value exceeds the maximum value of the column type, an error will be reported during data insertion. Therefore, the column type must be able to accommodate the maximum possible value of the sequence to prevent service exceptions.

- **START [WITH] start**

  (Optional) Specifies the starting value for the sequence.

  Default value: **minvalue** for an ascending sequence and **maxvalue** for a descending sequence.

- **CACHE cache | NOCACHE**

  (Optional) Specifies the number of sequences stored in the memory for quick access purposes. **NOCACHE** is equivalent to not specifying **CACHE**.

  Default value: **1** (only one value can be generated at a time, i.e., no cache).

  NOTE

  - It is not recommended that you define **cache** and **maxvalue** or **minvalue** at the same time. Doing so may disrupt the sequence's continuity and result in unused sequence numbers. If you have specific requirements on concurrency performance, set the GUC parameter **session_sequence_cache**.
  - **cache** determines the value that a single CN/DN can request from the GTM at a time. **session_sequence_cache** determines the value that a single session can request from a CN/DN at a time. This value will be automatically discarded once the session ends.
  - The larger the cache, the longer the time required for executing nextval () for the first time.

- **[NO] CYCLE | NOCYCLE**

  (Optional) Allows the sequence to wrap around when **maxvalue** or **minvalue** has been reached.

  If **NO CYCLE** is specified, any calls to nextval after the sequence has reached its maximum value will return an error.

  - **NOCYCLE** is equivalent to **NO CYCLE**. If the cycling option is not specified, the default value is **NO CYCLE**.
  - If **CYCLE** is specified, the sequence uniqueness cannot be ensured.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **RESTART [WITH start]**

  (Optional) Specifies the starting value for the sequence again.

  If **start** is not specified, the default value is **minvalue** for an ascending sequence and **maxvalue** for a descending sequence.

- **OWNED BY**

  (Optional) Joins a sequence with a specified column included in a table. In this way, the sequence will be deleted when you delete its joined column or the table where the column belongs to. The joined table and sequence must be owned by the same user and in the same schema. Note that OWNED BY only joins a table column with the sequence and does not automatically generate an auto-increment sequence when data is inserted.

  OWNED BY NONE, the default, specifies that there is no such join.

NOTICE

You are advised not to use the sequence created using OWNED BY in other tables. If multiple tables need to share a sequence, the sequence must not belong to a specific table.

Examples

- Create an ascending sequence named **seq1**. The sequence starts from 101 and the step is 10.

```
gaussdb=# CREATE SEQUENCE seq1
    START 101
    INCREMENT 10;

-- Select the next number from the sequence.
gaussdb=# SELECT nextval('seq1');
 nextval
---------
     101
(1 row)
gaussdb=# SELECT nextval('seq1');
 nextval
---------
     111

-- Delete the sequence.
gaussdb=# DROP SEQUENCE seq1;
```

- Implement the auto-increment column.

```
-- Create a table.
gaussdb=# CREATE TABLE test1(id int PRIMARY KEY, name varchar(20));

-- Create a sequence joined with the table.
gaussdb=# CREATE SEQUENCE test_seq2
    START 1
    NO CYCLE
    OWNED BY test1.id;

-- Set the default value of a column.
gaussdb=# ALTER TABLE test1 ALTER COLUMN id SET DEFAULT nextval('test_seq2'::regclass);

-- Insert data.
gaussdb=# INSERT INTO test1 (name) values ('Joe'),('Scott'),('Ben');

-- Query.
gaussdb=# SELECT * FROM test1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 id | name
----+-------
  3 | Ben
  1 | Joe
  2 | Scott
(3 rows)

-- Delete the sequence and the table.
gaussdb=# DROP SEQUENCE test_seq2 CASCADE;
gaussdb=# DROP TABLE test1;
```

- Create a cycle sequence.

```
-- Create an ascending sequence seq_test1. The maximum value is 3 and the sequence can be cycled.
gaussdb=# CREATE SEQUENCE seq_test1
    START 1
    MAXVALUE 3
    CYCLE;

-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE tbl_test1(
    c1 int DEFAULT nextval('seq_test1'),
    c2 varchar
);
gaussdb=# INSERT INTO tbl_test1 (c2) values ('test');
gaussdb=# INSERT INTO tbl_test1 (c2) values ('test');
gaussdb=# INSERT INTO tbl_test1 (c2) values ('test');
gaussdb=# INSERT INTO tbl_test1 (c2) values ('test');
gaussdb=# INSERT INTO tbl_test1 (c2) values ('test');

-- Query table data. When the value of ascending sequence seq_test reaches the maximum value, the value starts from the minimum value again.
gaussdb=# SELECT ctid,* FROM tbl_test1 ORDER BY 1,2;
 ctid  | c1 |  c2
-------+----+------
 (0,1) |  1 | test
 (0,1) |  3 | test
 (0,2) |  2 | test
 (0,3) |  1 | test
 (0,4) |  2 | test
(5 rows)

-- Delete the table and sequence.
gaussdb=# DROP TABLE tbl_test1;
gaussdb=# DROP SEQUENCE seq_test1;
```

Helpful Links

DROP SEQUENCE and **ALTER SEQUENCE**

1.13.9.42 CREATE SERVER

Description

Creates a foreign server. A foreign server stores other homogeneous cluster information.

Precautions

- Only a system administrator and users with permission to use a specified FOREIGN DATA WRAPPER can create a foreign server. The authorization syntax is as follows:

  ```
  GRANT USAGE ON FOREIGN DATA WRAPPER fdw_name TO username
  ```

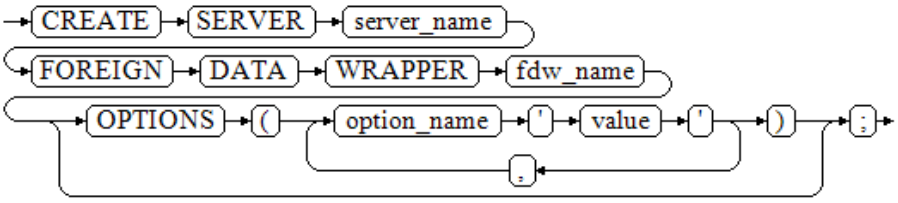
  fdw_name is the name of the FOREIGN DATA WRAPPER, and **username** is the name of the user creating a foreign server.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- When multi-layer quotation marks are used for sensitive columns (such as **password**) in **OPTIONS**, the semantics is different from that in the scenario where quotation marks are not used. Therefore, sensitive columns are not identified for anonymization.

Syntax

```
CREATE SERVER server_name
   FOREIGN DATA WRAPPER fdw_name
   [ OPTIONS ( { option_name ' value ' } [, ...] ) ];
```



Parameters

- **server_name**

  Name of the foreign server to be created.

  Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

- **FOREIGN DATA WRAPPER fdw_name**

  Specifies the name of the foreign data wrapper.

  Value range: **fdw_name** is the data wrapper created by the system during database initialization. Currently, **fdw_name** can only be **gc_fdw** for other homogeneous clusters. You can also create **dist_fdw**, **file_fdw**, and **log_fdw**. **log_fdw** is used only for syntax compatibility and can be used to create foreign tables, but it is meaningless. **dist_fdw** is used to import GDS data. You do not need to manually create a server using **dist_fdw** because **gsmpp_server** is already built in.

- **OPTIONS ({ option_name ' value ' } [, ...])**

  Specifies the parameters of a foreign server.

  – address

    Address of the foreign server.

  – dbname

    Database name of the foreign server.

  – username

    Username of a foreign server.

  – password

    Password of the foreign server.

Examples

Create a foreign server, in which **file_fdw** is the FOREIGN DATA WRAPPER existing in the database.

```
-- Create a foreign server.
gaussdb=# CREATE SERVER my_server FOREIGN DATA WRAPPER file_fdw;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Delete the foreign server.
gaussdb=# DROP SERVER my_server;
```

Create another foreign server in the homogeneous cluster, where **gc_fdw** is the foreign data wrapper in the database.

```
-- Create a foreign server.
gaussdb=# CREATE SERVER server_remote FOREIGN DATA WRAPPER GC_FDW OPTIONS
    (address '10.146.187.231:8000,10.180.157.130:8000' ,
  dbname 'test',
  username 'test',
  password '********'
);

-- Delete the foreign server.
gaussdb=# DROP SERVER server_remote;
```

Helpful Links

ALTER SERVER and **DROP SERVER**

1.13.9.43 CREATE SYNONYM

Description

Creates a synonym object. A synonym is an alias of a database object and is used to record the mapping between database object names. You can use synonyms to access associated database objects. To use a synonym, you must have the required permissions on the associated object. Currently, synonyms can be used to access the following database objects: tables, views, types, packages, functions, stored procedures, sequences, and other synonym objects.

Precautions

- Users granted the CREATE ANY SYNONYM permission can create synonyms in user schemas.
- The following DML statements support synonyms: SELECT, INSERT, UPDATE, DELETE, EXPLAIN, and CALL.
- DDL statements (CREATE, DROP, and ALTER) cannot be executed on associated objects through synonyms.
- Do not create synonyms in stored procedures or functions. Synonyms must be created in the pg_synonym system catalog in advance, so that they can be referenced in a stored procedure.
- To prevent sensitive information leakage, you are not allowed to create synonyms for functions that contain sensitive information such as passwords (for example, encryption and decryption functions gs_encrypt, gs_decrypt, gs_encrypt_bytea, and gs_decrypt_bytea) or call such functions through synonyms.
- You are advised not to create synonyms for temporary tables. To create a synonym, you need to specify the schema name of the target temporary table. Otherwise, the synonym cannot be used normally. In addition, you need to run the **DROP SYNONYM** command before the current session ends.
- Synonyms cannot be created for encrypted tables that contain encrypted columns and views, functions, and stored procedures based on encrypted tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- You cannot use meta-commands such as \d, \df, and \sf to access information about associated objects through synonyms.
- Synonyms can be nested. When a synonym is used, if the synonym points to another synonym, the system traces the synonym layer by layer until the final associated object is found.
- The synonym search and parsing rules are as follows:
  - If the **search_path** is set and the schema of the synonym is not specified, the parsing sequence is as follows:

    Stored procedures and functions: The pg_proc system catalog is preferentially searched by name. If no function with the same name is found, synonyms are searched. Finally, the search is performed on the **search_pat**h.

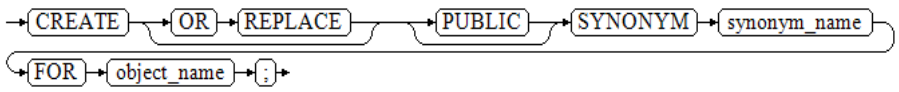
    Other objects (such as tables and views): The search is performed on the **search_path** first. In the same schema, the priority of objects themselves is higher than that of synonyms.
  - If the search statement specifies both an object and its schema, for example, **schema1.object1**:

    The system checks whether the object **schema1.object1** exists. If the object does not exist, it checks whether the synonym **schema1.object1** exists. If the synonym **schema1.object1** exists, the object **schema2.object2** associated with the synonym is searched. If neither **schema2.object2** nor the synonym **schema2.object2** exists and **schema2** is a username, the public synonym **object2** is searched.
  - If the search SQL statement does not give the schema of the object, for example, **object1**:

    The system traverses **search_path** to search for object **schema.object1** in the specified schema. If object **schema.object1** does not exist, the system searches for the synonym **schema.object1**. If the synonym **schema.object1** does not exist and the schema to be searched is a username, the public synonym **object1** is searched.

Syntax

```
CREATE [ OR REPLACE ] [PUBLIC] SYNONYM synonym_name
    FOR object_name;
```



Parameters

- **OR REPLACE**

  (Optional) Redefines the synonym if it already exists.
- **PUBLIC**

  (Optional) Creates a PUBLIC synonym.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- PUBLIC synonyms must be unique in the same database.
- If the database is upgraded from a version that does not support PUBLIC synonyms to a version that supports PUBLIC synonyms, PUBLIC synonyms cannot be created or deleted before the upgrade is committed.
- All users can access PUBLIC synonyms. Except the initial user and system administrators, you must have the CREATE PUBLIC SYNONYM and DROP PUBLIC SYNONYM permissions to create and drop PUBLIC synonyms, respectively.
- For the PUBLIC synonym, the values of **synnamespace** and **synowner** in the pg_synonym system catalog are **0**, the value of **owner** in the ADM_SYNONYMS and DB_SYNONYMS system views is **PUBLIC**, and the value of **schema_name** is **NULL**.
- If no synonym schema is specified, the system searches for objects with the same name to determine whether the objects exist. Then, the system searches for synonyms by **search_path**. Finally, the system searches for PUBLIC synonyms. If a synonym schema is specified, PUBLIC synonyms are not retrieved.

- **synonym_name**

  Name (optionally schema-qualified) of the synonym to be created.

  Value range: character strings complying with **Identifier Naming Conventions**.

  NOTE

  If a schema is specified, the synonym is created in the specified schema. Otherwise, the synonym is created in the current schema. If the schema of a synonym is the schema to which the user belongs, the owner of the synonym is also the owner of the schema. In other scenarios, the owner of the synonym is the creator of the synonym by default.

- **object_name**

  Name (optionally schema-qualified) of the associated object.

  Value range: character strings complying with **Identifier Naming Conventions**.

  NOTE

  - **object_name** can be the name of an object that does not exist.
  - **object_name** can be the name of a PACKAGE object, but cannot be associated with functions in the PACKAGE. You can use PACKAGE synonyms to access functions and variables in the PACKAGE.
  - **object_name** can be the name of a remote object accessed by using a database link. For details about database links, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.
  - After the original object is deleted, the synonym associated with it will not be deleted cascadingly. If you continue to access the synonym, an error will be reported. If the original object is a table, a message is displayed indicating that the table has expired. If the original object is a function, stored procedure, or PACKAGE, a message is displayed indicating that the object does not exist.

Examples

```
-- Create schema ot.
gaussdb=# CREATE SCHEMA ot;

-- Create a table ot.test_tbl1.
gaussdb=# CREATE TABLE ot.test_tbl1(c1 INT, c2 INT);
gaussdb=# INSERT INTO ot.test_tbl1 values(1,1);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- View the current value of search_path.
gaussdb=# SHOW search_path;
  search_path
----------------
 "$user",public
(1 row)

-- The current value of search_path does not contain ot, and the current user is not ot. Therefore, an error is reported when you directly view the table name.
gaussdb=# SELECT * FROM test_tbl1;
ERROR:  relation "test_tbl1" does not exist
LINE 1: SELECT * FROM test_tbl1;
                      ^
-- Create a synonym.
gaussdb=# CREATE OR REPLACE SYNONYM test_tbl1 FOR ot.test_tbl1;

-- Use the synonym.
gaussdb=# SELECT * FROM test_tbl1;
 c1 | c2
----+----
  1 |  1
(1 row)
gaussdb=# INSERT INTO test_tbl1 VALUES (2,2);

-- Query a created synonym.
gaussdb=# SELECT synname,synobjschema,synobjname FROM pg_synonym WHERE synname = 'test_tbl1';
  synname  | synobjschema | synobjname
-----------+--------------+------------
 test_tbl1 | ot           | test_tbl1
(1 row)

-- Drop.
gaussdb=# DROP SYNONYM test_tbl1;
gaussdb=# DROP TABLE ot.test_tbl1;
gaussdb=# DROP SCHEMA ot CASCADE;
```

Helpful Links

ALTER SYNONYM and **DROP SYNONYM**

1.13.9.44 CREATE TABLE

Description

Creates an initially empty table in the current database. The table will be owned by the user executing the statement. Row-store tables are created by default.

Precautions

- If a PRIMARY KEY or UNIQUE constraint is added to a non-distribution key of a hash distributed table, a GSI is created by default. For details about the constraints on GSIs, see **Limitations**. For a non-hash distributed table, the PRIMARY KEY or UNIQUE constraint must contain distribution keys. Otherwise, an error is reported.
- In distributed mode, **HASH(column_name)** is used by default, where **column_name** can be the primary-key column of the table, a column with a UNIQUE constraint (if any), or the first column of a data type that can be used as the distribution key. The order of priority is: Primary-key column > Column with a UNIQUE constraint > First column of a data type that can be used as the distribution key. If the table contains a primary-key column and

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

multiple columns with UNIQUE constraints, and the distribution mode is not specified, GSIs will be created for the other columns with UNIQUE constraints once the primary-key column has been determined based on priority.

NOTICE

If a distribution key of the character type is used in MySQL-compatible mode (**sql_compatibility = 'MYSQL'**), avoid using case-insensitive collations (such as **utf8mb4_general_ci**, **utf8mb4_unicode_ci**, and **utf8mb4_0900_ai_ci**) because these collations are prone to data skew and cannot guarantee even distribution. It is advisable to use the **utf8mb4_bin** collation instead.

- Distribution keys can be updated only when the table attribute **enable_update_distkey** is set to **on**, and the constraints outlined below are met.

  NOTE

  - Distribution keys can be updated only when **enable_update_distkey** is set to **on**.
  - The UPDATE statement cannot be pushed down to DNs for execution. Instead, a PGXC plan is directly generated, and it remains unchanged regardless of the values before and after the distribution key updates.
  - Tables with row-level UPDATE TRIGGER are not supported. Any attempt to use them will result in an error.
  - Row-level INSERT/DELETE TRIGGER is not operational, but statement-level UPDATE TRIGGER can be executed without issue.
  - Concurrent updates on the same row are not supported. To proceed, you must acquire a lock first. The result of acquiring a lock on DNs, whether it returns **0** or reports an error, is determined by the setting of the GUC parameter **concurrent_dml_mode**. If an error is reported, the following scenarios may occur:
    - The error message is "update distribute column conflict."
    - The error message indicates lock timeout if the time taken to acquire the lock exceeds the threshold.
  - Tables with GSIs are not supported. Any attempt to use them will result in an error.
  - List distribution and range distribution are not supported. Any attempt to use them will result in an error. Only hash distribution is allowed.
  - MERGE INTO and UPSERT cannot be used to update distribution keys. Any attempt to use them will result in an error.
  - **gtm_free** is not supported. Any attempt to use it will result in an error.
  - UPDATE RETURNING is not supported. Any attempt to use it will result in an error.
  - JOIN is not supported. Any attempt to use it will result in an error.
  - UPDATE LIMIT is not supported. Any attempt to use it will result in an error.
- If a fault occurs during table creation, the system may be unable to automatically clear the created disk files and disk files with a size of 0, even after the fault has been rectified.
- When JDBC is used, the **DEFAULT** value can be set through PreparedStatement.
- Row-store tables do not support using foreign keys as table-level constraints.
- Foreign key constraints cannot be created on the **rowid** or **rowno** system columns.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Users with the CREATE ANY TABLE permission can create tables in the schemas of user **PUBLIC** and other common users. To create a table that contains serial columns, you must also obtain the CREATE ANY SEQUENCE permission.
- Data of the XML type cannot be used as a primary key or foreign key.
- If you add or modify an ILM policy for a data object and include a row-level expression, you must use only the whitelisted functions (in **Row Expression Function Whitelist**) as the expression.

NOTICE

- If the database creates unlimited tables, CNs may be affected as follows:
  - Resource exhaustion: Each table occupies certain disk space. Unlimited table creation will occupy a large amount of memory and disk space, which may exhaust CN resources. As a result, the system breaks down or becomes unstable.
    - Performance deterioration: Unlimited table creation causes a large number of I/O operations and CPU computing, and the metadata information of the database becomes large, which may deteriorate the CN performance, including operations such as insert, query, update, and deletion. As a result, the system responds slowly or cannot meet service requirements.
    - Security issues: Excessive tables make database management and maintenance difficult. Creating unlimited tables may cause security issues such as data leakage or data loss. Database stability decreases, causing immeasurable loss to enterprises.

  Therefore, plan the number and size of tables in the database properly to avoid unlimited table creation and ensure system stability, reliability, and security.
- Among the functions supported by the row-level expression ON (EXPR) in the ILM policy, the output of some may be influenced by compatibility parameters. For instance, after you set **b_format_version** to **'5.7'** and **b_format_dev_version** to **'s2'** in MySQL-compatible mode, the upper function will be unable to convert data to uppercase as usual.

- The number of table constraints cannot exceed 32,767.

Syntax

Create a table.

```
CREATE [ [ GLOBAL | LOCAL ] { TEMPORARY | TEMP } | UNLOGGED ] TABLE [ IF NOT EXISTS ] table_name
    { ( { column_name data_type [ CHARACTER SET | CHARSET charset ] [ COLLATE collation ]
[ column_constraint [ ... ] ]
        | table_constraint
        | LIKE source_table [ like_option [...] ] }
        [, ... ] )
    | LIKE source_table }
    [ table_option [ ,... ] [ ... ] ]
    [ htap_option ]
    [ WITH ( {storage_parameter = value} [, ... ] ) ]
    [ WITHOUT OIDS ]
    [ ON COMMIT { PRESERVE ROWS | DELETE ROWS } ]
    [ ILM ADD POLICY ROW STORE COMPRESS
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | days | month | months | year | years } OF NO
MODIFICATION [ ON ( EXPR )]
 | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour| hours | day | days | month | months | year |
years } OF NO MODIFICATION
 | NONE}]
   [ TABLESPACE tablespace_name ]
   [ DISTRIBUTE BY { REPLICATION | HASH ( column_name [, ...] )
   | MURMURHASH ( expression )
   | RANGE ( column_name [, ...] ) { SLICE REFERENCES tablename | ( slice_less_than_item [, ...] )
   | ( slice_start_end_item [, ...] ) }
   | LIST ( column_name [, ...] ) { SLICE REFERENCES tablename | ( slice_values_item [, ...] ) }
   } ]
   [ TO { GROUP groupname | NODE ( nodename [, ... ] ) } ];
```

- **table_option**:

  ```
  { COMMENT [ = ] 'string' |
    AUTO_INCREMENT [ = ] value |
    [ DEFAULT ] CHARACTER SET | CHARSET [ = ] default_charset |
    [ DEFAULT ] COLLATE [ = ] default_collation |
    ENGINE [ = ] { InnoDB | 'InnoDB' | "InnoDB" } }
  ```

- **htap_option**:

  ```
  { COLVIEW [ PRIORITY { HIGH | LOW | NONE } ] |
    NOCOLVIEW }
  ```

- **column_constraint**:

  ```
  [ CONSTRAINT constraint_name ]
  { NOT NULL |
    NULL |
    CHECK ( expression ) |
    DEFAULT default_expr |
    ON UPDATE update_expr |
    AUTO_INCREMENT |
    COMMENT 'string' |
    COLVIEW |
    NOCOLVIEW |
    UNIQUE [KEY] index_parameters |
    PRIMARY KEY index_parameters |
    ENCRYPTED WITH ( [DATATYPE_CL = client_logic_type,]COLUMN_ENCRYPTION_KEY =
  column_encryption_key, ENCRYPTION_TYPE = encryption_type_value ) |
    REFERENCES reftable [ ( refcolumn ) ] [ MATCH FULL | MATCH PARTIAL | MATCH SIMPLE ]
       [ ON DELETE action ] [ ON UPDATE action ] }
  [ DEFERRABLE | NOT DEFERRABLE ] [ INITIALLY DEFERRED | INITIALLY IMMEDIATE ]
  ```

- **table_constraint**:

  ```
  [ CONSTRAINT [ constraint_name ] ]
  { CHECK ( expression ) |
    UNIQUE [ index_name ] [ USING method ] ( { { column_name [ ( length ) ] | ( expression ) } [ ASC |
  DESC ] }[, ... ] ) index_parameters |
    PRIMARY KEY [ USING method ] ( { column_name [ ASC | DESC ] } [, ... ] ) index_parameters
  [ DEFERRABLE | NOT DEFERRABLE ] [ INITIALLY DEFERRED | INITIALLY IMMEDIATE ]
  { [ COMMENT 'string' ] [ ... ] }
  ```

- **like_option**:

  ```
  { INCLUDING | EXCLUDING } { DEFAULTS | CONSTRAINTS | INDEXES | STORAGE | COMMENTS |
  PARTITION | RELOPTIONS | DISTRIBUTION | UPDATE | ILM | ALL }
  ```

- **index_parameters**:

  ```
  [ WITH ( {storage_parameter = value} [, ... ] ) ]
  [ USING INDEX TABLESPACE tablespace_name ]
  ```

- Range distribution rules:

  - **slice_less_than_item**:

    ```
    SLICE name VALUES LESS THAN ({ expression | MAXVALUE } [, ...]) [ DATANODE
    datanode_name | ( datanode_name_list [, ... ] ) ]
    ```

  - **slice_start_end_item**:

    ```
    SLICE name {
    { START ( expression ) END ( expression ) EVERY ( expression ) } |
    { START ( literal ) END ( { literal | MAXVALUE } ) } |
    { START ( literal ) } |
    ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    { END ( { literal | MAXVALUE } ) }
    }
```

- List distribution rules:

  slice_values_item:

```
[ ( SLICE name VALUES (expression [, ... ]) [DATANODE datanode_name | ( datanode_name_list )]
 [, ... ] ) |
  ( SLICE name VALUES (DEFAULT) [DATANODE datanode_name] | ( datanode_name_list ) )
]
```

- **update_expr**:

```
{ CURRENT_TIMESTAMP | LOCALTIMESTAMP | NOW() }
```

Parameters

- **UNLOGGED**

  Specifies whether to create the table as an unlogged table. Data written to unlogged tables is not written to the WALs, which makes them considerably faster than ordinary tables. However, in the event of a conflict, OS restart, database restart, primary/standby switchover, power supply interruption, or unclean shutdown, unlogged tables will be cleared, posing a risk of data loss. The contents of an unlogged table are not replicated to standby servers. Any indexes created on an unlogged table are automatically unlogged as well.

  Usage scenario: Considering the data security risks associated with unlogged tables, users should back up data before using such tables, for example, before a system upgrade.

  Troubleshooting: If data is missing in the indexes of unlogged tables due to some unexpected operations such as an unclean shutdown, users should rebuild the indexes with errors.

- **GLOBAL | LOCAL**

  Specifies whether to create a global or local temporary table by including the GLOBAL or LOCAL keyword before TEMP or TEMPORARY. If the GLOBAL keyword is specified, GaussDB will create a global temporary table. Otherwise, it will create a local temporary table.

- **TEMPORARY | TEMP**

  Specifies whether to create a temporary table by including the TEMP or TEMPORARY keyword. Temporary tables are categorized as global or local. When you use the GLOBAL keyword, a global temporary table is created; otherwise, it is a local table.

  The metadata of global temporary tables is visible to all sessions, and it remains in place even after the sessions have concluded. User data, indexes, and statistics are isolated between sessions, allowing each session to access or modify only the data it has committed. Global temporary tables operate in two modes. The first mode is **ON COMMIT PRESERVE ROWS** (session level), which automatically clears user data when sessions conclude. The second mode is **ON COMMIT DELETE ROWS** (transaction level), which automatically clears user data upon COMMIT or ROLLBACK. If the **ON COMMIT** option is not specified during table creation, the session level will be used by default. When creating global temporary tables, you have the flexibility to specify a schema, setting them apart from local temporary tables.

  Local temporary tables are visible only to the current session and will be automatically dropped at the end of the session. This allows you to create and utilize temporary tables within the current session, even if other CNs

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

encounter issues, provided that the CN connected to the session is operational. Temporary tables are created solely within the current session. If a DDL statement involves operations on temporary tables, it will result in a DDL error. Therefore, avoid performing operations on temporary tables through DDL statements. TEMP is equivalent to TEMPORARY.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- Local temporary tables are made visible to the current session through a schema that starts with **pg_temp**. Therefore, do not manually drop schemas that start with **pg_temp** or **pg_toast_temp** during routine operations.
- If a table is created without specifying the TEMPORARY or TEMP keyword but with a schema starting with **pg_temp** in the current session, the table will be considered a temporary table.
- You cannot ALTER or DROP global temporary tables or indexes if they are being used by other sessions (except for **ALTER INDEX index_name REBUILD**).
- DDL operations on global temporary tables affect only the user data and indexes of the current session. For example, TRUNCATE, REINDEX, and ANALYZE are restricted to the current session.
- You can control the global temporary table feature through the GUC parameter **max_active_global_temporary_table**. For example, setting **max_active_global_temporary_table** to **0** disables the feature.
- Temporary tables are visible only to the current session. Therefore, they cannot be used together with **\parallel on**.
- Temporary tables cannot withstand DN failures or primary/standby switchovers.
- Global temporary tables do not respond to automatic vacuum. In long link scenarios, you are advised to use global temporary tables of ON COMMIT DELETE ROWS or manually execute **VACUUM** periodically. Otherwise, Clogs may not be recycled. (You can set **vacuum_gtt_defer_check_age** to generate an alarm if global temporary tables are not cleared for a long time.)
- Global temporary tables do not support the following scenarios:
  - Global temporary sequences cannot be created. Global temporary tables use shared sequences across sessions, which can only ensure uniqueness but not continuity.
  - Global temporary views cannot be created.
  - Partitioned tables cannot be created.
  - Hash bucket tables cannot be created.
  - Extended statistics are not supported.
  - The ON COMMIT DROP attribute is not supported.
  - GSIs are not supported.
  - M compatibility is not supported.
  - Reading on the distributed standby node is not supported.
  - Stream plans are not supported.
  - Scaling is not supported because scaling rollback may cause data loss and the data will be cleared after scaling.
  - If a DN switchover, DN fault, or CN fault occurs, data may be lost. (Global temporary tables do not have Xlogs, so data is not synchronized between primary and standby nodes.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **IF NOT EXISTS**

  Sends a notice, but does not throw an error, if a table with the same name exists.

- **[ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [MEDIUM | HIGH] ROW AFTER n { day | days | month | months | year | years } OF NO MODIFICATION [ON (EXPR)] | TURBO [LOW | MEDIUM | HIGH] UNIT AFTER n { hour | hours | day | days | month | months | year | years } OF NO MODIFICATION | NONE}]**

  When creating a table, you can call ILM ADD POLICY ROW STORE COMPRESS ADVANCED [MEDIUM | HIGH] ROW to add a medium-or high-level advanced compression policy for row store. For example:

  ```
  CREATE TABLE t1 (a int) ILM ADD POLICY ROW STORE COMPRESS ADVANCED ROW AFTER 3 DAY OF NO MODIFICATION ON (a != 0);
  ```

  The command creates the **t1** table and adds the advanced compression policy: rows that are not modified in three days and with **a != 0** specified.

  - AFTER n { day | month | year } OF NO MODIFICATION: rows that have not been modified for *n* days, months, or years. Advanced compression does not support **hour**.
  - ON (EXPR): a row-level expression used to determine whether a row is hot or cold.
  - EXPR supports only the whitelisted functions (in **Row Expression Function Whitelist**).

    NOTE

    - Among the functions supported by the row-level expression ON (EXPR) in the ILM policy, the output of some may be influenced by compatibility parameters. For instance, after you set **b_format_version** to **'5.7'** and **b_format_dev_version** to **'s2'** in MySQL-compatible mode, the upper function will be unable to convert data to uppercase as usual.
    - The compression level is optional. If not specified, the MEDIUM level will be used by default.

  When creating a table, you can call ILM ADD POLICY to add a cross-page transparent compression policy for row store. For example:

  ```
  CREATE TABLE t1 (a int) ILM ADD POLICY ROW STORE COMPRESS TURBO HIGH UNIT AFTER 3 HOUR OF NO MODIFICATION;
  ```

  The command creates the **t1** table and adds a cross-page transparent compression policy with a high policy level and a 3-hour cold and hot data management interval.

  - **AFTER n { hour | day | month | year } OF NO MODIFICATION**: indicates the compression groups that have not been modified for *n* hours, days, months, or years.
  - If the compression level is not specified, the MEDIUM level is used by default.

- **table_name**

  Specifies the name of the table to be created.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

Based on prefixes in table names, the processing logic of materialized views determines whether the table is the logged table of a materialized view or whether the table is joined with a materialized view. Therefore, do not create tables with prefixes like **mlog_** or **matviewmap_**. Otherwise, certain functionalities of the tables may be affected.

- **column_name**

  Specifies the name of the column to be created in the new table.

- **data_type**

  Specifies the data type of the column.

- **CHARACTER SET | CHARSET charset**

  Assigns a character set to the column. If **COLLATE** is not specified separately, the column will use the default collation of the specified character set.

- **COLLATE collation**

  Assigns a collation to the column (which must be of a collatable data type). If not specified, the default collation will be used.

  You can run the **SELECT * FROM pg_collation;** command to query collations from the pg_collation system catalog. The row starting with **default** in the query result represents the default collation.

  NOTE

  - Only the character type supports specifying a character set.
  - When you specify the binary character set or collation, the system will convert the character type to the binary type. If the system cannot find a matching type mapping, it will report an error. Currently, only the TEXT-to-BLOB mapping is supported.
  - In addition to the binary character set and collation, you can only specify the character set that matches the database encoding.
  - If the character set or collation of a column is not explicitly specified and the default character set or collation of the table is specified, the character set or collation of the column is inherited from the table. If the default character set or collation of a table does not exist, the character set and collation of table columns inherit the character set and collation of the current database when **b_format_behavior_compat_options** contains **'default_collation'**.
  - MySQL-compatible databases (**sql_compatibility = 'MYSQL'**) also support the utf8mb4_bin, utf8mb4_general_ci, utf8mb4_unicode_ci, binary, gbk_chinese_ci, gbk_bin, gb18030_chinese_ci, gb18030_bin, utf8mb4_0900_ai_ci, utf8_general_ci, and utf8_bin collations. For details, see **Table 1-398**.

Table 1-398 Character sets and collations supported in MySQL-compatible mode (sql_compatibility = 'MYSQL')

Collation	Character Set	Description
utf8mb4_general_ci	utf8mb4 (utf8)	The general collation is used, which is case-insensitive.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Collation	Character Set	Description
utf8mb4_unicode_ci	utf8mb4 (utf8)	The general collation is used, which is case-insensitive.
utf8mb4_bin	utf8mb4 (utf8)	The binary collation is used, which is case-sensitive.
binary	binary	The binary collation is used.
gbk_chinese_ci	gbk	The Chinese collation is used.
gbk_bin	gbk	The binary collation is used, which is case-sensitive.
gb18030_chinese_ci	gb18030	The Chinese collation is used.
gb18030_bin	gb18030	The binary collation is used, which is case-sensitive.
utf8mb4_0900_ai_ci	utf8mb4	The unicode collation algorithm (UCA) rule is used, which is case-insensitive.
utf8_general_ci	utf8	The general collation is used, which is case-insensitive.
utf8_bin	utf8	The binary collation is used, which is case-sensitive.

- **LIKE source_table [like_option ...]**

  Specifies a table, known as the source table, from which the target table automatically inherits all column names, their data types, NOT NULL constraints, and default expressions declared as serial.

  The target table and source table are completely decoupled after creation is complete. Changes to the source table will not be applied to the target table, and it is not possible to include data of the target table in scans of the source table.

  The system does not merge inherited columns or constraints with those that share the same name. If the same name is specified explicitly or in another LIKE clause, an error is signaled.

  – Columns in the target table inherit the default expressions (except the serial ones) from the source table only when **INCLUDING DEFAULTS** is specified. The default behavior is to exclude the default expressions, causing all columns in the target table to default to **NULL**.

  – If **INCLUDING UPDATE** is specified, columns in the target table will inherit the **ON UPDATE CURRENT_TIMESTAMP** attribute from the source table. The default behavior is to exclude this attribute.

  – The target table inherits the CHECK constraints from the source table only when **INCLUDING CONSTRAINTS** is specified. Other types of constraints are not inherited, while NOT NULL constraints are always

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

inherited. This rule also applies to table constraints and column constraints.

- If **INCLUDING INDEXES** is specified, indexes on the source table will also be created on the target table. By default, no indexes are created. The tablespace where indexes on the target table are located is the same as that where indexes on the source table are located.
- If **INCLUDING STORAGE** is specified, columns in the target table will inherit **STORAGE** settings from the source table. The default behavior is to exclude **STORAGE** settings.
- If **INCLUDING COMMENTS** is specified, the target table will inherit comments on columns, constraints, and indexes from the source table. The default behavior is to exclude comments.
- If **INCLUDING PARTITION** is specified, the target table will inherit partition definitions from the source table and will no longer be able to use the PARTITION BY clause. The default behavior is to exclude partition definitions.
- If **INCLUDING RELOPTIONS** is specified, the target table will inherit storage parameters (WITH clause) from the source table. The default behavior is to exclude storage parameters.
- If **INCLUDING DISTRIBUTION** is specified, the target table will inherit distribution information (including distribution types and keys) from the source table and will no longer be able to use the DISTRIBUTE BY clause. The default behavior is to exclude distribution information.
- If **INCLUDING ILM** is specified, the target table will inherit ILM policy information from the source table. To also inherit the ILM policy information of partition objects, you will need to specify **INCLUDING PARTITION**.
- **INCLUDING ALL** is a condensed version that encompasses **INCLUDING DEFAULTS**, **INCLUDING UPDATE**, **INCLUDING CONSTRAINTS**, **INCLUDING INDEXES**, **INCLUDING STORAGE**, **INCLUDING COMMENTS**, **INCLUDING PARTITION**, **INCLUDING RELOPTIONS**, **INCLUDING DISTRIBUTION**, and **INCLUDING ILM**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- If **INCLUDING RELOPTIONS** is specified and the storage parameter (that is, the WITH clause of the source table) of the source table does not contain storage_type, the storage engine type of the target table is determined by the value of the GUC parameter **enable_default_ustore_table**.
- If the source table contains the serial, bigserial, or smallserial type, or if sequences created for the source table (using **CREATE SEQUENCE ... OWNED BY**) are set as default values for columns in the source table, these sequences will not be associated with the target table. Instead, the target table will rebuild its own sequences. To share sequences between the source and target tables, you must create shared sequences (without using **OWNED BY**) and set them as default values for columns in the source table.
- You are advised not to set the default value of a column in the source table to the sequence specific to another table especially when the table is distributed in specific node groups because doing so may result in CREATE TABLE ... LIKE execution failures. In addition, doing so may cause the sequence to become invalid in the source sequence because the sequence will also be deleted from the source table when it is deleted from the table that the sequence is specific to. To share a sequence among multiple tables, you are advised to create a shared sequence for them.
- If the source table is a ROWID table and INCLUDING ALL or INCLUDING RELOPTIONS is specified, the new table is also a ROWID table. If only INCLUDING INDEXES is specified, the new table is a non-ROWID table and the **ROWNO** index of the source table is not copied.
- **CREATE TABLE table_name LIKE source_table;** is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**) where **b_format_version** is **5.7** and **b_format_dev_version** is **s2**.
- In MySQL-compatible databases where **b_format_version** is **5.7** and **b_format_dev_version** is **s2**, the **INCLUDING** and **EXCLUDING** options cannot be specified. The default setup is equivalent to specifying **INCLUDING ALL**.
- If **source_table** is a system catalog, **INCLUDING ALL** or **INCLUDING DISTRIBUTION** cannot be specified.

- **AUTO_INCREMENT [=] value**

  Specifies an initial value for an auto-increment column. The value must be a positive number no greater than $2^{127} - 1$.

NOTICE

This clause takes effect only when **sql_compatibility** is set to **'MYSQL'**.

- **COMMENT [=] 'string'**
  - The COMMENT [=] 'string' clause is used to add comments to a table.
  - COMMENT 'string' in **column_constraint**: adds a comment on a column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - COMMENT 'string' in **table_constraint**: adds a comment on the indexes corresponding to the primary key and unique key.

NOTICE

- This clause takes effect only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).
- A table-level comment can contain a maximum of 2,048 characters, while a column- or index-level comment can contain a maximum of 1,024 characters.
- Comments in **table_constraint** support only the primary and unique keys. Other constraints are not supported.

- **ENGINE**

  This parameter is supported in MySQL-compatible mode and is used solely for syntax adaptation. It is limited to InnoDB, but no actual impact has been noted.

NOTICE

The ENGINE syntax cannot be used in **CREATE TABLE table_name LIKE source_table;**.

- **COLVIEW | NOCOLVIEW**

  Specifies whether to create IMCV table metadata for a table, column, or partition. To create an IMCV table, you should enable the **enable_htap** parameter.

  This clause cannot be used together with the LIKE source_table syntax.

- **PRIORITY**

  Specifies a priority for loading data into the IMCV table. The options include high, low, and default.

- **WITH ({ storage_parameter = value } [, ...])**

  Specifies an optional storage parameter for the table or an index. The WITH clause for a table can also include **OIDS=FALSE** to specify that no OID will be allocated.

NOTICE

- The WITH clause does not support **OIDS=TRUE**.
- For numeric columns of arbitrary precision, it is advisable to specify precision **p** and scale **s**. If you do not specify precision **p** or scale **s**, the output directly follows the input format.

See details of parameters below.

  - FILLFACTOR

    Specifies a fillfactor for the table. It is a percentage between 10 and 100. **100** (complete packing) is the default. In the case of Ustore, the default

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

is **92**. When a smaller fillfactor is specified, INSERT operations pack table pages only to the indicated percentage; The remaining space on each page is reserved for updating rows on that page. This gives UPDATE a chance to place the updated copy of a row on the same page as the original, which is more efficient than placing it on a different page. For a table whose entries are never updated, complete filling (fillfactor = 100) is the best choice. However, in heavily updated tables, smaller fillfactors are appropriate.

Value range: 10–100.

- ORIENTATION

  Specifies the orientation for storing table data. Once set, this parameter cannot be modified.

  Value range:

  - **ROW**: Table data is stored in row orientation.

    Row store is ideal for OLTP workloads because they can handle a high volume of interactive transactions, each of which usually involves multiple table columns. Utilizing row store can enhance query performance.

  Default value: **ROW** (when an ordinary tablespace is specified).

- STORAGE_TYPE

  Specifies the storage engine type. Once set, this parameter cannot be modified.

  Value range:

  - **USTORE**: The table uses an in-place update storage engine. To prevent space bloat, be sure to enable the **track_counts** and **track_activities** parameters for Ustore tables.
  - **ASTORE**: The table uses an append-only storage engine.

  Default value: subject to the **enable_default_ustore_table** parameter, which defaults to in-place update storage.

- TOAST.TOAST_STORAGE_TYPE

  Specifies the storage type of the TOAST table. When the GUC parameter **enable_enhance_toast_table** is set to **on**, the storage type of the TOAST table can be changed to enhanced TOAST through the VACUUM FULL operation.

  Value range:

  - **ENHANCED_TOAST**: indicates that the storage type of the TOAST table is enhanced TOAST.
  - **TOAST**: indicates that the storage type of the TOAST table is TOAST.

  Default value:

  If **TOAST.TOAST_STORAGE_TYPE** is not specified during table creation, the storage type of the TOAST table depends on the GUC parameter **enable_enhance_toast_table**. If this parameter is set to **on**, the storage type of the TOAST table is ENHANCED_TOAST. If this parameter is set to **off**, the storage type of the TOAST table is TOAST.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- INIT_TD

  Specifies the number of TDs to be initialized when a Ustore table is created. This parameter can be modified by running the **ALTER TABLE** command. Note that this parameter affects the maximum size of a single tuple stored on the data page, calculated as **MAX_TUPLE_SIZE** = **BLCKSZ** – **INIT_TD** × **TD_SIZE**. For instance, if you increase **INIT_TD** from **4** to **8**, the maximum size of a single tuple will decrease by 4 × **INIT_TD**.

  Value range: 2–128. The default value is **4**.

- COMPRESSLEVEL

  Specifies a compression level within the selected compression tier. It determines the data compression ratio and time. Offering a more granular division within the same compression tier gives users greater flexibility in choosing the compression ratio and compression time. Typically, a higher value signifies a higher compression ratio and longer compression time within the same compression tier—and the reverse also applies.

  Value range: 0–3.

  Default value: **0**.

- segment

  Specifies that data is stored in segment-page mode. This parameter supports only row-store tables. Protection against unauthorized deletion and damage of physical files 1 to 5 is not supported. This parameter cannot be created during the upgrade observation period.

  Value range: **on** and **off**.

  Default value: **off**.

- hashbucket

  Creates a hash bucket table. This parameter supports only row-store tables.

  Value range: **on** and **off**.

  Default value: **off**.

NOTICE

- When creating a hash bucket table, ensure that the current database is bound to the default node group of the cluster. For details about how to bind a database to a node group, see **ALTER DATABASE**.
- In current version, DDL operations on hash bucket tables are affected. Therefore, you are advised not to frequently perform DDL operations on hash bucket tables.
- Hash bucket tables are bound to segmented paging storage, that is, when **hashbucket** is set to **on**, **segment** is set to **on**.
- You are advised not to create a small table (ordinary table: < 8 MB; partitioned table: < 8 MB x Number of partitions) as a hash bucket table. The sequential scan performance deteriorates twice that of an ordinary table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- enable_tde

  Specifies that the table is an encrypted table. Before storing data in an encrypted table, the database will encrypt the data first. Before using this parameter, ensure that the GUC parameter **enable_tde** is enabled and **tde_key_info** is properly set. For detailed usage, consult "Feature Usage Guidelines > Transparent Data Encryption" in *Developer Guide*. This parameter supports only row-store tables, segment-page tables, hash bucket tables, temporary tables, and unlogged tables.

  Value range: **on** and **off**. When **enable_tde** is set to **on**, **key_type**, **cmk_id**, and **dek_cipher** are automatically generated by the database and cannot be manually specified or modified.

  Default value: **off**.

- **encrypt_algo**

  Specifies the algorithm for encrypting tables. This parameter must be used together with **enable_tde**.

  Value range: **AES_128_CTR**, **SM4_CTR**, **AES_128_CTR_SHA256_HMAC**, **SM4_CTR_SM3_HMAC**, and **AES_128_GCM** (character string).

  Default value:

  - When **enable_tde** is **on**:
    - If the **table_algorithm** subparameter within the GUC parameter **tde_encrypt_config** is not empty, the default value is the same as that of **table_algorithm**.
    - If the **table_algorithm** subparameter within the GUC parameter **tde_encrypt_config** is empty, the default value is **AES_128_CTR**.
  - When **enable_tde** is **off**, the default is empty.

- parallel_workers

  Specifies the number of bgworker threads started when an index is created. For example, value **2** indicates that two bgworker threads are started to create indexes concurrently.

  Value range: [0,32], int type. **0** indicates that parallel index creation is disabled.

  Default value: This parameter is not set, indicating that the parallel index creation function is disabled.

- **dek_cipher**

  Specifies the DEK ciphertext. This storage parameter cannot be manually set or modified. After a user sets the **enable_tde** parameter for a table, the database automatically generates a data key.

  Value range: character strings.

  Default value: empty.

- **key_type**

  Specifies the type of the master key. This storage parameter cannot be manually set or modified. After the **enable_tde** parameter is set for a table, the database automatically obtains the master key type from the GUC parameter **tde_key_info**.

  Value range: character strings.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: empty.

- **cmk_id**

  Specifies the ID of the master key. This storage parameter cannot be manually set or modified. After the **enable_tde** parameter is set for a table, the database automatically obtains the master key ID from the GUC parameter **tde_key_info**.

  Value range: character strings.

  Default value: empty.

- hasuids

  If this parameter is set to **on**, a unique table-level ID is allocated to a tuple when the tuple is updated.

  Value range: **on** and **off**.

  Default value: **off**.

- logical_repl_node

  Specifies the name of the DN responsible for returning logical logs to the CN during logical decoding of a replication table. For the replication table, if this parameter is not specified, the first node in the node group where the current table is located is used by default. When the RESET operation is performed on this option, the node is reset to the first node of current table.

  Value range: character strings.

  Default value: For the non-replication table, this parameter is empty by default. For the replication table, this parameter is set to the name of the first node by default.

- skip_logical_deparse

  Skips this table during logical decoding in the database. You are advised not to use this parameter. Otherwise, logical decoding may miss some content.

  Value range: **on** and **off**.

  Default value: **off**.

- STAT_STATE

  Identifies whether table statistics are locked. If locked, the table statistics cannot be updated. This parameter does not take effect currently.

  Value range: **locked** and **unlock**.

  Default value: **unlock**.

- statistic_granularity

  Records the default **partition_mode** when the table analyzes statistics. For details about **PARTITION_MODE**, see **ANALYZE | ANALYSE**. This parameter setting does not work for non-partitioned tables.

  Value range: See **PARTITION_MODE** in **ANALYZE | ANALYSE**.

  Default value: **AUTO**.

- autovacuum_enabled

  Specifies whether to enable AUTOVACUUM for the table.

  Value range: **on** and **off**.

  Default value: **on**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- autovacuum_vacuum_threshold

  (Applicable when AUTOVACUUM is enabled) Specifies the minimum number of updated or deleted tuples needed to trigger a VACUUM in the table. (This parameter applies specifically to Astore tables.)

  Value range: 0–2147483647.

  Default value: consistent with the GUC parameter **autovacuum_vacuum_threshold**.

- autovacuum_analyze_threshold

  (Applicable when AUTOVACUUM is enabled) Specifies the minimum number of inserted, updated, or deleted tuples needed to trigger an ANALYZE in the table.

  Value range: 0–2147483647.

  Default value: consistent with the GUC parameter **autovacuum_analyze_threshold**.

- autovacuum_vacuum_scale_factor

  (Applicable when AUTOVACUUM is enabled) Specifies the ratio of inserted, updated, or deleted tuples needed to trigger a VACUUM in the table. (This parameter applies specifically to Astore tables.)

  Value range: 0.0–100.0.

  Default value: consistent with the GUC parameter **autovacuum_vacuum_scale_factor**.

- autovacuum_analyze_scale_factor

  (Applicable when AUTOVACUUM is enabled) Specifies the ratio of inserted, updated, or deleted tuples needed to trigger an ANALYZE in the table.

  Value range: 0.0–100.0.

  Default value: consistent with the GUC parameter **autovacuum_analyze_scale_factor**.

- autovacuum_freeze_min_age

  (Applicable when AUTOVACUUM is enabled) Specifies the minimum age of row versions. Rows older than this age will be frozen.

  Value range: 0–576460752303423487.

  Default value: consistent with the GUC parameter **vacuum_freeze_min_age**.

- autovacuum_freeze_max_age

  (Applicable when AUTOVACUUM is enabled) Specifies the maximum age (in transactions) that the table's **pg_class.relfrozenxid** field can reach before a VACUUM operation is forced. Note that the system will launch AUTOVACUUM processes even when AUTOVACUUM is otherwise disabled. The VACUUM operation also allows removal of old files from the **pg_clog/** subdirectory. (This parameter applies specifically to Astore tables.)

  Value range: 100000–576460752303423487.

  Default value: consistent with the GUC parameter **autovacuum_freeze_max_age**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- autovacuum_freeze_table_age

  (Applicable when AUTOVACUUM is enabled) Specifies the time when the table will remain unchanged, marked as not requiring AUTOVACUUM. (This parameter applies specifically to Astore tables.)

  Value range: 0–576460752303423487.

  Default value: consistent with the GUC parameter **vacuum_freeze_table_age**.

- vacuum_truncate

  Specifies whether to enable the VACUUM TRUNCATE function of a table to truncate the last empty page during VACUUM.

  Value range: **on** and **off**.

  Default value: **on**.

  NOTE

  - This parameter is valid only for Astore tables and does not take effect for hash bucket tables and segment tables.
  - During the upgrade from a version that does not support this parameter to a version that supports this parameter, this parameter cannot be created or modified during the upgrade observation period.
  - The toast.**vacuum_truncate** parameter can be used to independently control whether to truncate the last empty page of a TOAST table.

- skip_logical_deparse

  Skips the table during logical decoding of the kernel. This parameter is for kernel processing and is not recommended for users. If it is set improperly, logical decoding may be incomplete.

  Value range: **on** and **off**.

  Default value: **off**.

- enable_update_distkey

  Determines whether the table supports the UPDATE operation on distribution keys. The distribution keys can be updated only when this attribute of the table is set to **on** and constraints are met. Otherwise, distribution keys do not support the UPDATE operation.

  Value range: **on** and **off**.

  Default value: **off**.

  NOTE

  For details about the constraints on updating distributed columns, see **Precautions**.

- hasrowid

  Specifies whether the table is a rowid table. If **hasrowid** is set to **on**, the **rowid** and **rowno** system columns are added to the table, and sequences and indexes are created on the **rowno** system column.

  Value range: **on** and **off**.

  Default value: **off**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This parameter supports only ORA-compatible databases (that is, **sql_compatibility** is set to **ORA**).
- This parameter does not support local temporary tables, unlogged tables, global temporary tables, materialized views, hash bucket tables, or replication tables.
- This parameter cannot be specified together with **hasuids** and **oids**.
- When this parameter is enabled, the ordinary column name cannot be defined as **rowno** or **rowid**.
- The number of data records in the **rowid** table cannot exceed 9223372036854775807.
- When the **ROWID_OPT** option is enabled for the GUC parameter **opt_behavior_knob** in the **rowid** table, the **ROWID** system column predicate condition rewriting, ORDER BY statement rewriting, max/min optimization, distribution pruning, and partition pruning are supported. For details, see **opt_behavior_knob**.

– partition_statistics_min_rows

Specifies the minimum number of rows to be sampled at the partition level when the ANALYZE statement is executed on a partitioned table.

Value range: [0,10000]

Default value: **50**

NOTE

- This parameter is valid only for partitioned tables.
- This parameter is valid only when **PARTITION_MODE** in the ANALYZE statement is set to **ALL** and the GUC parameter **enable_partition_statistics_min_rows** is enabled. For details about **PARTITION_MODE**, see **PARTITION_MODE** in **ANALYZE | ANALYSE**.
- Upper limit: The number of rows sampled from the entire partitioned table cannot exceed the number of rows allowed by the GUC parameter **maintenance_work_mem**.

- **WITHOUT OIDS**

  Specifies that no OID is allocated, equivalent to **WITH(OIDS=FALSE)**.

- **ON COMMIT { PRESERVE ROWS | DELETE ROWS }**

  Specifies subsequent operations on the temporary table when the temporary table is created in a transaction and the transaction is committed. Currently, **PRESERVE ROWS** and **DELETE ROWS** are supported.

  – **PRESERVE ROWS** (default): No action will be taken upon commit. The temporary table and its data will remain unchanged.

  – **DELETE ROWS**: Data in the temporary table will be deleted upon commit.

- **TABLESPACE tablespace_name**

  Specifies the name of the tablespace where the table is to be created. If not specified, the default tablespace will be used.

- **DISTRIBUTE BY**

  Specifies how the table is distributed or replicated between DNs.

  Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **REPLICATION**: replicates each table row to all DNs, ensuring that each DN holds the full set of table data.
- **HASH (column_name)**: distributes data to DNs based on hash values in the specified column.
- **MURMURHASH (diskey_expr)**: distributes data to DNs based on MurmurHash values in the specified column. **diskey_expr** can be a single column name or a column name modified by the **lpad_s(string text, length int [, fill text])** function. When this function is used, only the first parameter can be a column name, for example, lpad_s(a,10,'0'). When **lpad_s** is used as a distribution key expression, the third parameter cannot be omitted.
- **RANGE(column_name)**: maps the specified column by range and distributes its data to the relevant DN.
- **LIST(column_name)**: maps the specified column by value and distributes its data to the relevant DN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- In hash distribution, the maximum number of distribution keys equals the maximum number of columns, capped at 1,600. In MurmurHash distribution, a distribution key accepts only one column (either an expression or a table column). In RANGE(VALUE LESS THAN) distribution and list distribution, a distribution key can accept up to four columns. In RANGE(START END) distribution, a distribution key can only accept one column.
- Only the MURMURHASH distribution supports an expression as the distribution key. The distribution key expression supports only the **lpad_s(string text, length int [, fill text])** function.
- MurmurHash distribution has the following limitations:
  - Stream plans are not supported, but FQS and PGXC plans are supported.
  - Scale-out/in, hash bucket tables, GSIs, and materialized views are not supported.
  - If two hash tables are joined with an equality join condition on the distribution key, the FQS plan is not supported.
  - When working with a MurmurHash distributed table, you need to specify a user-defined node group, which must be a non-installation node group.
  - Tables cannot be created using CREATE TABLE AS.
  - The distribution key of the MURMURHASH distributed table must contain the NOT NULL constraint.
- MurmurHash distribution is designed to maintain consistent data distribution after the migration of a legacy system that employs the MurmurHash algorithm. Currently, scale-out and stream plans are not supported. You are advised to select hash distributed algorithms.
- For a RANGE distribution policy using the VALUE LESS THAN clause, the distribution rules are as follows:
  - The comparison starts from the first column of values to be inserted.
  - If the value of the inserted first column is smaller than the boundary value of the current column in the local slice, the values are directly inserted.
  - If the value of the inserted first column is equal to the boundary value of the current column in the local slice, compare the value of the inserted second column with the boundary value of the next column in the local slice. If the value of the inserted second column is smaller than the boundary value of the next column in the local slice, the values are directly inserted. If they are equal, the comparison of the next columns between the source and target continues.
  - If the value of the inserted first column is greater than the boundary value of the current column in the local slice, compare the value with that in the next slice.
- When a shard of a range distributed table corresponds to multiple DNs, the system calculates the hash value of the distribution key. This hash value is then used to perform modulo operations on the number of DNs to remap to a specific DN. For details, see the examples.
- When a shard of a list distributed table corresponds to multiple DNs, the system calculates the hash value of the distribution key for the default shard. This hash value is then used to perform modulo operations on the number of DNs to remap to a specific DN. For non-default shards, the system utilizes the round robin method to map values in the values list to DNs. For details, see the examples.
- Range/List distributed tables support only scale-out but not scale-in. For details about the slice scale-out rules, contact an administrator.
- When using a replication table, be aware of the following sufficient and necessary conditions for pushdown:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- In scenarios involving window functions, the sufficient and necessary conditions for pushdown may vary depending on the types of window functions:
  - For rank series functions and the ratio_to_report function, there are no restrictions, and all of them supports pushdown.
  - This is because window functions return the same calculation result for rows with the same PARTITION BY and ORDER BY conditions, regardless of the relative sequence of the data.
  - For the row_number and ntile functions, pushdown can be performed if any of the following conditions is met:
    - Only the projection columns of the window function are included.
    - Only the PARTITION BY or ORDER BY columns in the OVER clause (no other columns) and the projection columns of the window function are included.
    - If the OVER clause contains only PARTITION BY or ORDER BY, the PARTITION BY or ORDER BY column is the primary key.
    - If the OVER clause contains both PARTITION BY and ORDER BY, both the PARTITION BY and ORDER BY columns are primary keys.
  - When the column values of PARTITION BY or ORDER BY in the window function are the same but the values of other columns are different, if other columns appear in the projection columns of the window function, the data sequence on different DNs may be inconsistent. As a result, the window function results and data are inconsistent.
  - For the LAG and LEAD functions, pushdown is supported when the **offset** parameter is set to **0**.
  - When **offset** is **0**, a single column is queried independently of the relative data sequence. If **offset** is greater than 0, tuples with the same value may receive a NULL value because of different relative sequences.
  - The first_value, last_value, and nth_value functions supports pushdown if one of the following conditions is met:
    - The field in the parameter expression must be the PARTITION BY or ORDER BY field in the OVER clause.
    - If the OVER clause contains only PARTITION BY or ORDER BY, the PARTITION BY or ORDER BY column is the primary key.
    - If the OVER clause contains both PARTITION BY and ORDER BY, both the PARTITION BY and ORDER BY columns are primary keys.
- If the query of a replication table contains system columns and volatile functions, the query is not pushed down.
- In the context of concatenating aggregate functions, the sufficient and necessary condition for pushing down is that the columns to be concatenated matches the ORDER BY columns, or the ORDER BY columns form a primary key.
- If the UPDATE or DELETE statement uses the WHERE CURRENT OF cursor_name syntax, pushdown is not supported.

– In hash distribution, **column_name** accepts the following data types:

  - Integer types: TINYINT, SMALLINT, INT, BIGINT, and NUMERIC/DECIMAL
  - Character types: CHAR, BPCHAR, VARCHAR, VARCHAR2, NVARCHAR2, and TEXT
  - Date/Time types: DATE, DATEA (effective only in ORA-compatible mode), TIME, TIMETZ, TIMESTAMP, TIMESTAMPTZ, INTERVAL, and SMALLDATETIME

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- In MurmurHash distribution, the column name in **diskey_expr** accepts the following data types:
  - Integer types: TINYINT, SMALLINT, INT, and BIGINT
  - Character types: CHAR, BPCHAR, VARCHAR, VARCHAR2, and NVARCHAR2
- In RANGE(VALUES LESS THAN) distribution or list distribution, **column_name** accepts the following data types:
  - Integer types: TINYINT, SMALLINT, INT, BIGINT, and NUMERIC/DECIMAL
  - Character types: CHAR, BPCHAR, VARCHAR, VARCHAR2, NVARCHAR2, and TEXT
  - Date/Time types: DATE, DATEA (effective only in ORA-compatible mode), TIMESTAMP, and TIMESTAMPTZ
- In RANGE(START END) distribution, **column_name** accepts the following data types:
  - Integer types: TINYINT, SMALLINT, INT, BIGINT, and NUMERIC/DECIMAL
  - Date/Time types: DATE, DATEA (effective only in ORA-compatible mode), TIMESTAMP, and TIMESTAMPTZ

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

Distribution and partition keys can influence the performance of SQL queries. It is crucial to carefully select these keys based on service requirements when creating a table.

- **How to determine whether a distribution key is appropriate:**

  The distribution key of a hash table should evenly distribute data across all DNs to prevent data skew (that is, uneven data distribution among DNs). Here are principles to help you determine whether a distribution key is appropriate:

  1. Check whether data skew occurs.

     Connect to the database and use the following statement to check the number of tuples on each DN (replace *tablename* with the name of the table to be analyzed):

     ```
     gaussdb=# SELECT a.count,b.node_name FROM (SELECT count(*) AS count,xc_node_id FROM tablename GROUP BY xc_node_id) a, pgxc_node b WHERE a.xc_node_id=b.node_id ORDER BY a.count DESC;
     ```

     If there is a significant difference in tuple counts on each DN (for example, several times or dozens of times), it suggests data skew. In that event, follow **2** to adjust the distribution key.

  2. Select a different distribution key and rebuild the table. Currently, distribution keys cannot be adjusted using ALTER TABLE. Therefore, you must rebuild the table whenever adjustments to its distribution keys are required.

     Selection principles:

     Select a column with discrete values as the distribution key to ensure even data distribution to DNs. For instance, consider using the primary key of a table as the distribution key. In the context of a table containing personal information, the ID card number column is a suitable choice for the distribution key.

     In addition to meeting the above principle, you can utilize the join condition in a query as the distribution key. This approach allows join operations to be pushed down to DNs, reducing the amount of communication data between DNs.

- **How to determine whether a partition key is appropriate:**

  A table can be partitioned into "ranges" defined by a key column or set of columns, with no overlap between the ranges of values assigned to different partitions. A partition is created for each range to store corresponding data.

  By adjusting partition keys to store query results in the same or minimal partitions (a technique known as "partition pruning"), you can obtain consecutive I/O to significantly enhance query performance. In practical scenarios, time is commonly used as a filter for queries. Therefore, you can consider selecting the time column as the partition key, with the key value range being tailored based on the total data volume and the size of data queried at a time.

- Range/List distribution:

  If no DN is explicitly specified for the shards of a range/list distributed table, the database utilizes the round robin algorithm to allocate DNs to the shards. Moreover, in range/list distribution scenarios, it is advisable to define as many shards as possible when creating a table to meet future scale-out requirements. If the number of defined shards is less than the number of DNs before scale-out, data cannot be redistributed to new DNs. Of particular note is the fact that the sharding rules are designed by users. In certain extreme cases, scale-out may not resolve the issue of insufficient storage capacity.

- **TO { GROUP groupname | NODE (nodename [, ...]) }**

  Specifies the node group for which the table is to be created.

  TO NODE is intended only for use by internal scale-out tools and is not recommended for direct use by users.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **CONSTRAINT [constraint_name]**

  Specifies the name of a column constraint or table constraint. The optional CONSTRAINT clause specifies constraints that new or updated rows must satisfy for an INSERT or UPDATE operation to succeed.

  There are two ways to define constraints:

  – A column constraint is defined as part of a column definition, and it is bound to a particular column. You cannot add a constraint name to the clause AUTO_INCREMENT | COMMENT 'string' | ENCRYPTED WITH (COLUMN_ENCRYPTION_KEY = column_encryption_key, ENCRYPTION_TYPE = encryption_type_value).

  – A table constraint is not tied to a particular column, and it can encompass more than one column.

  NOTICE

  For each table constraint, **constraint_name** is optional in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**) but required in other databases.

- **NOT NULL**

  Forbids NULL values in columns.

- **NULL**

  Allows NULL values in columns. This is the default.

  This clause is provided solely for compatibility with non-standard SQL databases and is not recommended for use.

- **CHECK (expression)**

  Specifies an expression producing a Boolean result which new or updated rows must satisfy for an INSERT or UPDATE operation to succeed. Expressions evaluating to TRUE or UNKNOWN succeed. Otherwise, an error exception is raised, and the INSERT or UPDATE does not alter the database.

  A CHECK constraint specified as a column constraint should reference that column's value only, while an expression appearing in a table constraint can reference multiple columns.

  NOTE

  - **<>NULL** and **!=NULL** are invalid to expressions and need to be rewritten as **IS NOT NULL**.
  - CHECK constraints cannot be added to system columns. For details about system columns, see "Basic Concepts > System Columns" in *Developer Guide*.

- **DEFAULT default_expr**

  Assigns a default value to a column. The value can be any variable-free expression (in particular, subqueries and cross-references to other columns in the current table are not allowed). The data type of the default expression must match the data type of the column.

  The default expression can be used in any INSERT operation that does not specify a value for the column. If no default value is specified, **NULL** will be used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **ON UPDATE update_expr**

  Specifies an attribute constraint for the column.

  During an UPDATE operation on a tuple in a table:

  If the new values of the updated columns differ from the old values, columns (in the tuple) that have this attribute but are not part of the UPDATE will be automatically updated to the current timestamp. However, if the new values of the updated columns are the same as the old values, those columns will remain unchanged. If columns with this attribute are included in the UPDATE, their values will be updated to the specified value.

  NOTE

  - This attribute can be specified in MySQL-compatible databases version 5.7 (**sql_compatibility = 'MYSQL'**, **b_format_version='5.7'**, and **b_format_dev_version='s1'**). It can also be specified in PG-compatible databases, but the syntax is not supported during upgrade observation.
  - In terms of syntax, **update_expr** supports three keywords: CURRENT_TIMESTAMP, LOCALTIMESTAMP, and NOW(). You can also optionally specify the precision of a keyword with parentheses. The three types of keywords are synonyms of each other and have the same attribute effect. For example, ON UPDATE CURRENT_TIMESTAMP(), ON UPDATE CURRENT_TIMESTAMP(5), ON UPDATE LOCALTIMESTAMP(), and ON UPDATE LOCALTIMESTAMP(6). If the keyword does not contain parentheses or contains empty parentheses, the precision is 0. The NOW keyword cannot contain parentheses. In PG-compatible mode, ON UPDATE CURRENT_TIMESTAMP(), ON UPDATE LOCALTIMESTAMP(), and ON UPDATE NOW(5) are not supported.
  - This attribute can be specified only for columns of the following types: timestamp, datetime, date, time without time zone, smalldatetime, and abstime.
  - If the CREATE TABLE AS syntax is used, the column attributes will not be inherited.
  - The CREATE TABLE LIKE syntax can use **INCLUDING UPDATE** or **EXCLUDING UPDATE** to inherit or exclude a constraint. In PG-compatible mode, only **EXCLUDING UPDATE** can be used to exclude a constraint. The **INCLUDING ILM** option is added to copy ILM policy information from the source table. When combined with the **INCLUDING PARTITION** option, it can also copy the policy information of partition objects from the source table.
  - The precision specified by this attribute can be different from the precision specified by the type in the corresponding column. After the column value is updated through this attribute, the minimum precision is displayed. For example, **CREATE TABLE t1 (col1 timestamp(6) ON UPDATE CURRENT_TIMESTAMP(3));**. If the UPDATE syntax triggers the attribute to take effect, three decimal places in the value of **col1** are displayed after the update.
  - The same column cannot be specified for this attribute and the generated column constraint at the same time.
  - This attribute cannot be specified for the partition key in a partitioned table.
  - In distributed scenarios, this attribute cannot be specified for distribution keys and primary keys.

- **AUTO_INCREMENT**

  Marks a column as auto-increment.

  Currently, the auto-increment column is a local auto-increment column by default. The auto-increment column increases independently on each DN and does not affect each other. The auto-increment column is not unique globally.

  If the value of this column is not specified (or the value of this column is set to **0**, **NULL**, or **DEFAULT**), the value of this column is automatically increased by the auto-increment counter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

If this column is inserted or updated to a value greater than the current auto-increment counter, the auto-increment counter is updated to this value after the command is executed successfully.

The initial auto-increment value is set by the AUTO_INCREMENT [=] value clause. If it is not set, the default value **1** is used.

NOTE

- The auto-increment column can be specified only when **sql_compatibility** is set to **'MYSQL'**.
- The data type of the auto-increment column can only be integer, 4-byte or 8-byte floating point, or Boolean.

  An error will occur if auto-increment continues after the column reaches the upper limit of the data type.
- Each table can have only one auto-increment column.
- It is recommended that the local auto-increment column be the first column of a non-global secondary index. Otherwise, errors may occur when some operations are performed on a table that contains auto-increment columns, for example, ALTER TABLE EXCHANGE PARTITION.
- The DEFAULT value cannot be specified for an auto-increment column.
- The expression of the CHECK constraint cannot contain auto-increment columns.
- You can specify that the auto-increment column can be NULL. If it is not specified, the auto-increment column contains the NOT NULL constraint by default.
- When you create a table with a local auto-increment column, the system automatically creates a local sequence on each DN to serve as the auto-increment counter. You cannot modify or delete this sequence using sequence-related functionalities, but you can view its current value. Do not configure other sequences to depend on or join with this auto-increment column.
- The auto-increment and refresh operations of the auto-increment counter are not rolled back.
  - Before data is inserted into a table, **0** or **NULL** triggers auto-increment. After data is inserted or updated to a table, the auto-increment counter is updated. If an error is reported after auto-increment, and data is not inserted or updated to the table, the auto-increment counter does not roll back. Subsequent insert statements trigger auto-increment based on the auto-increment counter. As a result, the values of the auto-increment columns in the table are discontinuous.
  - Inserting or importing reserved auto-increment cache values in batches can also lead to discontinuous values in the auto-increment column. For details, see the description of the **auto_increment_cache** parameter.
- An auto-increment column cannot be used as a distribution key.
- Currently, local temporary tables do not support auto-increment columns.
- DISTRIBUTE BY REPLICATION does not support auto-increment columns.

- **UNIQUE [KEY] index_parameters**

  Specifies that a column or a combination of columns must be unique across the entire table.

  For the purpose of a UNIQUE constraint, NULL is not considered equal.

  Semantically equivalent to UNIQUE, UNIQUE KEY is available only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).

- **UNIQUE [index_name][USING method] ({ {column_name | (expression) } [ASC | DESC] }[, ...]) index_parameters**

  Specifies that a column or a combination of columns must be unique across the entire table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

For the purpose of a UNIQUE constraint, NULL is not considered equal.

index_name indicates the index name.

NOTICE

- **index_name** is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).
- If both **constraint_name** and **index_name** are specified for the UNIQUE constraint, the value of **index_name** will be used as the index name.

- **PRIMARY KEY index_parameters**

  PRIMARY KEY [USING method] ({ column_name [ASC | DESC] } [, ...]) index_parameters

  Specifies that one or more columns in the table require unique, non-NULL values.

  Only one primary key can be specified for a table.
- **REFERENCES**

  The current version does not support the REFERENCES clause.
- **USING method**

  Specifies the name of the index method to be used.

  For details about the value range, see **USING method** in **CREATE INDEX**.

NOTICE

- **USING method** is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).
- In MySQL-compatible mode, when **USING method** is not specified, Astore will default to B-tree indexing, and Ustore will default to UB-tree indexing.
- Since the underlying layer of GSIs is based on UB-tree storage, GSIs will be built with UB-tree storage, even if users specify B-tree storage.
- If the storage mode of a table is Ustore and the USING BTREE constraint is specified in an SQL statement, the underlying layer will automatically treat it as USING UBTREE.

- **ASC | DESC**

  ASC specifies an ascending (default) sort order. **DESC** specifies a descending sort order.

NOTICE

ASC | DESC is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).

- **expression**

  Specifies an expression based on one or more columns of the table. It must be written in parentheses.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

Expression indexes in the UNIQUE constraint are supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).

- **DEFERRABLE | NOT DEFERRABLE**

  Specifies whether the constraint can be deferred. A constraint that is not deferrable will be checked immediately after every command. Checking of constraints that are deferrable can be postponed until the end of the transaction (using the **SET CONSTRAINTS** command). **NOT DEFERRABLE** is the default. Currently, only the UNIQUE and PRIMARY KEY constraints accept this clause. (That is, other constraints are not deferrable.)

- **INITIALLY IMMEDIATE | INITIALLY DEFERRED**

  Specifies the default time to check the constraint if it is deferrable. A constraint that is not deferrable is always **INITIALLY IMMEDIATE** and cannot be declared as **INITIALLY DEFERRED**.

  - If the constraint is **INITIALLY IMMEDIATE**, it is checked after each statement. This is the default.
  - If the constraint is **INITIALLY DEFERRED**, it is checked only at the end of the transaction.

  The constraint check time can be altered with the **SET CONSTRAINTS** command.

- **USING INDEX TABLESPACE tablespace_name**

  Specifies a tablespace in which an index associated with a UNIQUE or PRIMARY KEY constraint will be created. If this clause is not specified, the index will be created in the default tablespace named **default_tablespace**. If **default_tablespace** is empty, the default tablespace of the database will be consulted.

- DATATYPE_CL = client_logic_type

  Specifies the type of the column to be encrypted. This parameter is optional.

  Value range: **BYTEAWITHOUTORDER** or **BYTEAWITHOUTORDERWITHEQUAL**.

- **COLUMN_ENCRYPTION_KEY = column_encryption_key**

  Specifies the name of the CEK in the ENCRYPTED WITH constraint.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **ENCRYPTION_TYPE = encryption_type_value**

  When setting the encryption type in the ENCRYPTED WITH constraint, you can choose between **DETERMINISTIC** or **RANDOMIZED** as **encryption_type_value**.

- **[DEFAULT] CHARACTER SET | CHARSET [=] default_charset**

  Specifies the default character set of the table. If this parameter is specified separately, the table will use the default collation of the specified character set.

- **[DEFAULT] COLLATE [=] default_collation**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the default collation of the table. If this parameter is specified separately, the table will default to the character set corresponding to the specified collation.

This syntax is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**). For details about the collation, see **Table 1-398**.

NOTE

If the character set or collation of a table is not explicitly specified but the default character set or collation of the schema is specified, the table will inherit the character set and collation from the schema. If the default character set or collation of the schema cannot be found, the table will inherit the character set and its corresponding collation from the current database when **b_format_behavior_compat_options** includes **'default_collation'**.

Examples

- Create an ordinary table.

```
gaussdb=# CREATE TABLE tbl_test1(
    id       int,
    name     varchar(50),
    province varchar (60), -- Province
    country  varchar (60) DEFAULT 'China' -- Country
);

gaussdb=# DROP TABLE tbl_test1;
```

- **The WITH clause adds storage parameters to a table or index.**

```
-- Create a table and specify a fillfactor.
gaussdb=# CREATE TABLE tbl_test2(
    id       int,
    name     varchar(50),
    province varchar (60), -- Province
    country  varchar (60) DEFAULT 'China' -- Country
) WITH (FILLFACTOR = 70);

-- Create a table and specify a storage engine.
gaussdb=# CREATE TABLE tbl_test3(
    id       int,
    name     varchar(50),
    province varchar (60), -- Province
    country  varchar (60) DEFAULT 'China' -- Country
) WITH (STORAGE_TYPE = ASTORE);

-- Drop.
gaussdb=# DROP TABLE tbl_test2;
gaussdb=# DROP TABLE tbl_test3;
```

- Use an encrypted table.

  Before using this parameter, ensure that the GUC parameter **enable_tde** is enabled and **tde_key_info** is properly set. For detailed usage, consult "Feature Usage Guidelines > Transparent Data Encryption" in *Developer Guide*.

```
gaussdb=# CREATE TABLE  t1 (c1 TEXT) WITH (enable_tde=on);

-- Drop the table.
gaussdb=# DROP TABLE t1;
```

- Use a temporary table.

```
-- Create a temporary table and drop it when the transaction is committed.
gaussdb=# CREATE TEMPORARY TABLE test_t2(
    id       CHAR(7),
    name     VARCHAR(20),
    province VARCHAR(60),                    -- Province
    country   VARCHAR(30) DEFAULT 'China'       -- Country
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
) ON COMMIT DELETE ROWS;
gaussdb=# DROP TABLE test_t2;
```

- Specify a character set and collation during table creation.

```
-- Create a frontend database.
gaussdb=# CREATE DATABASE testdb1 ENCODING = 'UTF8';
gaussdb=# \c testdb1

-- Create table t1. Set the default character set of t1 to utf8mb4 and the default collation to utf8mb4_bin. Set the character set and collation of the c1 column to the default values of the table. Set the character set of the c2 column to utf8mb4, and its collation to utf8mb4_unicode_ci.
testdb1=# CREATE TABLE t1(c1 text, c2 text charset utf8mb4 collate utf8mb4_unicode_ci) charset utf8mb4 collate utf8mb4_bin;

-- Drop.
testdb1=# DROP TABLE t1;
testdb1=# \c postgres
gaussdb=# DROP DATABASE testdb1;
```

- IF NOT EXISTS keyword

  If this keyword is used, NOTICE is reported when the table does not exist. If this keyword is not used, ERROR is reported. In either case, the table fails to be created.

```
gaussdb=# CREATE TABLE test_t3(id INT);
-- Create a table named test_t3.
gaussdb=# CREATE TABLE test_t3(id INT);
ERROR:  Relation "test_t3" already exists in schema "public".
DETAIL:  Creating new table with existing name in the same schema.

-- Use the IF NOT EXISTS keyword.
gaussdb=# CREATE TABLE IF NOT EXISTS test_t3(id INT);
NOTICE:  Relation test_t3 already exists, skipping.
CREATE TABLE

-- Drop the table.
gaussdb=# DROP TABLE test_t3;
```

- Specifying a tablespace during table creation

```
-- Create a tablespace.
gaussdb=# CREATE TABLESPACE ds_tbs1 RELATIVE LOCATION 'tablespace/tablespace_1';

-- Specify a tablespace when creating a table.
gaussdb=# CREATE TABLE test(id CHAR(7), name VARCHAR(20)) TABLESPACE ds_tbs1;

-- Drop the table and tablespace.
gaussdb=# DROP TABLE test;
gaussdb=# DROP TABLESPACE ds_tbs1;
```

- Specifying the AUTO_INCREMENT column during table creation

```
-- The local auto-increment column is not used as the distribution key. The value starts from 10 for each DN. (Change the DN name by running SELECT node_name FROM pgxc_node WHERE node_type = 'D' based on the actual situation.)
gaussdb=# CREATE TABLE local_autoinc(col int AUTO_INCREMENT, col1 int) AUTO_INCREMENT = 10 DISTRIBUTE BY LIST(col1)(
    SLICE s1 VALUES (1) DATANODE datanode1,
    SLICE s2 VALUES (2) DATANODE datanode2
);

-- You are advised to use the auto-increment column as the first column of the index to create an index.
gaussdb=# CREATE INDEX local_autoinc_ai ON local_autoinc(col);

-- Distribute data to DN1. The value NULL triggers auto-increment, and the auto-increment value is 10.
gaussdb=# INSERT INTO local_autoinc(col, col1) VALUES(NULL,1);

-- Distribute data to DN2. The value 0 triggers auto-increment, and the auto-increment value is 10.
gaussdb=# INSERT INTO local_autoinc(col, col1) VALUES(1 - 1,2);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Distribute data to DN1. The value 100 does not trigger auto-increment. After data is inserted successfully, the auto-increment value is updated to 100.
gaussdb=# INSERT INTO local_autoinc(col, col1) VALUES(100,1);

-- Distribute data to DN1. The value 0 triggers auto-increment, and the auto-increment value is 101.
gaussdb=# INSERT INTO local_autoinc(col, col1) VALUES(0,1);

-- Distribute data to DN2. The value DEFAULT triggers auto-increment, and the auto-increment value is 11.
gaussdb=# INSERT INTO local_autoinc(col, col1) VALUES(DEFAULT,2);

gaussdb=# SELECT col,col1 FROM local_autoinc ORDER BY 2,1;
 col | col1
-----+------
  10 |    1
 100 |    1
 101 |    1
  10 |    2
  11 |    2
(5 rows)

-- Drop.
gaussdb=# DROP TABLE local_autoinc;
```

- Creating a table using CREATE TABLE ... LIKE

```
-- Create the source table t1.
gaussdb=# CREATE TABLE t1(col INT);
CREATE TABLE

gaussdb=# \d t1
      Table "public.t1"
 Column |  Type   | Modifiers
--------+---------+-----------
 col    | integer |

-- Create the target table t2.
gaussdb=# CREATE TABLE t2(LIKE t1);
CREATE TABLE

gaussdb=# \d t2
      Table "public.t2"
 Column |  Type   | Modifiers
--------+---------+-----------
 col    | integer |

-- Drop.
gaussdb=# DROP TABLE t1,t2;
```

Examples of Creating a Table and Adding Constraints to the Table

- NOT NULL constraint

  If no value is specified for a column with a NOT NULL constraint when data is added, an error is reported. You can add NOT NULL constraints to multiple columns in a table.

```
-- Create a table and add a NOT NULL constraint to the id column.
gaussdb=# CREATE TABLE test_t4(
    id       CHAR(7) NOT NULL,
    name     VARCHAR(20),
    province VARCHAR(60),                    -- Province
    country   VARCHAR(30) DEFAULT 'China'        -- Country
);

-- If the value of id is not specified or is NULL during data insertion, the NOT NULL constraint is triggered. As a result, the insertion fails.
gaussdb=# INSERT INTO test_t4 (name,province) VALUES ('scott','Shanghai');
ERROR: The null value in column "id" violates the not-null constraint.
DETAIL: Failing row contains (null, scott, Shanghai, China).
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop the table.
gaussdb=# DROP TABLE test_t4;
```

- UNIQUE constraint

  The UNIQUE keyword applies a UNIQUE constraint to a column and automatically creates a unique index. This constraint prevents duplicate entries in the column, but multiple NULL values are not considered duplicate. You can add UNIQUE constraints to multiple columns in a table.

```
-- Create a table and add a UNIQUE constraint.
gaussdb=# CREATE TABLE test_t5(
    id       CHAR(7) UNIQUE USING INDEX TABLESPACE pg_default,
    name     VARCHAR(20),
    province VARCHAR(60),                    -- Province
    country   VARCHAR(30) DEFAULT 'China'        -- Country
);

-- You can also use the following writing to name a UNIQUE constraint and add the constraint to
multiple columns:
gaussdb=# CREATE TABLE test_t6(
    id       CHAR(7),
    name     VARCHAR(20),
    province VARCHAR(60),                     -- Province
    country   VARCHAR(30) DEFAULT 'China',        -- Country
    CONSTRAINT unq_test_id UNIQUE (id,name)
);
-- The constraint is enforced when data with duplicate IDs is inserted, causing the insertion to fail.
gaussdb=# INSERT INTO test_t5(id) VALUES('0000010');
INSERT 0 1
gaussdb=# INSERT INTO test_t5(id) VALUES('0000010');
ERROR:  Duplicate key value violates unique constraint "test_t5_id_key".
DETAIL:  Key (id)=(0000010) already exists.
-- The constraint is not enforced when multiple instances of data with a NULL ID are inserted.
gaussdb=# INSERT INTO test_t5(id) VALUES (NULL);
INSERT 0 1
gaussdb=# INSERT INTO test_t5(id) VALUES (NULL);
INSERT 0 1

-- Drop tables.
gaussdb=# DROP TABLE test_t5;
gaussdb=# DROP TABLE test_t6;
```

- PRIMARY KEY constraint

  The keyword PRIMARY KEY is used to add a UNIQUE constraint to a column. The column must be unique and cannot be empty. When a PRIMARY KEY constraint is added, a unique index is automatically created for the table, and a NOT NULL constraint is automatically added for the column.

  Only one PRIMARY KEY constraint can be defined in each table.

```
-- Create a table and add a PRIMARY KEY constraint.
gaussdb=# CREATE TABLE test_t6(
    id       CHAR(7) PRIMARY KEY,
    name     VARCHAR(20),
    province VARCHAR(60),                    -- Province
    country   VARCHAR(30) DEFAULT 'China'        -- Country
);
gaussdb=# INSERT INTO test_t6 (id,name,province) VALUES ('0000001','july','Beijing');

-- You can also use the following writing to manually name a UNIQUE constraint and add the
constraint to multiple columns:
gaussdb=# CREATE TABLE test_t7(
    id       CHAR(7),
    name     VARCHAR(20),
    province VARCHAR(60),                    -- Province
    country   VARCHAR(30) DEFAULT 'China',        -- Country
    CONSTRAINT pk_test_t6_id PRIMARY KEY (id,name)
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- The constraint is enforced when data with a NULL ID is inserted.
gaussdb=# INSERT INTO test_t6 (id,name,province) VALUES (NULL,'july','Beijing');
ERROR:  The null value in column "id" violates the not-null constraint.
DETAIL:  Failing row contains (null, july, Beijing, China).

-- The constraint is enforced when data with duplicate IDs is inserted.
gaussdb=# INSERT INTO test_t6 (id,name,province) VALUES ('0000001','ben','Shanghai');
ERROR:  Duplicate key value violates unique constraint "test_t6_pkey"
DETAIL:  Key (id)=(0000001) already exists.
-- Drop tables.
gaussdb=# DROP TABLE test_t6;
gaussdb=# DROP TABLE test_t7;
```

- CHECK constraint

  The keyword CHECK adds a constraint to a column. The CHECK constraint must reference one or more columns in the table, and the result returned by the expression must be a Boolean value. Currently, expressions cannot contain subqueries. Both CHECK and NOT NULL constraints can be defined for the same column.

```
-- Create a table and add a CHECK constraint.
gaussdb=# CREATE TABLE test_t8 (
    id       CHAR(7),
    name     VARCHAR(20),
    age      INT CHECK(age > 0 AND age < 150)
);

-- You can also use the following SQL statement to manually name a CHECK constraint and add the
constraint to one or more columns:
gaussdb=# CREATE TABLE test_t9 (
    id       CHAR(7),
    name     VARCHAR(20),
    age      INT,
    CONSTRAINT chek_test_t8_age CHECK(age > 0 AND age < 150)
);
-- The CHECK constraint is enforced when the inserted value does not comply with the expression,
causing the insertion to fail.
gaussdb=# INSERT INTO test_t8 (id,name,age) VALUES ('0000007','scott',200);
ERROR:  New row for relation test_t8 violates check constraint test_t8_age_check.
DETAIL:  N/A
-- Drop tables.
gaussdb=# DROP TABLE test_t8;
gaussdb=# DROP TABLE test_t9;
```

Table Data Distribution Examples

- REPLICATION

```
gaussdb=# CREATE TABLE test_replication(
    id       CHAR(7),
    name     VARCHAR(20),
    province VARCHAR(60),                    -- Province
    country   VARCHAR(30) DEFAULT 'China'        -- Country
)DISTRIBUTE BY REPLICATION;

-- Query table information.
gaussdb=# \d+ test_replication
                                  Table "public.test_replication"
  Column  |         Type          |             Modifiers              | Storage  | Stats target | Description
----------+-----------------------+------------------------------------+----------+--------------+-------------
 id       | character(7)          |                                    | extended |              |
 name     | character varying(20) |                                    | extended |              |
 province | character varying(60) |                                    | extended |              |
 country  | character varying(30) | default 'China'::character varying | extended |              |
Has OIDs: no
Distribute By: REPLICATION
Location Nodes: ALL DATANODES
Options: orientation=row, logical_repl_node=-1, compression=no, storage_type=USTORE, segment=off
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop.
gaussdb=# DROP TABLE test_replication;
```

- **HASH**

```
-- Define a table using hash distribution.
gaussdb=# CREATE TABLE test_hash(
    id       CHAR(7),
    name     VARCHAR(20),
    province VARCHAR(60),                    -- Province
    country   VARCHAR(30) DEFAULT 'China'        -- Country
)DISTRIBUTE BY HASH(id);

-- Insert data.
gaussdb=# INSERT INTO test_hash VALUES ('0000001', 'Bob',   'Shanghai', 'China'),
                          ('0000002', 'Jack',  'Beijing',  'China'),
                          ('0000003', 'Scott', 'Beijing',  'China');
-- View data distribution.
gaussdb=# SELECT a.count,b.node_name
FROM (SELECT COUNT(*) AS count, xc_node_id FROM test_hash GROUP BY xc_node_id) a,
     pgxc_node b
WHERE a.xc_node_id=b.node_id ORDER BY a.count DESC;

count |     node_name
-------+-------------------
     2 | dn_6001_6002_6003
     1 | dn_6004_6005_6006
(2 rows)
-- Drop the table.
gaussdb=# DROP TABLE test_hash;
```

- **MURMURHASH**

```
-- The test environment contains one CN and six DNs.
-- Create a node group. You can run the SELECT node_name FROM PGXC_NODE WHERE node_type = 'D' command to query the DN name in the node group. In the query result, replace the DN name following WITH in the CREATE NODE GROUP statement as required.
gaussdb=# CREATE NODE GROUP NG1 WITH(datanode1, datanode2, datanode3, datanode4, datanode5, datanode6);
-- Define a table using MurmurHash distribution.
gaussdb=# CREATE TABLE test_murmurhash1 (a int NOT NULL, b int) DISTRIBUTE BY MURMURHASH(a) TO GROUP NG1;
gaussdb=# CREATE TABLE test_murmurhash2 (a int NOT NULL, b int) DISTRIBUTE BY MURMURHASH(lpad_s(a,10,'0')) TO GROUP NG1;
-- Insert data.
gaussdb=# INSERT INTO test_murmurhash1 VALUES(0,1);
gaussdb=# INSERT INTO test_murmurhash2 VALUES(1,2);
-- Query data.
gaussdb=# SELECT * FROM test_murmurhash1;
 a | b
---+---
 0 | 1
(1 row)
gaussdb=# SELECT * FROM test_murmurhash2;
 a | b
---+---
 1 | 2
(1 row)
-- Drop the table.
gaussdb=# DROP TABLE test_murmurhash1;
gaussdb=# DROP TABLE test_murmurhash2;

-- Drop a node group.
gaussdb=# DROP NODE GROUP NG1;
```

- **RANGE**

```
-- Define a table using range distribution. (Change the DN name to the actual one. You can run the SELECT node_name FROM PGXC_NODE WHERE node_type = 'D'; command to query DN names.)
gaussdb=# CREATE TABLE test_range(
    id       INT,
    name     VARCHAR(20),
    province VARCHAR(60),                    -- Province
    country   VARCHAR(30) DEFAULT 'China'        -- Country
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
)DISTRIBUTE BY RANGE(id)(
    SLICE s1 VALUES LESS THAN (100) DATANODE dn_6001_6002_6003,
    SLICE s2 VALUES LESS THAN (200) DATANODE dn_6004_6005_6006,
    SLICE s3 VALUES LESS THAN (MAXVALUE) DATANODE dn_6007_6008_6009
);

-- Insert data.
gaussdb=# INSERT INTO test_range VALUES (52,  'Bob',    'Beijing',   'China');
gaussdb=# INSERT INTO test_range VALUES (100, 'Ben',    'Shanghai',  'China');
gaussdb=# INSERT INTO test_range VALUES (150, 'Scott',  'Guangzhou', 'China');
gaussdb=# INSERT INTO test_range VALUES (300, 'Jordan', 'Beijing',   'China');
-- View data distribution.
gaussdb=# SELECT a.count,b.node_name
FROM (SELECT COUNT(*) AS count, xc_node_id FROM test_range GROUP BY xc_node_id) a,
     pgxc_node b
WHERE a.xc_node_id=b.node_id ORDER BY a.count DESC;
 count |     node_name
-------+--------------------
     2 | dn_6004_6005_6006
     1 | dn_6001_6002_6003
     1 | dn_6007_6008_6009
(3 rows)
-- Query data stored on each DN.
gaussdb=# SELECT b.node_name, a.*
FROM (SELECT *, xc_node_id FROM test_range) a,
     pgxc_node b
WHERE a.xc_node_id=b.node_id order by node_name;
     node_name     | id |  name  | province  | country | xc_node_id
-------------------+-----+--------+-----------+---------+-------------
 dn_6001_6002_6003 |  52 | Bob    | Beijing   | China   | -1072999043
 dn_6004_6005_6006 | 100 | Ben    | Shanghai  | China   |  -564789568
 dn_6004_6005_6006 | 150 | Scott  | Guangzhou | China   |  -564789568
 dn_6007_6008_6009 | 300 | Jordan | Beijing   | China   |  1532339558
(4 rows)
-- Drop the table.
gaussdb=# DROP TABLE test_range;
```

- **LIST**

```
-- Define a table using list distribution. (Change the DN name to the actual one. You can run the
SELECT node_name FROM PGXC_NODE WHERE node_type = 'D';
 command to query DN names.)
gaussdb=# CREATE TABLE test_list(
    id       INT,
    name     VARCHAR(20),
    country   VARCHAR(30) DEFAULT 'China'        -- Country
)DISTRIBUTE BY LIST(country)(
    SLICE s1 VALUES ('China') DATANODE dn_6001_6002_6003,
    SLICE s2 VALUES ('USA')   DATANODE dn_6004_6005_6006,
    SLICE s3 VALUES (DEFAULT) DATANODE dn_6007_6008_6009
);

-- Insert data.
gaussdb=# INSERT INTO test_list VALUES (1,'Scott','China');
gaussdb=# INSERT INTO test_list VALUES (2,'Henry','USA');
gaussdb=# INSERT INTO test_list VALUES (3,'Michael','France');
gaussdb=# INSERT INTO test_list VALUES (4,'Jack','UK');
-- Query data stored on each DN.
gaussdb=# SELECT b.node_name, a.*
FROM (SELECT *, xc_node_id FROM test_list) a,
     pgxc_node b
WHERE a.xc_node_id=b.node_id order by node_name;
     node_name     | id |  name   | country | xc_node_id
-------------------+----+---------+---------+-------------
 dn_6001_6002_6003 |  1 | Scott   | China   | -1072999043
 dn_6004_6005_6006 |  2 | Henry   | USA     |  -564789568
 dn_6007_6008_6009 |  3 | Michael | France  |  1532339558
 dn_6007_6008_6009 |  4 | Jack    | UK      |  1532339558
(4 rows)
-- Drop the table.
gaussdb=# DROP TABLE test_list;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

ALTER TABLE, **DROP TABLE**, and **CREATE TABLESPACE**

Suggestions

- UNLOGGED
  - Data written to unlogged tables or indexes on such tables is not written to WALs, which makes them considerably faster than ordinary tables. Therefore, they are ideal for storing the intermediate result sets of complex queries to enhance query performance.
  - However, unlogged tables lack the primary/standby mechanism, increasing the risk of data loss in case of system failures or unexpected breakpoints. As a result, unlogged tables are not recommended for storing basic data.
- TEMPORARY | TEMP
  - Temporary tables are visible only to the current session and will be automatically dropped at the end of the session.
  - Temporary tables are visible only to the current CN.
- LIKE
  - The target table automatically inherits all column names, their data types, and NOT NULL constraints from the declared table, known as the source table. The new table and source table are completely decoupled after creation is complete.
- LIKE INCLUDING DEFAULTS
  - Columns in the target table inherit the default expressions from the source table only when **INCLUDING DEFAULTS** is specified. The default behavior is to exclude the default expressions, causing all columns in the target table to default to **NULL**.
- LIKE INCLUDING CONSTRAINTS
  - The target table inherits the CHECK constraints from the source table only when **INCLUDING CONSTRAINTS** is specified. Other types of constraints are not inherited, while NOT NULL constraints are always inherited. This rule also applies to table constraints and column constraints.
- LIKE INCLUDING INDEXES
  - If **INCLUDING INDEXES** is specified, indexes on the source table will also be created on the target table. By default, no indexes are created. The tablespace where indexes on the target table are located is the same as that where indexes on the source table are located.
- LIKE INCLUDING STORAGE
  - If **INCLUDING STORAGE** is specified, columns in the target table will inherit **STORAGE** settings from the source table. The default behavior is to exclude **STORAGE** settings.
- LIKE INCLUDING COMMENTS
  - If **INCLUDING COMMENTS** is specified, the target table will inherit comments on columns, constraints, and indexes from the source table. The default behavior is to exclude comments.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- LIKE INCLUDING PARTITION
  - If **INCLUDING PARTITION** is specified, the target table will inherit partition definitions from the source table and will no longer be able to use the PARTITION BY clause. The default behavior is to exclude partition definitions.
- LIKE INCLUDING RELOPTIONS
  - If **INCLUDING RELOPTIONS** is specified, the target table will inherit storage parameters (WITH clause) from the source table. The default behavior is to exclude storage parameters.
- LIKE INCLUDING DISTRIBUTION
  - If **INCLUDING DISTRIBUTION** is specified, the target table will inherit distribution information (including distribution types and keys) from the source table and will no longer be able to use the DISTRIBUTE BY clause. The default behavior is to exclude distribution information.
- LIKE INCLUDING ALL
  - **INCLUDING ALL** is a condensed version that encompasses **INCLUDING DEFAULTS**, **INCLUDING CONSTRAINTS**, **INCLUDING INDEXES**, **INCLUDING STORAGE**, **INCLUDING COMMENTS**, **INCLUDING PARTITION**, **INCLUDING RELOPTIONS**, **INCLUDING DISTRIBUTION**, and **INCLUDING ILM**.
- ORIENTATION ROW
  - Create a row-store table. Row store is ideal for OLTP workloads because they can handle a high volume of interactive transactions, each of which usually involves multiple table columns. Utilizing row store can enhance query performance.
- DISTRIBUTE BY
  - When dealing with a fact table or a dimension table that holds a significant volume of data, it is advisable to create it as a distributed table. This table distributes data to the specified DNs based on hash values in the specified column. The syntax is DISTRIBUTE BY HASH(column_name).
  - When dealing with a dimension table that holds a small volume of data, it is advisable to create it as a replication table. This table replicates each table record to all DNs, ensuring that each DN holds the full set of table data. The syntax is DISTRIBUTE BY REPLICATION.

1.13.9.45 CREATE TABLESPACE

Description

Creates a tablespace in a database.

Precautions

- A system administrator or a user who inherits permissions of the built-in role gs_role_tablespace can create a tablespace.
- **CREATE TABLESPACE** cannot be executed within a transaction block.
- If internal directory or file operations succeed but subsequent steps fail during tablespace creation, residual directories or files may remain. Therefore, before

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

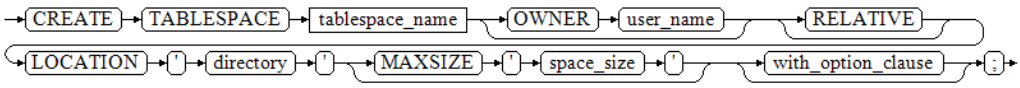
attempting to create a tablespace again, you will need to manually remove the residual content in the specified directory for the tablespace. If there are residual files of soft links for the tablespace in the data directory, delete the residual files, and then perform O&M operations.

- **CREATE TABLESPACE** cannot be used for two-phase transactions. If it fails on some nodes, the execution cannot be rolled back.
- When creating a tablespace, the system distributes the creation instruction to nodes in sequence. If the instruction fails to execute on a remote node, the system will return "CONTEXT: Error message received from nodes: *Node name*" to indicate the source of the error.
- You are advised not to use user-defined tablespaces in the Huawei Cloud scenario.

  This is because user-defined tablespaces are typically intended for use with storage media other than the main storage (device where the default tablespace is situated, like a disk) to isolate I/O resources for different services. In the Huawei Cloud scenario, however, storage devices follow standard configurations and lack additional storage media options. Improper use of user-defined tablespaces can lead to system instability over time and impact overall performance. Therefore, it is advisable to use the default tablespace.

Syntax

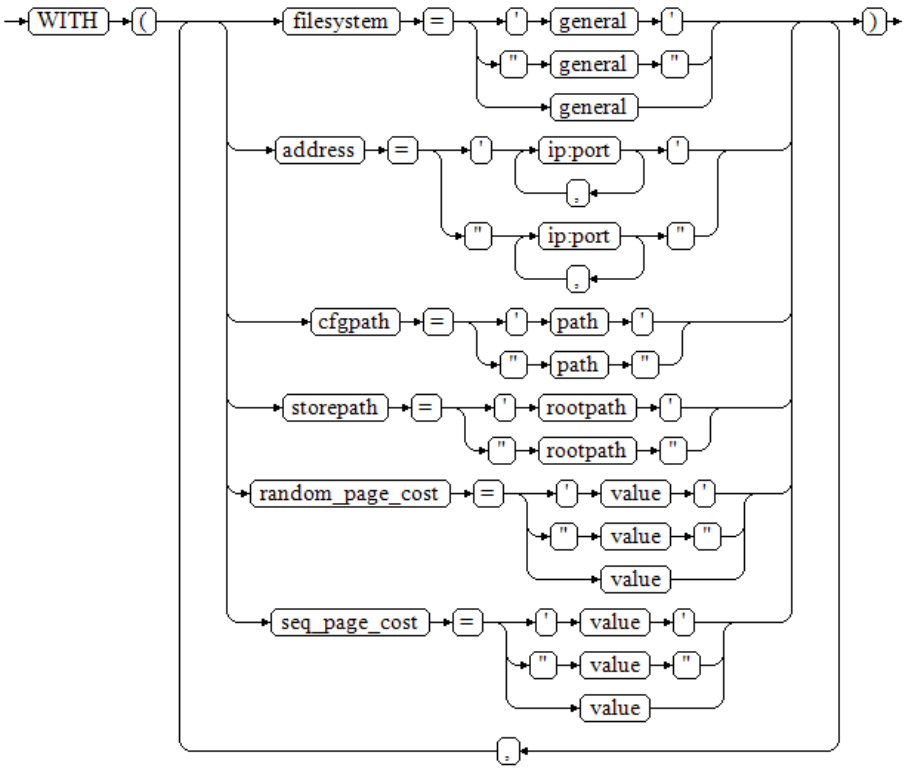
```
CREATE TABLESPACE tablespace_name
    [ OWNER user_name ] [ RELATIVE ] LOCATION 'directory' [ MAXSIZE 'space_size' ]
    [with_option_clause];
```



with_option_clause for creating an ordinary tablespace is as follows:

```
WITH ({filesystem= { ' general ' | '' general '' | general } | address = { ' ip:port [, ... ] ' | '' ip:port [, ... ] '' } |
cfgpath = { ' path ' | '' path '' } | storepath = { ' rootpath ' | '' rootpath '' } | random_page_cost = { ' value ' | ''
value '' | value } | seq_page_cost = { ' value ' | '' value '' | value }}[, ... ])
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- **tablespace_name**

  Specifies name of the tablespace to be created.

  Value range: character strings complying with **Identifier Naming Conventions**.

  NOTE

  The tablespace name must be unique within the database and cannot start with **pg**.

- **OWNER user_name**

  Specifies the owner of the tablespace. By default, the new tablespace is owned by the current user.

  Value range: existing usernames.

  NOTE

  Only system administrators can create tablespaces, but they can assign the ownership of tablespaces to non-system administrators using the OWNER clause.

- **RELATIVE**
  - If this parameter is specified, a relative path will be used. The **LOCATION** directory of the tablespace is relative to each CN/DN's data directory.

    Directory hierarchy: relative path of the CNs and DNs directory **/pg_location/**. A relative path contains a maximum of two levels.
  - If this parameter is not specified, an absolute path will be used for the **LOCATION** directory of the tablespace.
- **LOCATION directory**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the directory for the table space. When an absolute path is used (that is, the **RELATIVE** parameter is not specified), the directory must meet the following requirements:

- The system user must have the read and write permissions on the directory, and the directory must be empty. If the directory does not exist, the system automatically creates it.
- The directory must be an absolute path, and must not contain special characters such as the dollar sign ($) or greater-than sign (>).
- The directory cannot be specified under the database data directory.
- The directory must be a local path.
- You are advised to use a directory in Ext4 format.

Value range: valid directories (character string).

- **MAXSIZE 'space_size'**

  Specifies the maximum size of the tablespace on a single DN.

  Value range: numbers in KB, MB, GB, TB, or PB. Once converted to KB, the numbers must stay within the range of signed integers represented by 64 bits, that is, 1 KB to 9007199254740991 KB.

- **address**

  Specifies the IP address and port number.

- **cfgpath**

  Specifies the configuration file path.

- **storepath**

  Specifies the root path.

- **filesystem**

  Specifies the file system used by the tablespace.

  Value range:

  - **general**: general file system.
  - **hdfs**: Hadoop distributed file system. It is not supported in the current version.

  Default value: **general**.

- **random_page_cost**

  Specifies the time and resources required for reading pages randomly.

  Value range: 0 to 1.79769e+308.

  Default value: *value of the GUC parameter* ***random_page_cost***.

- **seq_page_cost**

  Specifies the time and resources required for reading pages in sequence.

  Value range: 0 to 1.79769e+308.

  Default value: *value of GUC parameter* ***seq_page_cost***.

Examples

```
-- Create a tablespace.
gaussdb=# CREATE TABLESPACE tbs_location1 RELATIVE LOCATION 'test_tablespace/test_tablespace_1';

-- Create a tablespace and specify the maximum value.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CREATE TABLESPACE tbs_location2 RELATIVE LOCATION 'test_tablespace/test_tablespace_2'
MAXSIZE '10G';

-- Query tablespace information.
gaussdb=# SELECT * FROM pg_tablespace WHERE spcname = 'tbs_location2';
    spcname     | spcowner | spcacl | spcoptions | spcmaxsize | relative
----------------+----------+--------+------------+------------+----------
 tbs_location2 |       10 |        |            | 10485760 K | t
(1 row)

-- Create user joe.
gaussdb=# CREATE ROLE joe IDENTIFIED BY '********';

-- Create a tablespace and set its owner to user joe.
gaussdb=# CREATE TABLESPACE tbs_location3 OWNER joe RELATIVE LOCATION 'test_tablespace/
test_tablespace_3';

-- Create a tablespace and specify an absolute path. (Change the path based on the actual situation.)
gaussdb=# CREATE TABLESPACE tbs_location4 LOCATION '/tmp/tbs_location4';

-- Delete tablespaces and users.
gaussdb=# DROP TABLESPACE tbs_location1;
gaussdb=# DROP TABLESPACE tbs_location2;
gaussdb=# DROP TABLESPACE tbs_location3;
gaussdb=# DROP TABLESPACE tbs_location4;
gaussdb=# DROP ROLE joe;
```

Helpful Links

CREATE DATABASE, **CREATE TABLE**, **CREATE INDEX**, **DROP TABLESPACE**, and **ALTER TABLESPACE**

1.13.9.46 CREATE TABLE AS

Description

Creates a table from the results of a query.

It creates a table and fills it with data computed by a **SELECT** command. The table columns have the names and data types associated with the output columns of the **SELECT** (except that you can override the column names by giving an explicit list of new column names).

CREATE TABLE AS bears some resemblance to creating a view, but it is really quite different: it creates a new table and evaluates the query just once to fill the new table initially. The new table will not track subsequent changes to the source tables of the query. In contrast, a view re-evaluates its defining SELECT statement whenever it is queried.

Precautions

- The table created in this way inherits only column names and data types from the output columns of **SELECT**.
- CREATE TABLE AS cannot be used to create a partitioned table.
- If a fault occurs during table creation, the system may be unable to automatically clear the created disk files with a nonzero size, even after the fault has been rectified. This problem seldom occurs and does not affect system running of the database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If you add or modify an ILM policy for a data object and include a row-level expression, you must use only the whitelisted functions (in **Row Expression Function Whitelist**) as the expression.

  NOTE

  Among the functions supported by the row-level expression ON (EXPR) in the ILM policy, the output of some may be influenced by compatibility parameters. For instance, after you set **b_format_version** to **'5.7'** and **b_format_dev_version** to **'s2'** in MySQL-compatible mode, the upper function will be unable to convert data to uppercase as usual.

Syntax

```
CREATE [ [ GLOBAL | LOCAL ] [ TEMPORARY | TEMP ] | UNLOGGED ] TABLE [ IF NOT EXISTS ] table_name
    [ (column_name [, ...] ) ]
    [ { ENGINE [ = ] { InnoDB | 'InnoDB' | "InnoDB" } } [ ,... ] [ ... ] ]
    [ WITH ( {storage_parameter = value} [, ... ] ) ]
    [ ON COMMIT { PRESERVE ROWS | DELETE ROWS } ]
    [ ILM ADD POLICY ROW STORE COMPRESS
 { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON
 ( EXPR )]
 | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION
 | NONE}]
    [ TABLESPACE tablespace_name ]
    [ DISTRIBUTE BY { REPLICATION | { [HASH ] ( column_name ) } } ]
    [ TO { GROUP groupname | NODE ( nodename [, ... ] ) } ]
    AS query
    [ WITH [ NO ] DATA ];
```

For details about each column, see **Syntax**.

Parameters

- **UNLOGGED**

  Specifies whether to create the table as an unlogged table. Data written to unlogged tables is not written to WALs, which makes them considerably faster than ordinary tables. However, in the event of a conflict, OS restart, database restart, primary/standby switchover, power supply interruption, or unclean shutdown, unlogged tables will be cleared, posing a risk of data loss. Contents of an unlogged table are also not replicated to standby nodes. Any indexes created on an unlogged table are automatically unlogged as well.

  - Usage scenario: Considering the data security risks associated with unlogged tables, users should back up data before using such tables, for example, before a system upgrade.
  - Troubleshooting: If data is missing in the indexes of unlogged tables due to some unexpected operations such as an unclean shutdown, users should rebuild the indexes with errors.

- **GLOBAL | LOCAL**

  Specifies whether to create a global or local temporary table by including the GLOBAL or LOCAL keyword before TEMP or TEMPORARY. If the GLOBAL keyword is specified, GaussDB will create a global temporary table. Otherwise, it will create a local temporary table.

- **TEMPORARY | TEMP**

  Specifies whether to create a temporary table by including the TEMP or TEMPORARY keyword. Temporary tables are categorized as global or local.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

When you use the GLOBAL keyword, a global temporary table is created; otherwise, it is a local table.

- The metadata of global temporary tables is visible to all sessions, and it remains in place even after the sessions have concluded. User data, indexes, and statistics are isolated between sessions, allowing each session to access or modify only the data it has committed. Global temporary tables operate in two modes:
  - **ON COMMIT PRESERVE ROWS** (session level): User data is automatically cleared when sessions conclude.
  - **ON COMMIT DELETE ROWS** (transaction level): User data is automatically cleared upon COMMIT or ROLLBACK.

  If the **ON COMMIT** option is not specified during table creation, the session level will be used by default. When creating global temporary tables, you have the flexibility to specify a schema that does not have to start with **pg_temp_**, setting them apart from local temporary tables.
- Local temporary tables are visible only to the current session and will be automatically dropped at the end of the session. This allows you to create and utilize temporary tables within the current session, even if other CNs encounter issues, provided that the CN connected to the session is operational. Temporary tables are created solely within the current session. If a DDL statement involves operations on temporary tables, it will result in a DDL error. Therefore, avoid performing operations on temporary tables through DDL statements.

NOTICE

- Local temporary tables are made visible to the current session through a schema that starts with **pg_temp**. Therefore, users are advised against manually dropping schemas that start with **pg_temp** or **pg_toast_temp** during their routine operations.
- If a table is created without specifying the TEMPORARY or TEMP keyword but with a schema starting with **pg_temp** in the current session, the table will be considered a temporary table.
- Temporary tables are visible only to the current session. Therefore, they cannot be used together with **\parallel on**.
- Temporary tables cannot withstand DN failures or primary/standby switchovers.
- If a global temporary table or index is in use by another session, you cannot perform ALTER or DROP operations on it.
- DDL operations on global temporary tables affect only the user data and indexes of the current session. For example, TRUNCATE, REINDEX, and ANALYZE are restricted to the current session.

- **IF NOT EXISTS**

  Check whether a table with the same name exists. When the IF NOT EXISTS keyword is specified, the system checks the current schema for any dependency relation with the same name as the table you intend to create. If it finds such a relation, the table creation process will be halted, and a

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE will be returned. When IF NOT EXISTS is not specified and a relation with the same name is present in the schema, an ERROR will be returned.

- **table_name**

  Specifies the name of the table to be created.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **column_name**

  (Optional) Specifies column names in the new table. If no column name is specified, the columns in the new table are the same as those entered in the SELECT statement.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **ENGINE**

  This parameter is supported in MySQL-compatible mode and is used solely for syntax adaptation. It is limited to InnoDB, but no actual impact has been noted.

- **WITH (storage_parameter [= value] [, ...])**

  Specifies an optional storage parameter for the table or an index. See details of parameters below.

  NOTE

  These TDE-related storage parameters are supported: **enable_tde**, **encrypt_algo**, **dek_cipher**, **key_type**, and **cmk_id**.

  For details about these parameters, see **CREATE TABLE**.

  – FILLFACTOR

    Specifies a fillfactor for the table. It is a percentage between 10 and 100. **100** (complete packing) is the default. In the case of Ustore, the default is **92**. When a smaller fillfactor is specified, INSERT operations pack table pages only to the indicated percentage; the remaining space on each page is reserved for updating rows on that page. This gives UPDATE a chance to place the updated copy of a row on the same page as the original, which is more efficient than placing it on a different page. For a table whose entries are never updated, complete packing (fillfactor = 100) is the best choice. However, in heavily updated tables, smaller fillfactors are appropriate. The parameter takes effect only on row-store tables.

    Value range: 10–100.

  – ORIENTATION

    Value range: **ROW**, indicating that table data is stored in row orientation.

    Default value: **ROW**.

  – autovacuum_enabled

    Specifies whether to enable AUTOVACUUM for the table.

    Value range: **on** and **off**.

    Default value: **on**.

  – autovacuum_vacuum_threshold

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

(Applicable when AUTOVACUUM is enabled) Specifies the minimum number of updated or deleted tuples needed to trigger a VACUUM in the table. (This parameter applies specifically to Astore tables.)

Value range: 0–2147483647.

Default value: consistent with the GUC parameter **autovacuum_vacuum_threshold** unless otherwise specified.

- autovacuum_analyze_threshold

  (Applicable when AUTOVACUUM is enabled) Specifies the minimum number of inserted, updated, or deleted tuples needed to trigger an ANALYZE in the table.

  Value range: 0–2147483647.

  Default value: consistent with the GUC parameter **autovacuum_analyze_threshold** unless otherwise specified.

- autovacuum_vacuum_scale_factor

  (Applicable when AUTOVACUUM is enabled) Specifies the ratio of inserted, updated, or deleted tuples needed to trigger a VACUUM in the table. (This parameter applies specifically to Astore tables.)

  Value range: 0.0–100.0.

  Default value: consistent with the GUC parameter **autovacuum_vacuum_scale_factor** unless otherwise specified.

- autovacuum_analyze_scale_factor

  (Applicable when AUTOVACUUM is enabled) Specifies the ratio of inserted, updated, or deleted tuples needed to trigger an ANALYZE in the table.

  Value range: 0.0–100.0.

  Default value: consistent with the GUC parameter **autovacuum_analyze_scale_factor** unless otherwise specified.

- autovacuum_freeze_min_age

  (Applicable when AUTOVACUUM is enabled) Specifies the minimum age for rows to be frozen. Rows older than this age will be frozen.

  Value range: 0–576460752303423487.

  Default value: consistent with the GUC parameter **vacuum_freeze_min_age** unless otherwise specified.

- autovacuum_freeze_max_age

  (Applicable when AUTOVACUUM is enabled) Specifies the maximum age (in transactions) that the table's **pg_class.relfrozenxid** column can reach before a VACUUM operation is forced. Note that the system will launch AUTOVACUUM processes even when AUTOVACUUM is otherwise disabled. The VACUUM operation also allows removal of old files from the **pg_clog/** subdirectory. (This parameter applies specifically to Astore tables.)

  Value range: 100000–576460752303423487.

  Default value: consistent with the GUC parameter **autovacuum_freeze_max_age** unless otherwise specified.

- autovacuum_freeze_table_age

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

(Applicable when AUTOVACUUM is enabled) Specifies the time when the table will remain unchanged, marked as not requiring AUTOVACUUM. (This parameter applies specifically to Astore tables.)

Value range: 0–576460752303423487.

Default value: consistent with the GUC parameter **vacuum_freeze_table_age** unless otherwise specified.

- vacuum_truncate

  Specifies whether to enable the VACUUM TRUNCATE function of a table to truncate the last empty page during VACUUM.

  Value range: **on** and **off**.

  Default value: **on**.

  NOTE

  - This parameter is valid only for Astore tables and does not take effect for hash bucket tables and segment tables.
  - During the upgrade from a version that does not support this parameter to a version that supports this parameter, this parameter cannot be created or modified during the upgrade observation period.
  - The toast.**vacuum_truncate** parameter can be used to independently control whether to truncate the last empty page of a TOAST table.

- hashbucket

  Creates a hash bucket table. This parameter supports only row-store tables, including row-store range tables.

  Value range: **on** and **off**.

  Default value: **off**.

  NOTICE

  In current version, DDL operations on hash bucket tables are affected. Therefore, you are advised not to frequently perform DDL operations on hash bucket tables.

- hasrowid

  Specifies whether the table contains the **ROWID** and **ROWNO** system columns. If **hasrowid** is set to **on**, the **rowid** and **rowno** system columns are added to the table, and sequences and indexes are created on the **rowno** system column.

  Value range: **on** and **off**.

  Default value: **off**.

  NOTE

  - This parameter supports only ORA-compatible databases (that is, **sql_compatibility** is set to **ORA**).
  - This parameter does not support local temporary tables, unlogged tables, and global temporary tables.
  - This parameter cannot be specified together with **hasuids** and **oids**.
  - When this parameter is enabled, the ordinary column name cannot be defined as **rowno** or **rowid**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - partition_statistics_min_rows

    Specifies the minimum number of rows to be sampled at the partition level when the ANALYZE statement is executed on a partitioned table.

    Value range: [0,10000]

    Default value: **50**

    NOTE

    - This parameter is valid only for partitioned tables.
    - This parameter is valid only when **PARTITION_MODE** in the ANALYZE statement is set to **ALL** and the GUC parameter **enable_partition_statistics_min_rows** is enabled. For details about **PARTITION_MODE**, see **PARTITION_MODE** in **ANALYZE | ANALYSE**.
    - Upper limit: The number of rows sampled from the entire partitioned table cannot exceed the number of rows allowed by the GUC parameter **maintenance_work_mem**.

- **ON COMMIT { PRESERVE ROWS | DELETE ROWS }**

  Specifies subsequent operations on the temporary table when the temporary table is created in a transaction and the transaction is committed. Currently, only **PRESERVE ROWS** and **DELETE ROWS** are supported.

  - **PRESERVE ROWS** (default): No special action is taken at the ends of transactions. The temporary table and its table data are unchanged.
  - **DELETE ROWS**: All rows in the temporary table will be deleted at the end of each transaction block.

- **ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [MEDIUM | HIGH] ROW AFTER n { day | month | year } OF NO MODIFICATION [ON (EXPR)] | TURBO [LOW | MEDIUM | HIGH] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}**

  - When creating a row-store table, you can call ILM ADD POLICY ROW STORE COMPRESS ADVANCED [MEDIUM | HIGH] ROW to add a medium- or high-level advanced compression policy.

    - AFTER n { day | month | year } OF NO MODIFICATION: rows that have not been modified for *n* days, months, or years. Advanced compression does not support **hour**.
    - ON (EXPR): a row-level expression used to determine whether a row is hot or cold.

    NOTE

    - Among the functions supported by the row-level expression ON (EXPR) in the ILM policy, the output of some may be influenced by compatibility parameters. For instance, after you set **b_format_version** to **'5.7'** and **b_format_dev_version** to **'s2'** in MySQL-compatible mode, the upper function will be unable to convert data to uppercase as usual.
    - The compression level is optional. If not specified, the MEDIUM level will be used by default.

  - When creating a row-store table, you can call ILM ADD POLICY ROW STORE COMPRESS TURBO HIGH UNIT to add a cross-page transparent compression policy. Subpartitions will inherit this policy from partitions.

    AFTER n { hour | day | month | year } OF NO MODIFICATION: rows that have not been modified for *n* hours, days, months, or years.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

If the compression level is not specified, the MEDIUM level is used by default.

– When creating a row-store table, you can call ILM ADD POLICY ROW STORE COMPRESS NONE to explicitly specify that no cross-page transparent compression policy needs to be added.

- **TABLESPACE tablespace_name**

  Specifies the name of the tablespace where the table is to be created. If not specified, the default tablespace will be used.

- **DISTRIBUTE BY**

  For details, see **DISTRIBUTE BY** in **CREATE TABLE**.

- **TO { GROUP groupname | NODE (nodename [, ...]) }**

  Specifies the node group for which the table is to be created.

  This syntax is intended only for use by internal scale-out tools and is not recommended for direct use by users.

- **AS query**

  Specifies a **SELECT** or **VALUES** command, or an **EXECUTE** command that runs a prepared **SELECT** or **VALUES** query.

NOTICE

When data type conversion is required, especially when the CREATE TABLE AS SELECT statement is used, you are advised to use explicit type conversion to avoid unexpected results caused by implicit type conversion.

- **[WITH [NO] DATA]**

  Specifies whether to insert the queried data into the table being created. By default, the data is inserted. If you specify **NO**, the data will not be inserted.

Examples

- If no column name is specified, the columns in the new table are the same as those queried in the SELECT statement.

```
-- Create the test1 table and insert two records into the table.
gaussdb=# CREATE TABLE test1(col1 int PRIMARY KEY,col2 varchar(10));
gaussdb=# INSERT INTO test1 VALUES (1,'col1'),(101,'col101');
-- Query the data whose col1 is less than 100 in the table.
gaussdb=# SELECT * FROM test1 WHERE col1 < 100;
 col1 | col2
------+------
    1 | col1
(1 row)

-- Create the test2 table and insert the queried data into the table.
gaussdb=# CREATE TABLE test2 AS SELECT * FROM test1 WHERE col1 < 100;

-- Query the structure of the test2 table.
gaussdb=# \d test2;
 Column |         Type          | Modifiers
--------+-----------------------+-----------
 col1   | integer               |
 col2   | character varying(10) |
```

- Specify column names for the new table.

```
-- Use test1 to copy a new table test3 and specify column names.
gaussdb=# CREATE TABLE test3(c1,c2) AS SELECT * FROM test1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Query the structure of the test3 table.
gaussdb=# \d test3
           Table "public.test3"
 Column |         Type          | Modifiers
--------+-----------------------+-----------
 c1     | integer               |
 c2     | character varying(10) |

-- Drop.
gaussdb=# DROP TABLE test1,test2,test3;
```

- Create a table with the ILM policy enabled.

```
-- Create an ORA-compatible database.
gaussdb=# CREATE DATABASE ilmtabledb WITH dbcompatibility = 'ORA';
gaussdb=# \c ilmtabledb

-- Enable the ILM feature of the database.
ilmtabledb=#  ALTER DATABASE SET ILM = on;

-- Create a table ilm_table with the ILM policy enabled and insert the value of old_table.
ilmtabledb=# CREATE TABLE old_table (a int);
ilmtabledb=# CREATE TABLE ilm_table
  ILM ADD POLICY ROW STORE COMPRESS ADVANCED
  ROW AFTER 3 MONTHS OF NO MODIFICATION
  AS (SELECT * FROM old_table);

-- Drop.
ilmtabledb=# DROP TABLE old_table,ilm_table;
ilmtabledb=# \c postgres
gaussdb=# DROP DATABASE ilmtabledb;
```

Helpful Links

CREATE TABLE and **SELECT**

1.13.9.47 CREATE TABLE PARTITION

Description

Creates a partitioned table. Partitioning refers to splitting what is logically one large table into smaller physical pieces based on specific schemes. The initial table is known as a partitioned table, and the resulting pieces are called partitions. The partitioned table is a logical table, meaning it does not store data; instead, data is stored within these partitions.

Commonly used partitioning policies include range partitioning, interval partitioning, hash partitioning, list partitioning, and value partitioning. Currently, row-store tables support range partitioning, hash partitioning, and list partitioning.

- Range partitioning:

  In range partitioning, a table is partitioned based on ranges defined by one or more columns, with no overlap between the ranges of values assigned to different partitions. A partition is created for each range to store corresponding data.

  Range partitioning policy: It defines how records are inserted into partitions. Records are mapped to created partitions based on the value of the partition key. If a record is successfully mapped to a partition, it is inserted into that specific partition. If mapping is not possible, an error message will be generated. Range partitioning is the most commonly used partitioning policy.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Hash Partitioning

  A table is partitioned by specifying a modulus and a remainder for each partition. Each partition will hold the rows for which the hash value of the partition key divided by the specified modulus will produce the specified remainder.

  Hash partitioning policy: Records are mapped to created partitions based on the value of the partition key. If a record is successfully mapped to a partition, it is inserted into that specific partition. If mapping is not possible, an error message will be generated.

- List Partitioning

  List partitioning involves partitioning a table by explicitly listing which key values appear in each partition, with no overlap between the key values assigned to different partitions. A partition is created for each group of key values to store corresponding data.

  List partitioning policy: Records are mapped to created partitions based on the value of the partition key. If a record is successfully mapped to a partition, it is inserted into that specific partition. If mapping is not possible, an error message will be generated.

Partitioning can provide several benefits:

- Query performance is improved, particularly when most of the heavily accessed rows of the table are in a single partition or a small number of partitions. Partitioning narrows the range of data search and improves data access efficiency.
- When queries or updates access a large percentage of a single partition, performance can be improved by taking advantage of sequential scan of that partition instead of accessing the whole table.
- If bulk load or DELETE operations are needed on a separate partition, you can directly read or delete data from that partition. It also entirely avoids the VACUUM overload caused by a bulk DELETE. (Hash partitioning does not support partition deletion.)

Precautions

- When a UNIQUE or PRIMARY KEY constraint key includes all partition keys, a local index will be created for the constraint. Otherwise, a global index will be created.
- Currently, hash partitioning supports only single-column partition keys, and does not support multi-column partition keys.
- In the PARTITION FOR (values) syntax for partitioned tables, **values** can only be a constant.
- If data type conversion is required for **values** in the PARTITION FOR (values) syntax for partitioned tables, it is advisable to employ forcible type conversion rather than implicit type conversion, as the latter can result in unexpected outcomes.
- Each table can accommodate a maximum of 1,048,575 partitions and subpartitions in total. It is advisable to keep the number of partitions per table at 400 or less by default. For details, see "Feature Usage Guidelines > Partitioned Table > Introduction to Partitioned Tables > Basic Usage of Partitions > Creating Partitioned Tables" in *Developer Guide*.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Having too many partitions can lead to insufficient memory, which in turn results in a sharp decrease in performance.
- Data of the XML type cannot be used as partition keys or subpartition keys.
- Foreign key constraints cannot be created on the **rowid** or **rowno** system columns.
- When you perform an UPDATE or DELETE on a partitioned table and the generated plan is not an FQS or Stream plan, the efficiency of statement execution will be suboptimal. You are advised to check statements and eliminate factors that cannot be pushed down to generate FQS or Stream plans.
- If you add a row-level expression when adding or changing an ILM policy for a data object, note that the row-level expression supports only the functions listed in the whitelist. For details about the whitelist function list, see **Row Expression Function Whitelist**.

  NOTE

  Among the functions supported by the row-level expression ON (EXPR) in the ILM policy, the output of some may be influenced by compatibility parameters. For instance, after you set **b_format_version** to **'5.7'** and **b_format_dev_version** to **'s2'** in MySQL-compatible mode, the upper function will be unable to convert data to uppercase as usual.

Syntax

```
CREATE TABLE [ IF NOT EXISTS ] partition_table_name
{ ( [
   { column_name data_type [ CHARACTER SET | CHARSET charset ] [ COLLATE collation ]
[ column_constraint [ ... ] ]
   | table_constraint
   | LIKE source_table [ like_option [...] ] }
   [, ... ]
] )
   | LIKE source_table }
   [ table_option [ ,... ] [ ... ] ]
   [ htap_option ]
   [ WITH ( {storage_parameter = value} [, ... ] ) ]
   [ ILM ADD POLICY ROW STORE COMPRESS
 { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON
( EXPR )]
 | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION
 | NONE}]
   [ TABLESPACE tablespace_name ]
   [ DISTRIBUTE BY { REPLICATION | HASH ( column_name [, ...] )
   | MURMURHASH ( diskey_expr )
   | RANGE ( column_name [, ...] ) { SLICE REFERENCES tablename | ( slice_less_than_item [, ...] )
   | ( slice_start_end_item [, ...] ) }
   | LIST ( column_name [, ...] ) { SLICE REFERENCES tablename | ( slice_values_item [, ...] ) }
   } ]
   [ TO { GROUP groupname | NODE ( nodename [, ... ] ) } ]
   PARTITION BY {
      {RANGE [COLUMNS] (partition_key) [ PARTITIONS integer ] ( partition_less_than_item [, ... ] )} |
      {RANGE [COLUMNS] (partition_key) [ PARTITIONS integer ] ( partition_start_end_item [, ... ] )} |
      {LIST [COLUMNS] (partition_key) [ PARTITIONS integer ] ( PARTITION partition_name VALUES [IN]
(list_values) [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n
{ day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT
AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [ htap_option ][TABLESPACE [=]
tablespace_name][, ... ])} |
      { HASH (partition_key) [ PARTITIONS integer ] ( PARTITION partition_name [ ILM ADD POLICY ROW
STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO
MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month |
year } OF NO MODIFICATION | NONE}] [ htap_option ][TABLESPACE [=] tablespace_name][, ... ])}
   } [ { ENABLE | DISABLE } ROW MOVEMENT ];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **table_option**:

```
{ COMMENT [ = ] 'string' |
  AUTO_INCREMENT [ = ] value |
  [ DEFAULT ] CHARACTER SET | CHARSET [ = ] default_charset |
  [ DEFAULT ] COLLATE [ = ] default_collation }
```

- **htap_option**:

```
{ COLVIEW [ PRIORITY { HIGH | LOW | NONE } ] |
  NOCOLVIEW }
```

- **column_constraint**:

```
[ CONSTRAINT constraint_name ]
{ NOT NULL |
  NULL |
  CHECK ( expression ) |
  AUTO_INCREMENT |
  COMMENT 'string' |
  DEFAULT default_expr |
  ON UPDATE update_expr |
  UNIQUE [KEY] [ index_parameters ] |
  PRIMARY KEY [ index_parameters] }
[ DEFERRABLE | NOT DEFERRABLE ][ INITIALLY DEFERRED | INITIALLY IMMEDIATE ]
```

- **table_constraint**:

```
[ CONSTRAINT constraint_name ]
{ CHECK ( expression ) |
  UNIQUE [ index_name ] [ USING method ] ( { column_name [ ASC | DESC ] } [, ... ] )
[ index_parameters ] |
  PRIMARY KEY ( column_name [, ... ] ) [ index_parameters]}
[ DEFERRABLE | NOT DEFERRABLE ][ INITIALLY DEFERRED | INITIALLY IMMEDIATE ]
{ [ COMMENT 'string' ] [ ... ] }
```

- **like_option**:

```
{ INCLUDING | EXCLUDING } { DEFAULTS | CONSTRAINTS | INDEXES | STORAGE | COMMENTS |
RELOPTIONS | DISTRIBUTION | UPDATE | ILM | ALL }
```

- **index_parameters**:

```
[ WITH ( {storage_parameter = value} [, ... ] ) ]
[ USING INDEX TABLESPACE tablespace_name ]
```

- **partition_less_than_item**:

```
PARTITION partition_name VALUES LESS THAN { ( { partition_value | MAXVALUE }[, ... ] ) |
MAXVALUE } [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW
AFTER n { day | month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM |
HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [TABLESPACE
tablespace_name]
```

- **partition_start_end_item**:

```
PARTITION partition_name {
      {START(partition_value) END (partition_value) EVERY (interval_value)} |
      {START(partition_value) END ({partition_value | MAXVALUE})} |
      {START(partition_value)} |
      {END({partition_value | MAXVALUE})}
} [ ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day |
month | year } OF NO MODIFICATION [ ON ( EXPR )] | TURBO [ LOW | MEDIUM | HIGH ] UNIT
AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}] [TABLESPACE
tablespace_name]
```

- **update_expr**:

```
{ CURRENT_TIMESTAMP | LOCALTIMESTAMP | NOW() }
```

Parameters

- **IF NOT EXISTS**

  Specifies how to handle name conflicts with tables.

  If you specify the IF NOT EXISTS keyword and the system detects a table with the same name, it will not issue an error during table creation; instead, it will return a NOTICE indicating that the table already exists.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **partition_table_name**

  Specifies the name of a partitioned table.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **column_name**

  Specifies the name of a column to be created in the new table.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **data_type**

  Specifies the data type of the column.

- **COLLATE collation**

  Assigns a collation to the column (which must be of a collatable data type). If not specified, the default collation is used. You can run the **SELECT * FROM pg_collation;** command to query collations from the pg_collation system catalog. The row starting with **default** in the query result represents the default collation.

- **CONSTRAINT constraint_name**

  Specifies the name of a column constraint or table constraint. The optional CONSTRAINT clause specifies constraints that new or updated rows must satisfy for an INSERT or UPDATE operation to succeed. You can run the **SELECT * FROM pg_collation** command to query collations from the pg_collation system catalog. The row starting with **default** in the query result represents the default collation.

  There are two ways to define constraints:

  – A column constraint is defined as part of a column definition, and it is bound to a particular column. You cannot add a constraint name to the clause AUTO_INCREMENT | COMMENT 'string' | ENCRYPTED WITH (COLUMN_ENCRYPTION_KEY = column_encryption_key, ENCRYPTION_TYPE = encryption_type_value).

  – A table constraint is not tied to a particular column, and it can encompass more than one column.

- **LIKE source_table [like_option ...]**

  Specifies a table, known as the source table, from which the target table automatically inherits all column names, their data types, and NOT NULL constraints.

  The target table and source table are completely decoupled after creation is complete. Changes to the source table will not be applied to the target table, and it is not possible to include data of the target table in scans of the source table.

  – The default expressions of columns are included only when **INCLUDING DEFAULTS** is specified. The default behavior is to exclude the default expressions, causing all columns in the target table to default to **NULL**.

  – The target table always inherits the NOT NULL constraints from the source table. The CHECK constraints are inherited only when **INCLUDING CONSTRAINTS** is specified. Other types of constraints are not inherited. This rule also applies to table constraints and column constraints.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The system does not merge inherited columns or constraints with those that share the same name. If the same name is specified explicitly or in another LIKE clause, an error is signaled.

- If **INCLUDING UPDATE** is specified, columns in the target table will inherit the **ON UPDATE CURRENT_TIMESTAMP** attribute from the source table. The default behavior is to exclude this attribute.
- If **INCLUDING INDEXES** is specified, indexes on the source table will also be created on the target table. By default, no indexes are created. The tablespace where indexes on the target table are located is the same as that where indexes on the source table are located.
- If **INCLUDING STORAGE** is specified, columns in the target table will inherit **STORAGE** settings from the source table. The default behavior is to exclude **STORAGE** settings.
- If **INCLUDING COMMENTS** is specified, the target table will inherit comments on columns, constraints, and indexes from the source table. The default behavior is to exclude comments.
- If **INCLUDING RELOPTIONS** is specified, the target table will inherit storage parameters (WITH clause) from the source table. The default behavior is to exclude storage parameters.
- If **INCLUDING DISTRIBUTION** is specified, the target table will inherit distribution information (including distribution types and keys) from the source table and will no longer be able to use the DISTRIBUTE BY clause. The default behavior is to exclude distribution information.
- If **INCLUDING ILM** is specified, the target table will inherit ILM policy information from the source table. To also inherit the ILM policy information of partition objects, you will need to specify **INCLUDING PARTITION**.
- **INCLUDING ALL** is a condensed version that encompasses **INCLUDING DEFAULTS**, **INCLUDING UPDATE**, **INCLUDING CONSTRAINTS**, **INCLUDING INDEXES**, **INCLUDING STORAGE**, **INCLUDING COMMENTS**, **INCLUDING RELOPTIONS**, **INCLUDING DISTRIBUTION**, and **INCLUDING ILM**.

NOTICE

- If the source table is a ROWID table and INCLUDING ALL or INCLUDING RELOPTIONS is specified, the new table is also a ROWID table. If only INCLUDING INDEXES is specified, the new table is a non-ROWID table and the **ROWNO** index of the source table is not copied.
- **CREATE TABLE table_name LIKE source_table;** is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**) where **b_format_version** is **5.7** and **b_format_dev_version** is **s2**.
- In MySQL-compatible databases where **b_format_version** is **5.7** and **b_format_dev_version** is **s2**, the **INCLUDING** and **EXCLUDING** options cannot be specified. The default setup is equivalent to specifying **INCLUDING ALL**.

- **AUTO_INCREMENT [=] value**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies an initial value for an auto-increment column. The value must be a positive number no greater than $2^{127} - 1$.

NOTICE

This clause takes effect only when **sql_compatibility** is set to **'MYSQL'**.

- **COMMENT [=] 'string'**
  - COMMENT [=] 'string: adds a comment on the table.
  - COMMENT 'string' in **column_constraint**: adds a comment on a column.
  - COMMENT 'string' in **table_constraint**: adds a comment on the indexes corresponding to the primary key and unique key.

  For details, see **COMMENT [=] 'string'** in **CREATE TABLE**.
- **CHARACTER SET | CHARSET charset**

  Assigns a character set to the column. If a character set is explicitly specified, the column will use the default collation associated with that character set.
- **COLLATE collation**

  Assigns a collation to the column (which must be of a collatable data type). If not specified, the default collation is used. You can run the **SELECT * FROM pg_collation** command to query collations from the pg_collation system catalog. The row starting with **default** in the query result represents the default collation. MySQL-compatible databases (**sql_compatibility = 'MYSQL'**) also support the utf8mb4_bin, utf8mb4_general_ci, utf8mb4_unicode_ci, binary, gbk_chinese_ci, gbk_bin, gb18030_chinese_ci, and gb18030_bin collations.
- **WITH (storage_parameter [= value] [, ...])**

  Specifies an optional storage parameter for the table or index. See details of parameters below.

  NOTE

  These TDE-related storage parameters are supported: **enable_tde**, **encrypt_algo**, **dek_cipher**, **key_type**, and **cmk_id**.

  For details about the parameters, see **CREATE TABLE**.

  - FILLFACTOR

    Specifies a fillfactor for the table. It is a number between 10 and 100. **100** (complete packing) is the default. When you specify a smaller fillfactor, INSERT operations pack table pages only to the indicated percentage. The remaining space on each page is reserved for updating rows on that page. This gives UPDATE a chance to place the updated copy of a row on the same page as the original, which is more efficient than placing it on a different page. For a table whose entries are never updated, complete filling (fillfactor = 100) is the best choice. However, in heavily updated tables, smaller fillfactors are appropriate.

    Value range: 10–100.
  - ORIENTATION

    Specifies the orientation for storing table data.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: **ROW** (default value), indicating that table data is stored in row orientation.

NOTICE

ORIENTATION cannot be modified.

- STORAGE_TYPE

  Specifies the storage engine type. Once set, this parameter cannot be modified.

  Value range:

  - **USTORE**: The table uses an in-place update storage engine. To prevent space bloat, be sure to enable the **track_counts** and **track_activities** parameters for Ustore tables.
  - **ASTORE**: The table uses an append-only storage engine.

  Default value:

  Subject to the **enable_default_ustore_table** parameter, which defaults to in-place update storage.

- statistic_granularity

  Records the default **partition_mode** when the table analyzes statistics. For details about **partition_mode**, see **PARTITION_MODE** in **ANALYZE | ANALYSE**. This parameter setting does not work for non-partitioned tables.

  Value range: See **PARTITION_MODE** in **ANALYZE | ANALYSE**.

  Default value: **AUTO**.

- autovacuum_enabled

  Specifies whether to enable AUTOVACUUM for the table.

  Value range: **on** and **off**.

  Default value: **on**.

- autovacuum_vacuum_threshold

  (Applicable when AUTOVACUUM is enabled) Specifies the minimum number of updated or deleted tuples needed to trigger a VACUUM in the table. (This parameter applies specifically to Astore tables.)

  Value range: 0–2147483647.

  Default value: consistent with the GUC parameter **autovacuum_vacuum_threshold**.

- autovacuum_analyze_threshold

  (Applicable when AUTOVACUUM is enabled) Specifies the minimum number of inserted, updated, or deleted tuples needed to trigger an ANALYZE in the table.

  Value range: 0–2147483647.

  Default value: consistent with the GUC parameter **autovacuum_analyze_threshold**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- autovacuum_vacuum_scale_factor

  (Applicable when AUTOVACUUM is enabled) Specifies the ratio of inserted, updated, or deleted tuples needed to trigger a VACUUM in the table. (This parameter applies specifically to Astore tables.)

  Value range: 0.0–100.0.

  Default value: consistent with the GUC parameter **autovacuum_vacuum_scale_factor**.

- autovacuum_analyze_scale_factor

  (Applicable when AUTOVACUUM is enabled) Specifies the ratio of inserted, updated, or deleted tuples needed to trigger an ANALYZE in the table.

  Value range: 0.0–100.0.

  Default value: consistent with the GUC parameter **autovacuum_analyze_scale_factor**.

- autovacuum_freeze_min_age

  (Applicable when AUTOVACUUM is enabled) Specifies the minimum age of row versions. Rows older than this age will be frozen.

  Value range: 0–576460752303423487.

  Default value: consistent with the GUC parameter **vacuum_freeze_min_age**.

- autovacuum_freeze_max_age

  (Applicable when AUTOVACUUM is enabled) Specifies the maximum age (in transactions) that the table's **pg_class.relfrozenxid** field can reach before a VACUUM operation is forced. Note that the system will launch AUTOVACUUM processes even when AUTOVACUUM is otherwise disabled. The VACUUM operation also allows removal of old files from the **pg_clog/** subdirectory. (This parameter applies specifically to Astore tables.)

  Value range: 100000–576460752303423487.

  Default value: consistent with the GUC parameter **autovacuum_freeze_max_age**.

- autovacuum_freeze_table_age

  (Applicable when AUTOVACUUM is enabled) Specifies the time when the table will remain unchanged, marked as not requiring AUTOVACUUM. (This parameter applies specifically to Astore tables.)

  Value range: 0–576460752303423487.

  Default value: consistent with the GUC parameter **vacuum_freeze_table_age**.

- vacuum_truncate

  Specifies whether to enable the VACUUM TRUNCATE function of a table to truncate the last empty page during VACUUM.

  Value range: **on** and **off**.

  Default value: **on**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This parameter is valid only for Astore tables and does not take effect for hash bucket tables and segment tables.
- During the upgrade from a version that does not support this parameter to a version that supports this parameter, this parameter cannot be created or modified during the upgrade observation period.
- The toast.**vacuum_truncate** parameter can be used to independently control whether to truncate the last empty page of a TOAST table.

– hashbucket

Creates a hash bucket table. This parameter supports only row-store tables, including row-store range tables.

Value range: **on** and **off**.

Default value: **off**.

NOTICE

In current version, DDL operations on hash bucket tables are affected. Therefore, you are advised not to frequently perform DDL operations on hash bucket tables.

– hasrowid

Specifies whether the table is a rowid table. If **hasrowid** is set to **on**, the **rowid** and **rowno** system columns are added to the table, and sequences and indexes are created on the **rowno** system column.

Value range: **on** and **off**.

Default value: **off**.

NOTE

- This parameter supports only ORA-compatible databases (that is, **sql_compatibility** is set to **ORA**).
- This parameter does not support local temporary tables, unlogged tables, global temporary tables, materialized views, hash bucket tables, or replication tables.
- This parameter cannot be specified together with **hasuids** and **oids**.
- When this parameter is enabled, the ordinary column name cannot be defined as **rowno** or **rowid**.
- The number of data records in the **rowid** table cannot exceed 9223372036854775807.

– partition_statistics_min_rows

Specifies the minimum number of rows to be sampled at the partition level when the ANALYZE statement is executed on a partitioned table.

Value range: [0,10000]

Default value: **50**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This parameter is valid only for partitioned tables.
- This parameter is valid only when **PARTITION_MODE** in the ANALYZE statement is set to **ALL** and the GUC parameter **enable_partition_statistics_min_rows** is enabled. For details about **PARTITION_MODE**, see **PARTITION_MODE** in **ANALYZE | ANALYSE**.
- Upper limit: The number of rows sampled from the entire partitioned table cannot exceed the number of rows allowed by the GUC parameter **maintenance_work_mem**.

- **[ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [MEDIUM | HIGH] ROW AFTER n { day | month | year } OF NO MODIFICATION [ON (EXPR)] | TURBO [LOW | MEDIUM | HIGH] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}]**
  - When creating a table, you can call ILM ADD POLICY ROW STORE COMPRESS ADVANCED [MEDIUM | HIGH] ROW to add a medium- or high-level advanced compression policy for row store. A policy can be separately specified for a partition. If no policy is specified, partitions inherit the policy of the table.
    - AFTER n { day | month | year } OF NO MODIFICATION: rows that have not been modified for *n* days, months, or years. Advanced compression does not support **hour**.
    - ON (EXPR): a row-level expression used to determine whether a row is hot or cold.

    NOTE

    - Among the functions supported by the row-level expression ON (EXPR) in the ILM policy, the output of some may be influenced by compatibility parameters. For instance, after you set **b_format_version** to **'5.7'** and **b_format_dev_version** to **'s2'** in MySQL-compatible mode, the upper function will be unable to convert data to uppercase as usual.
    - The compression level is optional. If not specified, the MEDIUM level will be used by default.
  - When creating a row-store table, you can call ILM ADD POLICY ROW STORE COMPRESS TURBO HIGH UNIT to add a cross-page transparent compression policy. Subpartitions will inherit this policy from partitions.

    AFTER n { hour | day | month | year } OF NO MODIFICATION indicates that rows have not been modified for *n* hours, days, months, or years.

    If the compression level is not specified, the MEDIUM level is used by default.
  - When creating a row-store table, you can call ILM ADD POLICY ROW STORE COMPRESS NONE to explicitly specify that no cross-page transparent compression policy needs to be added.
- **TABLESPACE tablespace_name**

  Specifies the name of the tablespace where the table is to be created. If not specified, the default tablespace will be used.
- **DISTRIBUTE BY**

  Specifies how the table is distributed or replicated between nodes.

  For details, see **DISTRIBUTE BY** in **CREATE TABLE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **TO { GROUP groupname | NODE (nodename [, ...]) }**

  Specifies the node group for which the table is to be created. **TO NODE** is intended only for use by internal scale-out tools and is not recommended for direct use by users.

- **PARTITION BY RANGE [COLUMNS] (partition_key)**

  Creates a range partition. **partition_key** indicates the name of the partition key.

  NOTE

  The COLUMNS keyword is available only when **sql_compatibility** is **'MYSQL'**. PARTITION BY RANGE COLUMNS is semantically equivalent to PARTITION BY RANGE.

  – If the VALUES LESS THAN clause is present in the syntax:

    The range partitioning policy permits using up to 16 columns as the partition key.

    The partition key accepts the following data types:

    - Integer types: TINYINT, SMALLINT, INTEGER, BIGINT, DECIMAL, and NUMERIC
    - Float types: REAL, FLOAT4, FLOAT8, and DOUBLE PRECISION
    - Character types: CHARACTER VARYING, VARCHAR, CHARACTER, CHAR, BPCHAR, TEXT, NVARCHAR2, and NAME
    - Date/Time types: TIMESTAMP[(p)] [WITHOUT TIME ZONE], TIMESTAMP[(p)] [WITH TIME ZONE], YEAR, DATE, and DATEA (valid only in ORA-compatible mode)

  – If the START END clause is present in the syntax:

    The range partitioning policy only permits using a single column as the partition key.

    The partition key accepts the following data types:

    - Integer types: TINYINT, SMALLINT, INTEGER, BIGINT, DECIMAL, and NUMERIC
    - Float types: REAL, FLOAT4, FLOAT8, and DOUBLE PRECISION
    - Date/Time types: TIMESTAMP[(p)] [WITHOUT TIME ZONE], DATE, and DATEA (valid only in ORA-compatible mode)

- **PARTITION partition_name VALUES LESS THAN {({ partition_value | MAXVALUE } [,...]) | MAXVALUE }**

  Specifies the partition information. **partition_name** indicates the name of a range partition. **partition_value** indicates the upper limit of the range partition. The limit varies depending on the type of **partition_key**. **MAXVALUE** indicates the maximum value, which is typically the upper limit of the last range partition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- Each partition requires an upper limit.
- The data type of the upper limit must be the same as that of the partition key.
- In a partition list, partitions are arranged in ascending order of upper limits. A partition with a smaller upper limit is placed before another partition with a larger one.
- MAXVALUE that is not in parentheses can be used only when **sql_compatibility** is set to **'MYSQL'** and can have only one partition key.

- **PARTITION partition_name {START (partition_value) END (partition_value) EVERY (interval_value)} | {START (partition_value) END (partition_value|MAXVALUE)} | {START(partition_value)} | {END (partition_value | MAXVALUE)}**

  Specifies the partition information, including:

  - **partition_name**: name or prefix of the range partition. It specifies the partition name, except in the following cases (assuming **partition_name** is **p1**):

    - If this definition is a START+END+EVERY clause, it semantically defines partition names as **p1_1**, **p1_2**, and so forth. For example, using **PARTITION p1 START(1) END(4) EVERY(1)** will generate partitions [1, 2), [2, 3), and [3, 4), named **p1_1**, **p1_2**, and **p1_3**, respectively. Here, **p1** acts as a name prefix.

    - If this is the first partition definition and it includes a **START** value, the (**MINVALUE**, **START**) range will be treated as the first partition, named **p1_0**. Subsequent partitions will be named **p1_1**, **p1_2**, and so forth. For example, using **PARTITION p1 START(1), PARTITION p2 START(2)** will generate partitions (**MINVALUE**, 1), [1, 2), and [2, **MAXVALUE**), named **p1_0**, **p1_1**, and **p2**, respectively. Here, **p1** acts as a name prefix, and **p2** is a partition name. **MINVALUE** indicates the minimum value.

  - **partition_value**: endpoint value (start or end) of the range partition. The value depends on the type of **partition_key** and must not be equal to **MAXVALUE**.
  - **interval_value**: width of each partition generated within the [**START**, **END**) range. It must not be equal to **MAXVALUE**. If the difference between **END** and **START** is not divisible by **EVERY**, only the width of the last partition will be less than **EVERY**.
  - **MAXVALUE**: maximum upper limit, which is typically the upper limit of the last range partition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

1. During the creation of a partitioned table, if the first partition definition includes a **START** value, the (**MINVALUE**, **START**) range will be treated as the first partition.
2. The START END syntax must comply with the following rules:
   - Given a specific **partition_start_end_item**, the value of **START** (if specified) must be less than the value of **END**.
   - Given two consecutive **partition_start_end_item** instances, the value of the first **END** must be equal to the value of the second **START**.
   - For **partition_start_end_item**, the value of **EVERY** must increment in a positive direction and be less than **END** – **START**.
   - Each partition includes its start value but excludes its end value, following the format [**START**, **END**). If the start value is **MINVALUE**, it will also be excluded.
   - Partitions created by the same **partition_start_end_item** belong to the same tablespace.
   - When **partition_name** serves as the prefix of partition names, its length must not exceed 57 bytes; otherwise, it will be automatically truncated.
   - When creating or modifying partitioned tables, keep in mind that the total number of partitions in the tables must not exceed the maximum limit of 1,048,575.
3. Do not mix the START END and LESS THAN syntaxes when creating partitioned tables.
4. Even if a partitioned table is created with the START END syntax, the SQL statement backed up (by gs_dump) will still use the VALUES LESS THAN syntax.

- **PARTITION BY LIST [COLUMNS] (partition_key)**

  Creates a list partition, with **partition_key** indicating the name of the partition key.

  The COLUMNS keyword can be used only when **sql_compatibility** is set to **'MYSQL'**. The semantics of PARTITION BY LIST COLUMNS is the same as that of PARTITION BY LIST.

  – The list partitioning policy allows up to 16 columns as **partition_key**.
  – In the presence of the VALUES [IN] (list_values) clause, **list_values** contains the key values of partitions. Each partition can have a maximum of 64 key values.
  – Semantically equivalent to VALUES, the VALUES IN clause is available only when **sql_compatibility** is **'MYSQL'**.

  The partition key accepts the following data types:

  – Integer types: TINYINT, SMALLINT, INTEGER, BIGINT, and NUMERIC
  – Character types: VARCHAR, CHAR, BPCHAR, and NVARCHAR2
  – Date/Time types: TIMESTAMP[(p)] [WITHOUT TIME ZONE], TIMESTAMP[(p)] [WITH TIME ZONE], YEAR, DATE, and DATEA (valid only in ORA-compatible mode)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The number of partitions cannot exceed 1,048,575.

- **PARTITION BY HASH(partition_key)**

  Creates a hash partition, with **partition_key** indicating the name of the partition key.

  The hash partitioning policy only permits using a single column as **partition_key**.

  The partition key accepts the following data types:

  – Integer types: TINYINT, SMALLINT, INTEGER, BIGINT, and NUMERIC

  – Character types: VARCHAR, CHAR, BPCHAR, TEXT, and NVARCHAR2

  – Date/Time types: TIME, TIMESTAMP[(p)] [WITHOUT TIME ZONE], TIMESTAMP[(p)] [WITH TIME ZONE], YEAR, DATE, DATEA (valid only in ORA mode)

  The number of partitions cannot exceed 1,048,575.

- **PARTITIONS integer**

  Specifies the number of partitions.

  integer must be an integer within (0,1048575].

  – When this clause is specified after range or list partitions, each partition must be explicitly defined, and the number of defined partitions must be equal to **integer**. This clause can be specified only when **sql_compatibility** is **'MYSQL'**, and it must come after range or list partitions.

  – When this clause is specified after hash/key partitions, the following scenarios may occur: If you do not list the definition of each partition, the system will automatically generate *integer* partitions, named in **p**+*number* format, where *number* ranges from 0 to (**integer** – 1). These partitions will be placed within the tablespace of the table by default. If you explicitly list the partition definitions, the number of defined partitions must be equal to **integer**. If you neither list the partition definitions nor specify the number of partitions, the system will create only one partition.

- **{ ENABLE | DISABLE } ROW MOVEMENT**

  Specifies whether to enable row movement.

  If an UPDATE operation changes the value of a tuple on the partition key, causing the tuple to move to a different partition, an error will occur based on the setting you have chosen, or the tuple will be transferred between partitions.

  Value range:

  – **ENABLE**: Row movement is enabled.

  – **DISABLE** (default): Row movement is disabled.

  If row movement is enabled, concurrent UPDATE and DELETE operations may result in an error. Possible causes include:

  Both UPDATE and DELETE operations mark old data as deleted. In addition, if changing the partition key leads to cross-partition updates, the kernel will mark the old data in the original partition as deleted and generate a new record in a different partition, making it impossible to locate the new data based on the old data.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

If concurrent UPDATE+UPDATE, DELETE+DELETE, or UPDATE+DELETE operations are performed on the same row, the behavior differs based on whether the data is distributed across partitions.

a. When data is not distributed across partitions, performing the second operation after the first operation will not result in an error.

   If the first operation is UPDATE, the most recent data can be found and accessed during the second operation.

   If the first operation is DELETE, the system will terminate the second operation because it detects that the current data has been deleted but cannot match it with the most recent data.

b. When data is distributed across partitions, performing the second operation after the first operation will result in an error.

   If the first operation is UPDATE, the system will report an error during the second operation because it cannot find the most recent data, which is stored in a different partition.

   If the first operation is DELETE, the system can detect that the current data has been deleted during the second operation, but cannot match it with the most recent data or determine whether the data was deleted by an UPDATE or DELETE operation. If the operation is UPDATE, an error is reported. If the operation is DELETE, the operation is terminated. However, to ensure data correctness, it can only report an error.

In UPDATE+UPDATE and UPDATE+DELETE concurrency scenarios, you can solve the problem by performing these operations sequentially. In the DELETE +DELETE concurrency scenario, you can solve the problem by disabling row movement.

- **NOT NULL**

  Forbids NULL values in columns. **ENABLE** is provided solely for syntax compatibility and can be omitted.

- **NULL**

  Allows NULL values in columns. This is the default.

  This clause is provided solely for compatibility with non-standard SQL databases and is not recommended for use.

- **CHECK (condition) [NO INHERIT]**

  Specifies an expression producing a Boolean result which new or updated rows must satisfy for an INSERT or UPDATE operation to succeed. Expressions evaluating to TRUE or UNKNOWN succeed. Otherwise, an error exception is raised, and the INSERT or UPDATE does not alter the database.

  A CHECK constraint specified as a column constraint should reference that column's value only, while an expression appearing in a table constraint can reference multiple columns.

  A constraint marked with **NO INHERIT** will not propagate to child tables.

  ENABLE is provided solely for syntax compatibility and can be omitted.

- **DEFAULT default_expr**

  Assigns a default value to a column. The value can be any variable-free expression (in particular, subqueries and cross-references to other columns in the current table are not allowed). The data type of the default expression must match the data type of the column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The default expression can be used in any INSERT operation that does not specify a value for the column. If no default value is specified, **NULL** will be used.

- **ON UPDATE update_expr**

  Specifies an attribute constraint for the column.

  During an UPDATE operation on a tuple in a table, if the new values of the updated columns differ from the old values, columns (in the tuple) that have this attribute but are not part of the UPDATE will be automatically updated to the current timestamp. However, if the new values of the updated columns are the same as the old values, those columns will remain unchanged. If columns with this attribute are included in the UPDATE, their values will be updated to the specified value.

  NOTE

  - This attribute can be specified only in MySQL-compatible databases version 5.7 (**sql_compatibility = 'MYSQL'**, **b_format_version='5.7'**, and **b_format_dev_version='s1'**).
  - In terms of syntax, **update_expr** supports three keywords: CURRENT_TIMESTAMP, LOCALTIMESTAMP, and NOW(). You can also optionally specify the precision of a keyword with parentheses. For example, ON UPDATE CURRENT_TIMESTAMP(), ON UPDATE CURRENT_TIMESTAMP(5), ON UPDATE LOCALTIMESTAMP(), and ON UPDATE LOCALTIMESTAMP(6). If the keyword does not contain parentheses or contains empty parentheses, the precision is 0. The NOW keyword cannot contain parentheses. The three types of keywords are synonyms of each other and have the same attribute effect.
  - This attribute can be specified only for columns of the following types: timestamp, datetime, date, time without time zone, smalldatetime, and abstime.
  - If the CREATE TABLE AS syntax is used, the column attributes will not be inherited.
  - The CREATE TABLE LIKE syntax can use **INCLUDING UPDATE** or **EXCLUDING UPDATE** to inherit or exclude a constraint. The **INCLUDING ILM** option is added to copy ILM policy information from the source table. When combined with the **INCLUDING PARTITION** option, it can also copy the policy information of partition objects from the source table.
  - The precision specified by this attribute can be different from the precision specified by the type in the corresponding column. After the column value is updated through this attribute, the minimum precision is displayed. For example, **CREATE TABLE t1 (col1 timestamp(6) ON UPDATE CURRENT_TIMESTAMP(3));**. If the UPDATE syntax triggers the attribute to take effect, three decimal places in the value of **col1** are displayed after the update.
  - The same column cannot be specified for this attribute and the generated column constraint at the same time.
  - This attribute cannot be specified for the partition key in a partitioned table.
  - In distributed scenarios, this attribute cannot be specified for distribution keys or the primary key.

- **AUTO_INCREMENT**

  Marks a column as auto-increment.

  For details, see **AUTO_INCREMENT** in **CREATE TABLE**.

- **UNIQUE [KEY] index_parameters**

  UNIQUE (column_name [, ...]) index_parameters

  Specifies that a column or a combination of columns must be unique across the entire table.

  For the purpose of a UNIQUE constraint, NULL is not considered equal.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Semantically equivalent to UNIQUE, UNIQUE KEY is available only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).

- **PRIMARY KEY index_parameters**

  PRIMARY KEY (column_name [, ...]) index_parameters

  Specifies that one or more columns in the table require unique, non-NULL values.

  Only one primary key can be specified for a table.

- **DEFERRABLE | NOT DEFERRABLE**

  Specifies whether the constraint can be deferred. A constraint that is not deferrable will be checked immediately after every command. Checking of constraints that are deferrable can be postponed until the end of the transaction (using the **SET CONSTRAINTS** command). **NOT DEFERRABLE** is the default. Currently, only the UNIQUE and PRIMARY KEY constraints accept this clause. (That is, other constraints are not deferrable.)

- **INITIALLY IMMEDIATE | INITIALLY DEFERRED**

  Specifies the default time to check the constraint if it is deferrable. A constraint that is not deferrable is always **INITIALLY IMMEDIATE** and cannot be declared as **INITIALLY DEFERRED**.

  – If the constraint is **INITIALLY IMMEDIATE**, it is checked after each statement. This is the default.

  – If the constraint is **INITIALLY DEFERRED**, it is checked only at the end of the transaction.

  The constraint check time can be altered with the **SET CONSTRAINTS** command.

- **USING INDEX TABLESPACE tablespace_name**

  Specifies a tablespace in which an index associated with a UNIQUE or PRIMARY KEY constraint will be created. If this clause is not specified, the index will be created in the default tablespace named **default_tablespace**. If **default_tablespace** is empty, the default tablespace of the database will be consulted.

Range Partitioning

- VALUES LESS THAN

```
-- Create a tablespace.
CREATE TABLESPACE tbs_test_range1_p1 RELATIVE LOCATION 'tbs_test_range1/tablespace_1';
CREATE TABLESPACE tbs_test_range1_p2 RELATIVE LOCATION 'tbs_test_range1/tablespace_2';
CREATE TABLESPACE tbs_test_range1_p3 RELATIVE LOCATION 'tbs_test_range1/tablespace_3';
CREATE TABLESPACE tbs_test_range1_p4 RELATIVE LOCATION 'tbs_test_range1/tablespace_4';

-- Create a partitioned table test_range1.
CREATE TABLE test_range1(
    id INT,
    info VARCHAR(20)
) PARTITION BY RANGE (id) (
    PARTITION p1 VALUES LESS THAN (200) TABLESPACE tbs_test_range1_p1,
    PARTITION p2 VALUES LESS THAN (400) TABLESPACE tbs_test_range1_p2,
    PARTITION p3 VALUES LESS THAN (600) TABLESPACE tbs_test_range1_p3,
    PARTITION pmax VALUES LESS THAN (MAXVALUE) TABLESPACE tbs_test_range1_p4
);

-- Insert 1,000 data records.
INSERT INTO test_range1 VALUES(GENERATE_SERIES(1,1000),'abcd');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Check that the number of rows in the p1 partition is 199, that is, the record range is [1, 200).
SELECT COUNT(*) FROM test_range1 PARTITION (p1);
 count
-------
   199
(1 row)

-- Check that the number of rows in the p2 partition is 200, that is, the record range is [200, 400).
SELECT COUNT(*) FROM test_range1 PARTITION (p2);
 count
-------
   200
(1 row)

-- View the partition information.
SELECT a.relname, a.boundaries, b.spcname
FROM pg_partition a, pg_tablespace b
WHERE a.reltablespace = b.oid AND a.parentid = 'test_range1'::regclass;
 relname | boundaries |      spcname
---------+------------+--------------------
 p1      | {200}      | tbs_test_range1_p1
 p2      | {400}      | tbs_test_range1_p2
 p3      | {600}      | tbs_test_range1_p3
 pmax    | {NULL}     | tbs_test_range1_p4
(4 rows)

-- Drop.
DROP TABLE test_range1;
DROP TABLESPACE tbs_test_range1_p1;
DROP TABLESPACE tbs_test_range1_p2;
DROP TABLESPACE tbs_test_range1_p3;
DROP TABLESPACE tbs_test_range1_p4;
```

- START END

```
-- Create a partitioned table.
CREATE TABLE test_range2(
    id INT,
    info VARCHAR(20)
) PARTITION BY RANGE (id) (
    PARTITION p1 START(1) END(600) EVERY(200),
    PARTITION p2 START(600) END(800),
    PARTITION pmax START(800) END(MAXVALUE)
);

-- View the partition information.
SELECT relname, boundaries FROM pg_partition WHERE parentid = 'test_range2'::regclass AND
parttype = 'p' ORDER BY 1;
 relname | boundaries
---------+------------
 p1_0    | {1}
 p1_1    | {201}
 p1_2    | {401}
 p1_3    | {600}
 p2      | {800}
 pmax    | {NULL}
(6 rows)

-- Drop.
DROP TABLE test_range2;
```

List Partitioning

```
-- Create a list partitioned table.
CREATE TABLE test_list ( NAME VARCHAR ( 50 ), area VARCHAR ( 50 ) )
PARTITION BY LIST (area) (
    PARTITION p1 VALUES ('Beijing'),
    PARTITION p2 VALUES ('Shanghai'),
    PARTITION p3 VALUES ('Guangzhou'),
    PARTITION p4 VALUES ('Shenzhen'),
    PARTITION pdefault VALUES (DEFAULT)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
);

-- Insert data.
INSERT INTO test_list VALUES ('bob', 'Shanghai'),('scott', 'Sichuan');

-- Query partition data.
SELECT * FROM test_list PARTITION (p2);
 name |   area
------+----------
 bob  | Shanghai
(1 row)
SELECT * FROM test_list PARTITION (pdefault);
 name  |  area
-------+---------
 scott | Sichuan
(1 row)

-- Drop.
DROP TABLE test_list;
```

Hash Partitioning

```
-- Create a hash partitioned table and specify the number of partitions.
CREATE TABLE test_hash1(c1 int) PARTITION BY HASH(c1) PARTITIONS 3;

-- Create a hash partitioned table and specify the names of partitions.
CREATE TABLE test_hash2(c1 int) PARTITION BY HASH(C1)(
    PARTITION pa,
    PARTITION pb,
    PARTITION pc
);

-- View the partition information.
SELECT b.relname AS table_name,
       a.relname AS partition_name
FROM pg_partition a,
     pg_class b
WHERE b.relname LIKE 'test_hash%'
  AND a.parttype = 'p'
  AND a.parentid = b.oid;
 table_name | partition_name
------------+----------------
 test_hash1 | p2
 test_hash1 | p1
 test_hash1 | p0
 test_hash2 | pc
 test_hash2 | pb
 test_hash2 | pa
(6 rows)

-- Drop.
DROP TABLE test_hash1,test_hash2;
```

Helpful Links

ALTER TABLE PARTITION and **DROP TABLE**

1.13.9.48 CREATE TABLE PARTITION | SUBPARTITION AS

Description

Creates a partitioned table from the results of a query.

CREATE TABLE PARTITION | SUBPARTITION AS creates a partitioned table, fills it with the result of a **SELECT** command, and partitions data based on the specified

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

partitioning policy. The table columns have the names and data types associated with the output columns of the **SELECT** (except that you can override the column names by giving an explicit list of new column names).

Syntax

- Use CREATE TABLE PARTITION AS to create a partitioned table and fill it with data from a subquery.

```
CREATE TABLE [ IF NOT EXISTS ] partition_table_name
    [ (column_name [, ...] ) ]
    [ { ENGINE [ = ] { InnoDB | 'InnoDB' | "InnoDB" } } [ ,... ] [ ... ] ]
    [ WITH ( {storage_parameter = value} [, ... ] ) ]
    [ ILM ADD POLICY ROW STORE COMPRESS
 { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON
( EXPR )]
 | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO
MODIFICATION
 | NONE}]
    [ TABLESPACE tablespace_name ]
    [ DISTRIBUTE BY { REPLICATION | { [HASH ] ( column_name ) } } ]
    [ TO { GROUP groupname | NODE ( nodename [, ... ] ) } ]
    PARTITION BY partition_clause partition_definition_list
    AS query
    [ WITH [ NO ] DATA ];
```

- Use CREATE TABLE SUBPARTITION AS to create a subpartitioned table and fill it with data from a subquery.

```
CREATE TABLE [ IF NOT EXISTS ] partition_table_name
    [ (column_name [, ...] ) ]
    [ { ENGINE [ = ] { InnoDB | 'InnoDB' | "InnoDB" } } [ ,... ] [ ... ] ]
    [ WITH ( {storage_parameter = value} [, ... ] ) ]
    [ ILM ADD POLICY ROW STORE COMPRESS
 { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON
( EXPR )]
 | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO
MODIFICATION
 | NONE}]
    [ TABLESPACE tablespace_name ]
     PARTITION BY partition_clause
     SUBPARTITION BY subpartition_clause
     partition_subpartition_definition_list
    AS query
    [ WITH [ NO ] DATA ];
```

For details about each column, see **Syntax**.

Parameters

- **IF NOT EXISTS**

  Specifies how to handle name conflicts with relations.

  When the IF NOT EXISTS keyword is specified, the system checks the current schema for any relation with the same name as the table you intend to create. If it finds such a relation, the table creation process will be halted, and a NOTICE will be returned. When IF NOT EXISTS is not specified and a relation with the same name is present in the schema, an ERROR will be returned.

- **partition_table_name**

  Specifies the name of the partitioned table to be created.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **column_name**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

(Optional) Specifies the name of a column to be created in the new table. If no column name is specified, the columns in the new table are the same as those entered in the SELECT statement.

Value range: character strings complying with **Identifier Naming Conventions**.

- **ENGINE**

  This parameter is supported in MySQL-compatible mode and is used solely for syntax adaptation. It is limited to InnoDB, but no actual impact has been noted.

- **WITH (storage_parameter [= value] [, ...])**

  Specifies an optional storage parameter for the table or an index. See details of parameters below.

  NOTE

  These TDE-related storage parameters are supported: **enable_tde**, **encrypt_algo**, **dek_cipher**, **key_type**, and **cmk_id**.

  For details about the parameters, see **CREATE TABLE**.

  – FILLFACTOR

    Specifies a fillfactor for the table. It is a percentage between 10 and 100. **100** (complete packing) is the default. In the case of Ustore, the default is **92**. When a smaller fillfactor is specified, INSERT operations pack table pages only to the indicated percentage; the remaining space on each page is reserved for updating rows on that page. This gives UPDATE a chance to place the updated copy of a row on the same page as the original, which is more efficient than placing it on a different page. For a table whose entries are never updated, complete packing (fillfactor = 100) is the best choice. However, in heavily updated tables, smaller fillfactors are appropriate. The parameter takes effect only on row-store tables.

    Value range: 10–100.

  – ORIENTATION

    Value range: **ROW**, indicating that table data is stored in row orientation.

    Default value: **ROW**.

  – autovacuum_enabled

    Specifies whether to enable AUTOVACUUM for the table.

    Value range: **on** and **off**.

    Default value: **on**.

  – autovacuum_vacuum_threshold

    (Applicable when AUTOVACUUM is enabled) Specifies the minimum number of updated or deleted tuples needed to trigger a VACUUM in the table. (This parameter applies specifically to Astore tables.)

    Value range: 0–2147483647.

    Default value: consistent with the GUC parameter **autovacuum_vacuum_threshold** unless otherwise specified.

  – autovacuum_analyze_threshold

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

(Applicable when AUTOVACUUM is enabled) Specifies the minimum number of inserted, updated, or deleted tuples needed to trigger an ANALYZE in the table.

Value range: 0–2147483647.

Default value: consistent with the GUC parameter **autovacuum_analyze_threshold** unless otherwise specified.

- autovacuum_vacuum_scale_factor

  (Applicable when AUTOVACUUM is enabled) Specifies the ratio of inserted, updated, or deleted tuples needed to trigger a VACUUM in the table. (This parameter applies specifically to Astore tables.)

  Value range: 0.0–100.0.

  Default value: consistent with the GUC parameter **autovacuum_vacuum_scale_factor** unless otherwise specified.

- autovacuum_analyze_scale_factor

  (Applicable when AUTOVACUUM is enabled) Specifies the ratio of inserted, updated, or deleted tuples needed to trigger an ANALYZE in the table.

  Value range: 0.0–100.0.

  Default value: consistent with the GUC parameter **autovacuum_analyze_scale_factor** unless otherwise specified.

- autovacuum_freeze_min_age

  (Applicable when AUTOVACUUM is enabled) Specifies the minimum range of row versions. Rows beyond this range will be frozen.

  Value range: 0–576460752303423487.

  Default value: consistent with the GUC parameter **vacuum_freeze_min_age** unless otherwise specified.

- autovacuum_freeze_max_age

  (Applicable when AUTOVACUUM is enabled) Specifies the maximum age (in transactions) that the table's **pg_class.relfrozenxid** field can reach before a VACUUM operation is forced. Note that the system will launch AUTOVACUUM processes even when AUTOVACUUM is otherwise disabled. The VACUUM operation also allows removal of old files from the **pg_clog/** subdirectory. (This parameter applies specifically to Astore tables.)

  Value range: 100000–576460752303423487.

  Default value: consistent with the GUC parameter **autovacuum_freeze_max_age** unless otherwise specified.

- autovacuum_freeze_table_age

  (Applicable when AUTOVACUUM is enabled) Specifies the time when the table will remain unchanged, marked as not requiring AUTOVACUUM. (This parameter applies specifically to Astore tables.)

  Value range: 0–576460752303423487.

  Default value: consistent with the GUC parameter **vacuum_freeze_table_age** unless otherwise specified.

- vacuum_truncate

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies whether to enable the VACUUM TRUNCATE function of a table to truncate the last empty page during VACUUM.

Value range: **on** and **off**.

Default value: **on**.

NOTE

- This parameter is valid only for Astore tables and does not take effect for hash bucket tables and segment tables.
- During the upgrade from a version that does not support this parameter to a version that supports this parameter, this parameter cannot be created or modified during the upgrade observation period.
- The toast.**vacuum_truncate** parameter can be used to independently control whether to truncate the last empty page of a TOAST table.

– hashbucket

Creates a hash bucket table. This parameter supports only row-store tables, including row-store range tables.

Value range: **on** and **off**.

Default value: **off**.

NOTICE

In current version, DDL operations on hash bucket tables are affected. Therefore, you are advised not to frequently perform DDL operations on hash bucket tables.

– hasrowid

Specifies whether the table is a rowid table. If **hasrowid** is set to **on**, the **rowid** and **rowno** system columns are added to the table, and sequences and indexes are created on the **rowno** system column.

Value range: **on** and **off**.

Default value: **off**.

NOTE

- This parameter supports only ORA-compatible databases (that is, **sql_compatibility** is set to **ORA**).
- This parameter does not support local temporary tables, unlogged tables, global temporary tables, materialized views, hash bucket tables, or replication tables.
- This parameter cannot be specified together with **hasuids** and **oids**.
- When this parameter is enabled, the ordinary column name cannot be defined as **rowno** or **rowid**.
- The number of data records in the **rowid** table cannot exceed 9223372036854775807.

– partition_statistics_min_rows

Specifies the minimum number of rows to be sampled at the partition level when the ANALYZE statement is executed on a partitioned table.

Value range: [0,10000]

Default value: **50**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This parameter is valid only for partitioned tables.
- This parameter is valid only when **PARTITION_MODE** in the ANALYZE statement is set to **ALL** and the GUC parameter **enable_partition_statistics_min_rows** is enabled. For details about **PARTITION_MODE**, see **PARTITION_MODE** in **ANALYZE | ANALYSE**.
- Upper limit: The number of rows sampled from the entire partitioned table cannot exceed the number of rows allowed by the GUC parameter **maintenance_work_mem**.

- **ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [MEDIUM | HIGH] ROW AFTER n { day | month | year } OF NO MODIFICATION [ON (EXPR)] | TURBO [LOW | MEDIUM | HIGH] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}**
  - When creating a row-store table, you can call ILM ADD POLICY ROW STORE COMPRESS ADVANCED [MEDIUM | HIGH] ROW to add a medium- or high-level advanced compression policy.
    - AFTER n { day | month | year } OF NO MODIFICATION: rows that have not been modified for *n* days, months, or years. Advanced compression does not support **hour**.
    - ON (EXPR): a row-level expression used to determine whether a row is hot or cold.

      NOTE

      - Among the functions supported by the row-level expression ON (EXPR) in the ILM policy, the output of some may be influenced by compatibility parameters. For instance, after you set **b_format_version** to **'5.7'** and **b_format_dev_version** to **'s2'** in MySQL-compatible mode, the upper function will be unable to convert data to uppercase as usual.
      - The compression level is optional. If not specified, the MEDIUM level will be used by default.
  - When creating a row-store table, you can call ILM ADD POLICY ROW STORE COMPRESS TURBO HIGH UNIT to add a cross-page transparent compression policy. Subpartitions will inherit this policy from partitions.

    AFTER n { hour | day | month | year } OF NO MODIFICATION: rows that have not been modified for *n* hours, days, months, or years.

    If the compression level is not specified, the MEDIUM level is used by default.
  - When creating a row-store table, you can call ILM ADD POLICY ROW STORE COMPRESS NONE to explicitly specify that no cross-page transparent compression policy needs to be added.
- **TABLESPACE tablespace_name**

  Specifies the name of the tablespace where the table is to be created. If not specified, the default tablespace will be used.
- **DISTRIBUTE BY**

  For details, see **DISTRIBUTE BY** in **CREATE TABLE**.
- **TO { GROUP groupname | NODE (nodename [, ...]) }**

  Specifies the node group for which the table is to be created.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

This syntax is intended only for use by internal scale-out tools and is not recommended for direct use by users.

- **PARTITION BY**

  For details about the columns of partition_clause and partition_definition_list, see **Parameters**.

- **AS query**

  Specifies a **SELECT** or **VALUES** command, or an **EXECUTE** command that runs a prepared **SELECT** or **VALUES** query.

- **[WITH [NO] DATA]**

  Specifies whether to insert the queried data into the partitioned table being created. By default, the data is inserted. If you specify **NO**, the data will not be inserted.

Examples

- Create a partitioned table and fill it with data from a subquery.

```
-- Create source table t1 and insert three data records.
gaussdb=# CREATE TABLE t1(a int, b int) DISTRIBUTE BY HASH(a);
gaussdb=# INSERT INTO t1 VALUES(8, 2);
gaussdb=# INSERT INTO t1 VALUES(18, 3);
gaussdb=# INSERT INTO t1 VALUES(28, 4);

-- Use CREATE TABLE PARTITION AS to create a partitioned table and fill it with data from a subquery.
gaussdb=# CREATE TABLE t1_part_dup DISTRIBUTE BY HASH(a) PARTITION BY RANGE(a)
(
    PARTITION p1 VALUES LESS THAN(10),
    PARTITION p2 VALUES LESS THAN(20),
    PARTITION p3 VALUES LESS THAN(MAXVALUE)
) AS SELECT * FROM t1;

gaussdb=# \d+ t1_part_dup;
          Table "create_table_as_partition_by.t1_part_dup"
 Column |  Type   | Modifiers | Storage | Stats target | Description
--------+---------+-----------+---------+--------------+-------------
 a      | integer |           | plain   |              |
 b      | integer |           | plain   |              |
Partition By RANGE(a)
Number of partitions: 3 (View pg_partition to check each partition range.)
Has OIDs: no
Distribute By: HASH(a)
Location Nodes: ALL DATANODES
Options: orientation=row, compression=no

gaussdb=# SELECT a, b FROM t1_part_dup PARTITION(p1) ORDER BY a;
 a | b
---+---
 8 | 2
(1 row)

gaussdb=# SELECT a, b FROM t1_part_dup PARTITION(p2) ORDER BY a;
 a  | b
----+---
 18 | 3
(1 row)

gaussdb=# SELECT a, b FROM t1_part_dup PARTITION(p3) ORDER BY a;
 a  | b
----+---
 28 | 4
(1 row)

gaussdb=# DROP TABLE t1, t1_part_dup;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Create a subpartitioned table and fill it with data from a subquery.

```
-- Create source table t1 and insert six data records.
gaussdb=# CREATE TABLE t1(a int, b int, c int, d varchar, e varchar) PARTITION BY RANGE(a)
SUBPARTITION BY RANGE(b)
(
    PARTITION p1 VALUES LESS THAN(200)
    (SUBPARTITION s1 VALUES LESS THAN (500),
     SUBPARTITION s2 VALUES LESS THAN (MAXVALUE)
    ),
    PARTITION p2 VALUES LESS THAN(500)
    (SUBPARTITION s3 VALUES LESS THAN (500),
     SUBPARTITION s4 VALUES LESS THAN (MAXVALUE)
    ),
    PARTITION p3 VALUES LESS THAN(MAXVALUE)
    (SUBPARTITION s5 VALUES LESS THAN (500),
     SUBPARTITION s6 VALUES LESS THAN (MAXVALUE)
    )
);

gaussdb=# INSERT INTO t1 VALUES (100, 300, 1, 'aA');
gaussdb=# INSERT INTO t1 VALUES (150, 550, 1, 'bB');
gaussdb=# INSERT INTO t1 VALUES (250, 150, 1, 'cC');
gaussdb=# INSERT INTO t1 VALUES (350, 560, 1, 'dD');
gaussdb=# INSERT INTO t1 VALUES (550, 100, 1, 'eE');
gaussdb=# INSERT INTO t1 VALUES (650, 600, 1, 'fF');

-- Use CREATE TABLE SUBPARTITION AS to create a subpartitioned table and fill it with data from a
subquery.
gaussdb=# CREATE TABLE t1_part_dup PARTITION BY RANGE(a) SUBPARTITION BY RANGE(b)
(
    PARTITION p1 VALUES LESS THAN(200)
    (SUBPARTITION s1 VALUES LESS THAN (500),
     SUBPARTITION s2 VALUES LESS THAN (MAXVALUE)
    ),
    PARTITION p2 VALUES LESS THAN(500)
    (SUBPARTITION s3 VALUES LESS THAN (500),
     SUBPARTITION s4 VALUES LESS THAN (MAXVALUE)
    ),
    PARTITION p3 VALUES LESS THAN(MAXVALUE)
    (SUBPARTITION s5 VALUES LESS THAN (500),
     SUBPARTITION s6 VALUES LESS THAN (MAXVALUE)
    )
) AS SELECT * FROM t1;

gaussdb=# \d+ t1_part_dup;
              Table "create_table_as_partition_by.t1_part_dup"
 Column |       Type        | Modifiers | Storage  | Stats target | Description
--------+-------------------+-----------+----------+--------------+-------------
 a      | integer           |           | plain    |              |
 b      | integer           |           | plain    |              |
 c      | integer           |           | plain    |              |
 d      | character varying |           | extended |              |
 e      | character varying |           | extended |              |
Partition By RANGE(a) Subpartition By RANGE(b)
Number of partitions: 3 (View pg_partition to check each partition range.)
Number of subpartitions: 6 (View pg_partition to check each subpartition range.)
Has OIDs: no
Options: orientation=row, compression=no

gaussdb=# SELECT a, b, c, d, e FROM t1_part_dup ORDER BY a;
  a  |  b  | c | d  | e
-----+-----+---+----+---
 100 | 300 | 1 | aA |
 150 | 550 | 1 | bB |
 250 | 150 | 1 | cC |
 350 | 560 | 1 | dD |
 550 | 100 | 1 | eE |
 650 | 600 | 1 | fF |
(6 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT a, b, c, d, e FROM t1_part_dup PARTITION(p1) ORDER BY a;
 a  |  b  | c | d  | e
-----+-----+---+----+---
 100 | 300 | 1 | aA |
 150 | 550 | 1 | bB |
(2 rows)

gaussdb=# SELECT a, b, c, d, e FROM t1_part_dup PARTITION(s1) ORDER BY a;
 a  |  b  | c | d  | e
-----+-----+---+----+---
 100 | 300 | 1 | aA |
(1 row)

gaussdb=# DROP TABLE t1, t1_part_dup;
```

Helpful Links

CREATE TABLE AS, **CREATE TABLE PARTITION**, and **CREATE TABLE SUBPARTITION**

1.13.9.49 CREATE TABLE SUBPARTITION

Description

Creates a subpartitioned table. Partitioning refers to splitting what is logically one large table into smaller physical pieces based on specific schemes. The initial table is known as a partitioned table, and the resulting pieces are called partitions. The partitioned table is a logical table, meaning it does not store data; instead, data is stored within these partitions. For any given subpartitioned table, the top-level node table and partitions are logical tables and do not store data. Only subpartitions (leaf nodes) store data.

The partitioning solution of a subpartitioned table is a combination of the partitioning solutions of two partitions. For details about the partitioning solution of a partitioned table, see **CREATE TABLE PARTITION**.

Common combination solutions for subpartitioned tables include range-range partitioning, range-list partitioning, range-hash partitioning, list-range partitioning, list-list partitioning, list-hash partitioning, hash-range partitioning, hash-list partitioning, and hash-hash partitioning. Currently, subpartitions are applicable only to row-store tables.

Precautions

- A subpartitioned table has two partition keys. A range/list partition key supports only 16 columns, and a hash partition key supports only one column.
- When a UNIQUE or PRIMARY KEY constraint key includes all partition keys, a local index will be created for the constraint. Otherwise, a global index will be created. If a local unique index is created, all partition keys must be included.
- If a subpartitioned table is created without explicitly specifying a subpartition within the corresponding partition, a subpartition with the same range will be created automatically.
- Each table can accommodate a maximum of 1,048,575 partitions and subpartitions in total. It is advisable to keep the number of partitions per table at 400 or less by default. For details, see "Feature Usage Guidelines >

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Partitioned Table > Introduction to Partitioned Tables > Basic Usage of Partitions > Creating Partitioned Tables" in *Developer Guide*.

- Subpartitioned tables support only row store and do not support hash bucket.
- Foreign key constraints cannot be created on the **rowid** or **rowno** system columns.
- Clusters are not supported.
- When executing a query on a specified partition, such as **SELECT * FROM** *table name* **PARTITION/SUBPARTITION** *(partition name)*, be sure to spell the keywords PARTITION and SUBPARTITION correctly. If they are misspelled, the query will not trigger an error. Instead, it will proceed with the corresponding alias rather than the specified partition.
- Encrypted databases, ledger databases, and row-level security are not supported.
- In the PARTITION/SUBPARTITION FOR (VALUES) syntax for subpartitioned tables, **VALUES** can only be a constant.
- If data type conversion is required for **VALUES** in the PARTITION/SUBPARTITION FOR (VALUES) syntax for subpartitioned tables, it is advisable to employ forcible type conversion rather than implicit type conversion, as the latter can result in unexpected outcomes.
- When creating a subpartitioned table, you can define partitions with VALUES LESS THAN (val1_1) under the range policy, or with VALUES (val1[, …]) under the list policy. The hash policy, on the other hand, does not involve defining VALUES.
- If you add or modify an ILM policy for a data object and include a row-level expression, you must use only the whitelisted functions (in **Row Expression Function Whitelist**) as the expression.

  NOTE

  Among the functions supported by the row-level expression ON (EXPR) in the ILM policy, the output of some may be influenced by compatibility parameters. For instance, after you set **b_format_version** to **'5.7'** and **b_format_dev_version** to **'s2'** in MySQL-compatible mode, the upper function will be unable to convert data to uppercase as usual.

Syntax

```
CREATE TABLE [ IF NOT EXISTS ] subpartition_table_name
{(
{ column_name data_type [ CHARACTER SET | CHARSET charset ] [ COLLATE collation ]
[ column_constraint [ ... ] ]
    | table_constraint
    | LIKE source_table [ like_option [...] ] }[, ... ]
)
    | LIKE source_table }
    [ table_option [ ,... ] [ ... ] ]
    [ WITH ( {storage_parameter = value} [, ... ] ) ]
    [ ILM ADD POLICY ROW STORE COMPRESS
 { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON
( EXPR )]
 | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION
 | NONE}][ htap_option ]
    [ TABLESPACE tablespace_name ]
    PARTITION BY {RANGE [ COLUMNS ] | LIST [ COLUMNS ] | HASH | KEY} (partition_keys) [ PARTITIONS
integer ]
    SUBPARTITION BY {RANGE | LIST | HASH | KEY} (subpartition_keys) [ SUBPARTITIONS integer ]
    (
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    PARTITION partition_name1 [ VALUES LESS THAN { (val1 | MAXVALUE) | MAXVALUE } | VALUES [ IN ]
(val1[, ...]) ]
[ ILM ADD POLICY ROW STORE COMPRESS
 { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON
( EXPR )]
 | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION
 | NONE}][ htap_option ]
[ TABLESPACE [=] tablespace ]
    [(
      { SUBPARTITION subpartition_name1 [ VALUES LESS THAN (val1_1) | VALUES (val1_1[, ...])]
[ ILM ADD POLICY ROW STORE COMPRESS
 { ADVANCED [ MEDIUM | HIGH ] ROW AFTER n { day | month | year } OF NO MODIFICATION [ ON
( EXPR )]
 | TURBO [ LOW | MEDIUM | HIGH ] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION
 | NONE}][ htap_option ]
[ TABLESPACE [=] tablespace ] } [, ...]
)][, ...]
)[ { ENABLE | DISABLE } ROW MOVEMENT ];
```

- **table_option**:

  ```
  { COMMENT [ = ] 'string' |
    AUTO_INCREMENT [ = ] value |
    [ DEFAULT ] CHARACTER SET | CHARSET [ = ] default_charset |
    [ DEFAULT ] COLLATE [ = ] default_collation |
    ENGINE [ = ] { InnoDB | 'InnoDB' | "InnoDB" } }
  ```

- **htap_option**:

  ```
  { COLVIEW [ PRIORITY { HIGH | LOW | NONE } ] |
    NOCOLVIEW }
  ```

- **column_constraint**:

  ```
  [ CONSTRAINT constraint_name ]
  { NOT NULL |
    NULL |
    CHECK ( expression ) |
    DEFAULT default_expr |
    ON UPDATE update_expr |
    AUTO_INCREMENT |
    COMMENT 'string' |
    UNIQUE [KEY] index_parameters |
    PRIMARY KEY index_parameters |
    REFERENCES reftable [ ( refcolumn ) ] [ MATCH FULL | MATCH PARTIAL | MATCH SIMPLE ]
      [ ON DELETE action ] [ ON UPDATE action ] }
  [ DEFERRABLE | NOT DEFERRABLE ] [ INITIALLY DEFERRED | INITIALLY IMMEDIATE ]
  ```

- **table_constraint**:

  ```
  [ CONSTRAINT [ constraint_name ] ]
  { CHECK ( expression ) |
    UNIQUE [ index_name ] [ USING method ] ( { column_name [ ASC | DESC ] } [, ... ] )
  index_parameters |
    PRIMARY KEY [ USING method ] ( { column_name [ ASC | DESC ] } [, ... ] ) index_parameters |
    FOREIGN KEY [ index_name ] ( column_name [, ... ] ) REFERENCES reftable [ ( refcolumn [, ... ] ) ]
      [ MATCH FULL | MATCH PARTIAL | MATCH SIMPLE ] [ ON DELETE action ] [ ON UPDATE action ] }
  [ DEFERRABLE | NOT DEFERRABLE ] [ INITIALLY DEFERRED | INITIALLY IMMEDIATE ]
  { [ COMMENT 'string' ] [ ... ] }
  ```

- **like_option**:

  ```
  { INCLUDING | EXCLUDING } { DEFAULTS | CONSTRAINTS | INDEXES | STORAGE | COMMENTS |
  RELOPTIONS | UPDATE | ALL }
  ```

- **index_parameters**:

  ```
  [ WITH ( {storage_parameter = value} [, ... ] ) ]
  [ USING INDEX TABLESPACE tablespace_name ]
  ```

- **update_expr**:

  ```
  { CURRENT_TIMESTAMP | LOCALTIMESTAMP | NOW() }
  ```

Parameters

- **IF NOT EXISTS**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Issues a notice rather than an error if a table with the same name already exists.

- **subpartition_table_name**

  Specifies the name of a subpartitioned table.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **column_name**

  Specifies the name of the column to be created in the new table.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **data_type**

  Specifies the data type of the column.

- **COLLATE collation**

  Assigns a collation to the column (which must be of a collatable data type). If not specified, the default collation is used. You can run the **SELECT * FROM pg_collation;** command to query collations from the pg_collation system catalog. The row starting with **default** in the query result represents the default collation.

- **CONSTRAINT constraint_name**

  Specifies the name of a column constraint or table constraint. The optional CONSTRAINT clause specifies constraints that new or updated rows must satisfy for an INSERT or UPDATE operation to succeed.

  There are two ways to define constraints:

  – A column constraint is defined as part of a column definition, and it is tied to a particular column.

  – A table constraint can apply to multiple columns.

  NOTICE

  The **constraint_name** parameter is optional in MYSQL-compatible mode (that is, **sql_compatibility** is set to **'MYSQL'**). For other modes, **constraint_name** must be added.

- **index_name**

  Specifies an index name.

  NOTICE

  - **index_name** is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).
  - If both **constraint_name** and **index_name** are specified for a UNIQUE constraint, the value of **index_name** will be used as the index name.

- **USING method**

  Specifies the method used to create the index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: Refer to **USING method** in **CREATE INDEX**.

NOTICE

- **USING method** is supported only in MySQL-compatible databases (**sql_compatibility = 'MYSQL'**).
- In MYSQL-compatible mode, if USING method is not specified, the default index method is B-tree for Astore or UB-tree for Ustore.

- **ASC | DESC**

  ASC specifies an ascending (default) sort order. **DESC** specifies a descending sort order.

NOTICE

ASC|DESC is supported only in a MYSQL-compatible database (that is, **sql_compatibility** is set to **'MYSQL'**).

- **LIKE source_table [like_option ...]**

  Specifies a table, known as the source table, from which the target table automatically inherits all column names, their data types, and NOT NULL constraints.

  The target table and source table are completely decoupled after creation is complete. Changes to the source table will not be applied to the target table, and it is not possible to include data of the target table in scans of the source table.

  - The default expressions of columns are included only when **INCLUDING DEFAULTS** is specified. The default behavior is to exclude the default expressions, causing all columns in the target table to default to **NULL**.
  - If **INCLUDING UPDATE** is specified, columns in the target table will inherit the **ON UPDATE CURRENT_TIMESTAMP** attribute from the source table. The default behavior is to exclude this attribute.
  - The target table always inherits the NOT NULL constraints from the source table. The CHECK constraints are inherited only when **INCLUDING CONSTRAINTS** is specified. Other types of constraints are not inherited. This rule also applies to table constraints and column constraints.

  The system does not merge inherited columns or constraints with those that share the same name. If the same name is explicitly specified or in another LIKE clause, an error is reported.

  - If **INCLUDING INDEXES** is specified, indexes on the source table will also be created on the target table. By default, no indexes are created. The tablespace where indexes on the target table are located is the same as that where indexes on the source table are located.
  - If **INCLUDING STORAGE** is specified, columns in the target table will inherit **STORAGE** settings from the source table. The default behavior is to exclude **STORAGE** settings.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - If **INCLUDING COMMENTS** is specified, the target table will inherit comments on columns, constraints, and indexes from the source table. The default behavior is to exclude comments.
  - If **INCLUDING RELOPTIONS** is specified, the target table will inherit storage parameters (WITH clause) from the source table. The default behavior is to exclude storage parameters.
  - If **INCLUDING IDENTITY** is specified, the target table will inherit the IDENTITY functionality from the source table and create a **SEQUENCE** parameter that matches the one in the source table. The default behavior is to exclude the IDENTITY functionality.
  - If **INCLUDING ILM** is specified, the target table will inherit ILM policy information from the source table. To also inherit the ILM policy information of partition objects, you will need to specify **INCLUDING PARTITION**.
  - **INCLUDING ALL** is a condensed version that encompasses **INCLUDING DEFAULTS**, **INCLUDING UPDATE**, **INCLUDING CONSTRAINTS**, **INCLUDING INDEXES**, **INCLUDING STORAGE**, **INCLUDING COMMENTS**, **INCLUDING PARTITION**, **INCLUDING RELOPTIONS**, **INCLUDING IDENTITY**, and **INCLUDING ILM**.

- **AUTO_INCREMENT [=] value**

  Specifies an initial value for an auto-increment column. The value must be a positive integer no greater than $2^{127} - 1$.

  NOTICE

  This clause takes effect only when **sql_compatibility** is set to **'MYSQL'**.

- **COMMENT [=] 'string'**
  - COMMENT [=] 'string: adds a comment on the table.
  - COMMENT 'string' in **column_constraint**: adds a comment on a column.
  - COMMENT 'string' in **table_constraint**: adds a comment on the indexes corresponding to the primary key and unique key.

  For details, see **COMMENT [=] 'string'** in **CREATE TABLE**.

- **WITH (storage_parameter [= value] [, ...])**

  Specifies an optional storage parameter for the table or index. See details of parameters below.

  NOTE

  These TDE-related storage parameters are supported: **enable_tde**, **encrypt_algo**, **dek_cipher**, **key_type**, and **cmk_id**.

  For details about the parameters, see **CREATE TABLE**.

  - FILLFACTOR

    Specifies a fillfactor for the table. It is a percentage between 10 and 100. In the case of Ustore, the default is **92**. In the case of Astore, the default is **100** (complete packing). When you specify a smaller fillfactor, INSERT operations pack table pages only to the indicated percentage. The remaining space on each page is reserved for updating rows on that

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

page. This gives UPDATE a chance to place the updated copy of a row on the same page as the original, which is more efficient than placing it on a different page. For a table whose entries are never updated, complete filling (fillfactor = 100) is the best choice. However, in heavily updated tables, smaller fillfactors are appropriate.

Value range: 10–100.

- ORIENTATION

  Specifies the orientation for storing table data.

  Value range:

  - **ROW** (default): Table data is stored in row orientation.

    NOTICE

    ORIENTATION cannot be modified.

- STORAGE_TYPE

  Specifies the storage engine type. Once set, this parameter cannot be modified.

  Value range:

  - **USTORE**: The table uses an in-place update storage engine.

    NOTICE

    To prevent space bloat, be sure to enable the **track_counts** and **track_activities** parameters for Ustore tables.

  - **ASTORE**: The table uses an append-only storage engine.

  Default value:

  In-place update storage unless otherwise specified.

- segment

  Specifies that data is stored in segment-page mode. This parameter applies only to row-store tables. Temporary tables and unlogged tables are not supported.

  Value range: **on** and **off**.

  Default value: **off**.

- statistic_granularity

  Records the default **partition_mode** when the table analyzes statistics. For details about **partition_mode**, see **PARTITION_MODE** in **ANALYZE | ANALYSE**. This parameter setting does not work for non-partitioned tables.

  Value range: See **PARTITION_MODE** in **ANALYZE | ANALYSE**.

  Default value: **AUTO**.

- hasrowid

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies whether the table is a rowid table. If **hasrowid** is set to **on**, the **rowid** and **rowno** system columns are added to the table, and sequences and indexes are created on the **rowno** system column.

Value range: **on** and **off**.

Default value: **off**.

NOTE

- This parameter supports only ORA-compatible databases (that is, **sql_compatibility** is set to **ORA**).
- This parameter does not support local temporary tables, unlogged tables, global temporary tables, materialized views, hash bucket tables, or replication tables.
- This parameter cannot be specified together with **hasuids** and **oids**.
- When this parameter is enabled, the ordinary column name cannot be defined as **rowno** or **rowid**.
- The number of data records in the **rowid** table cannot exceed 9223372036854775807.

– vacuum_truncate

Specifies whether to enable the VACUUM TRUNCATE function of a table to truncate the last empty page during VACUUM.

Value range: **on** and **off**.

Default value: **on**.

NOTE

- This parameter is valid only for Astore tables and does not take effect for hash bucket tables and segment tables.
- During the upgrade from a version that does not support this parameter to a version that supports this parameter, this parameter cannot be created or modified during the upgrade observation period.
- The toast.**vacuum_truncate** parameter can be used to independently control whether to truncate the last empty page of a TOAST table.

– partition_statistics_min_rows

Specifies the minimum number of rows to be sampled at the partition level when the ANALYZE statement is executed on a partitioned table.

Value range: [0,10000]

Default value: **50**

NOTE

- This parameter is valid only for partitioned tables.
- This parameter is valid only when **PARTITION_MODE** in the ANALYZE statement is set to **ALL** and the GUC parameter **enable_partition_statistics_min_rows** is enabled. For details about **PARTITION_MODE**, see **PARTITION_MODE** in **ANALYZE | ANALYSE**.
- Upper limit: The number of rows sampled from the entire partitioned table cannot exceed the number of rows allowed by the GUC parameter **maintenance_work_mem**.

- **[ILM ADD POLICY ROW STORE COMPRESS { ADVANCED [MEDIUM | HIGH] ROW AFTER n { day | month | year } OF NO MODIFICATION [ON (EXPR)] | TURBO [LOW | MEDIUM | HIGH] UNIT AFTER n { hour | day | month | year } OF NO MODIFICATION | NONE}]**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - When creating a table, you can call ILM ADD POLICY ROW STORE COMPRESS ADVANCED [MEDIUM | HIGH] ROW to add a medium- or high-level advanced compression policy for row store. A policy can be separately specified for a subpartition. If no policy is specified for a subpartition, subpartitions inherit the policy of the partition. If no policy is specified for a partition, subpartitions inherit the policy of the table.

    - AFTER n { day | month | year } OF NO MODIFICATION: rows that have not been modified for *n* days, months, or years. Advanced compression does not support **hour**.
    - ON (EXPR): a row-level expression used to determine whether a row is hot or cold.

    NOTE

    - Among the functions supported by the row-level expression ON (EXPR) in the ILM policy, the output of some may be influenced by compatibility parameters. For instance, after you set **b_format_version** to **'5.7'** and **b_format_dev_version** to **'s2'** in MySQL-compatible mode, the upper function will be unable to convert data to uppercase as usual.
    - The compression level is optional. If not specified, the MEDIUM level will be used by default.
  - When creating a table, you can call ILM ADD POLICY ROW STORE COMPRESS TURBO HIGH UNIT to add a cross-page transparent compression policy for row store. Subpartitions inherit the policy of partitions.

    AFTER n { hour | day | month | year } OF NO MODIFICATION: rows that have not been modified for *n* hours, days, months, or years.

    If the compression level is not specified, the MEDIUM level is used by default.
  - When creating a row-store table, you can call ILM ADD POLICY ROW STORE COMPRESS NONE to explicitly specify that no cross-page transparent compression policy needs to be added.

- **TABLESPACE tablespace_name**

  Specifies the name of the tablespace where the table is to be created. If not specified, the default tablespace is used.

- **PARTITION BY {RANGE [COLUMNS] | LIST [COLUMNS] | HASH | KEY} (partition_keys)**
  - For **partition_keys**, only 16 columns are supported for the range and list partition keys, and only one column is supported for the hash partition key.
  - The data types supported by the partition key are the same as those supported by the partitioned table constraints.
  - The COLUMNS keyword can be used only when **sql_compatibility** is set to **'MYSQL'** and can be added only after RANGE or LIST. The semantics of RANGE COLUMNS is the same as that of RANGE, and the semantics of LIST COLUMNS is the same as that of LIST.
  - The KEY keyword can be used only when **sql_compatibility** is set to **'MYSQL'**. The meaning of KEY is the same as that of HASH.
- **SUBPARTITION BY {RANGE | LIST | HASH | KEY} (subpartition_keys)**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - For **subpartition_keys**, only 16 columns are supported for the range and list partition keys, and only one column is supported for the hash partition key.
  - The data types supported by the partition key are the same as those supported by the partitioned table constraints.
  - The KEY keyword can be used only when **sql_compatibility** is set to **'MYSQL'**. The meaning of KEY is the same as that of HASH.

- **PARTITIONS integer**

  Specifies the number of partitions.

  integer must be an integer within (0,1048575].

  - When this clause is specified after range or list partitions, each partition must be explicitly defined, and the number of defined partitions must be equal to **integer**. This clause can be specified after the range and list partitions only when **sql_compatibility** is set to **'MYSQL'**.
  - When this clause is specified after hash/key partitions, the following scenarios may occur: If you do not list the definition of each partition, the system will automatically generate *integer* partitions, named in **p**+*number* format, where *number* ranges from 0 to (**integer** – 1). These partitions will be placed within the tablespace of the table by default. If you explicitly list the partition definitions, the number of defined partitions must be equal to **integer**. If you neither list the partition definitions nor specify the number of partitions, the system will create only one partition.

- **SUBPARTITIONS integer**

  Specifies the number of subpartitions.

  integer must be an integer within (0,1048575].

  - This clause must come after hash/key subpartitions.
    - If you do not list the subpartition definitions, the system will automatically generate *integer* subpartitions within each partition, named in *partition name*+**sp**+*number* format, where *number* ranges from 0 to (**integer** – 1). These subpartitions will be placed within the tablespace of the table by default.
    - If you list the subpartition definitions, the number of defined subpartitions must be equal to **integer**.
    - If you neither list the subpartition definitions nor specify the number of subpartitions, the system will create only one subpartition.

- **{ ENABLE | DISABLE } ROW MOVEMENT**

  Specifies whether to enable row movement.

  If the tuple value is updated on the partition key during the UPDATE operation, the partition where the tuple is located is altered. Setting this parameter enables error messages to be reported or movement of the tuple between partitions.

  Value range:

  - **ENABLE**: Row movement is enabled.
  - **DISABLE** (default value): Row movement is disabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

If row movement is enabled, concurrent UPDATE and DELETE operations may result in an error. Possible causes include:

Both UPDATE and DELETE operations mark old data as deleted. In addition, if changing the partition key leads to cross-partition updates, the kernel will mark the old data in the original partition as deleted and generate a new record in a different partition, making it impossible to locate the new data based on the old data.

If concurrent UPDATE+UPDATE, DELETE+DELETE, or UPDATE+DELETE operations are performed on the same row, the behavior differs based on whether the data is distributed across partitions.

a. When data is not distributed across partitions, performing the second operation after the first operation will not result in an error.

   - If the first operation is UPDATE, the most recent data can be found and accessed during the second operation.
   - If the first operation is DELETE, the system will terminate the second operation because it detects that the current data has been deleted but cannot match it with the most recent data.

b. When data is distributed across partitions, performing the second operation after the first operation will result in an error.

   - If the first operation is UPDATE, the system will report an error during the second operation because it cannot find the most recent data, which is stored in a different partition.
   - If the first operation is DELETE, the system can detect that the current data has been deleted during the second operation, but cannot match it with the most recent data or determine whether the data was deleted by an UPDATE or DELETE operation. If the operation is UPDATE, an error is reported. If the operation is DELETE, the operation is terminated. However, to ensure data correctness, it can only report an error.

In UPDATE+UPDATE and UPDATE+DELETE concurrency scenarios, you can solve the problem by performing these operations sequentially. In the DELETE +DELETE concurrency scenario, you can solve the problem by disabling row movement.

- **NOT NULL**

  Forbids NULL values in columns. **ENABLE** is provided solely for syntax compatibility and can be omitted.

- **NULL**

  Allows NULL values in columns. This is the default.

  This clause is provided solely for compatibility with non-standard SQL databases and is not recommended for use.

- **CHECK (condition) [NO INHERIT]**

  Specifies an expression producing a Boolean result which new or updated rows must satisfy for an INSERT or UPDATE operation to succeed. Expressions evaluating to TRUE or UNKNOWN succeed. Otherwise, an error exception is raised, and the INSERT or UPDATE does not alter the database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

A CHECK constraint specified as a column constraint should reference that column's value only, while an expression appearing in a table constraint can reference multiple columns.

A constraint marked with **NO INHERIT** will not propagate to child tables.

ENABLE is provided solely for syntax compatibility and can be omitted.

- **DEFAULT default_expr**

  Assigns a default value to a column. The value can be any variable-free expression (in particular, subqueries and cross-references to other columns in the current table are not allowed). The data type of the default expression must match the data type of the column.

  The default expression can be used in any INSERT operation that does not specify a value for the column. If no default value is specified, **NULL** will be used.

- **ON UPDATE update_expr**

  Specifies an attribute constraint for the column.

  During an UPDATE operation on a tuple in a table, if the new values of the updated columns differ from the old values, columns (in the tuple) that have this attribute but are not part of the UPDATE will be automatically updated to the current timestamp. However, if the new values of the updated columns are the same as the old values, those columns will remain unchanged. If columns with this attribute are included in the UPDATE, their values will be updated to the specified value.

  NOTE

  - This attribute can be specified only in MySQL-compatible databases version 5.7 (**sql_compatibility = 'MYSQL'**, **b_format_version='5.7'**, and **b_format_dev_version='s1'**).
  - In terms of syntax, **update_expr** supports three keywords: CURRENT_TIMESTAMP, LOCALTIMESTAMP, and NOW(). You can also optionally specify the precision of a keyword with parentheses. For example, ON UPDATE CURRENT_TIMESTAMP(), ON UPDATE CURRENT_TIMESTAMP(5), ON UPDATE LOCALTIMESTAMP(), and ON UPDATE LOCALTIMESTAMP(6). If a keyword does not contain parentheses or contains empty parentheses, the precision is 0. The NOW keyword cannot work without parentheses. The three types of keywords are synonyms of each other and have the same attribute effect.
  - This attribute can be specified only for columns of the following types: timestamp, datetime, date, time without time zone, smalldatetime, and abstime.
  - If the CREATE TABLE AS syntax is used, the column attributes will not be inherited.
  - The CREATE TABLE LIKE syntax can use **INCLUDING UPDATE** or **EXCLUDING UPDATE** to inherit or exclude a constraint. The **INCLUDING ILM** option is added to copy ILM policy information from the source table. When combined with the **INCLUDING PARTITION** option, it can also copy the policy information of partition objects from the source table.
  - The precision specified by this attribute can be different from the precision specified by the type in the corresponding column. After the column value is updated through this attribute, the minimum precision is displayed. For example, **CREATE TABLE t1 (col1 timestamp(6) ON UPDATE CURRENT_TIMESTAMP(3));**. If the UPDATE syntax triggers the attribute to take effect, three decimal places in the value of **col1** are displayed after the update.
  - A column cannot be designated as both this attribute and a generated column constraint simultaneously.
  - Specifying this attribute within the partition key of a partitioned table is not allowed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **AUTO_INCREMENT**

  Marks a column as auto-increment.

  For details, see **AUTO_INCREMENT** in **CREATE TABLE**.

- **UNIQUE [KEY] index_parameters**

  UNIQUE (column_name [, ...]) index_parameters

  Specifies that a column or a combination of columns must be unique across the entire table.

  For the purpose of a UNIQUE constraint, NULL is not considered equal.

  UNIQUE KEY can be used only when **sql_compatibility** is set to **'MYSQL'**, which has the same semantics as UNIQUE.

- **PRIMARY KEY index_parameters**

  PRIMARY KEY (column_name [, ...]) index_parameters

  Specifies that one or more columns in the table require unique, non-NULL values.

  Only one primary key can be specified for a table.

- **DEFERRABLE | NOT DEFERRABLE**

  Specifies whether the constraint can be deferred. A constraint that is not deferrable will be checked immediately after every command. Checking of constraints that are deferrable can be postponed until the end of the transaction (using the **SET CONSTRAINTS** command). **NOT DEFERRABLE** is the default. Currently, only the UNIQUE, PRIMARY KEY, and FOREIGN KEY constraints accept this clause. (That is, other constraints are not deferrable.)

- **INITIALLY IMMEDIATE | INITIALLY DEFERRED**

  Specifies the default time to check the constraint if it is deferrable. A constraint that is not deferrable is always **INITIALLY IMMEDIATE** and cannot be declared as **INITIALLY DEFERRED**.

  – If the constraint is **INITIALLY IMMEDIATE**, it is checked after each statement. This is the default.

  – If the constraint is **INITIALLY DEFERRED**, it is checked only at the end of the transaction.

  The constraint check time can be altered with the **SET CONSTRAINTS** command.

- **USING INDEX TABLESPACE tablespace_name**

  Specifies a tablespace in which an index associated with a UNIQUE or PRIMARY KEY constraint will be created. If this clause is not specified, the index will be created in the default tablespace named **default_tablespace**. If **default_tablespace** is empty, the default tablespace of the database will be consulted.

Examples

- Creating a subpartitioned table

```
-- Create a subpartitioned table tbl_list_list. Both of the partitions and subpartitions are of the LIST type.
gaussdb=# CREATE TABLE tbl_list_list(
    sal_year    varchar(4)  NOT NULL,
    area_id     char(5)     NOT NULL,
    emp_id      char(5)     NOT NULL,
    sales_amt   int
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
) PARTITION BY LIST (sal_year) SUBPARTITION BY LIST(area_id)(
   PARTITION P_2019 VALUES ('2019')(
      SUBPARTITION p_2019_01001 VALUES ('01001'),
      SUBPARTITION p_2019_01002 VALUES ('01002'),
      SUBPARTITION p_2019_01003 VALUES ('01003')
   ),
   PARTITION p_2020 VALUES ('2020')(
      SUBPARTITION p_2020_01001 VALUES ('01001'),
      SUBPARTITION p_2020_01002 VALUES ('01002'),
      SUBPARTITION p_2020_01003 VALUES ('01003')
   )
);

-- Create a subpartitioned table tbl_range_list. The partition is of the RANGE type and subpartition is of the LIST type.
gaussdb=# CREATE TABLE tbl_range_list(
   sal_date    varchar(6)  NOT NULL,
   area_id     char(5)     NOT NULL,
   emp_id      char(5)     NOT NULL,
   sales_amt   int
) PARTITION BY RANGE (sal_date) SUBPARTITION BY LIST(area_id)(
   PARTITION p_201901 VALUES LESS THAN (201902)(
      SUBPARTITION p_201901_01001 VALUES ('01001'),
      SUBPARTITION p_201901_01002 VALUES ('01002'),
      SUBPARTITION p_201901_01003 VALUES ('01003')
   ),
   PARTITION p_201902 VALUES LESS THAN (201903)(
      SUBPARTITION p_201902_01001 VALUES ('01001'),
      SUBPARTITION p_201902_01002 VALUES ('01002'),
      SUBPARTITION p_201902_01003 VALUES ('01003')
   )
);

-- Create a subpartitioned table multi_range_list for multi-column partition keys. The partition is of the RANGE type and subpartition is of the LIST type.
gaussdb=# CREATE TABLE multi_range_list(c1 int, c2 int, c3 int)
PARTITION BY RANGE (c1,c2) SUBPARTITION BY LIST (c2, c3)
(
   PARTITION P_RANGE1 VALUES LESS THAN (10,10)
   (
      SUBPARTITION P_RANGE1_LIST1 VALUES ((1,1),(2,2)),
      SUBPARTITION P_RANGE1_LIST2 VALUES ((5,5),(6,6)),
      SUBPARTITION P_RANGE1_LIST3 VALUES ((8,8),(9,9))
   ),
   PARTITION P_RANGE2 VALUES LESS THAN (20,20)
   (
      SUBPARTITION P_RANGE2_LIST1 VALUES ((11,11),(12,12)),
      SUBPARTITION P_RANGE2_LIST2 VALUES ((15,15),(16,16)),
      SUBPARTITION P_RANGE2_LIST3 VALUES ((18,18),(19,19))
   ),
   PARTITION P_RANGE3 VALUES LESS THAN (30,30)
   (
      SUBPARTITION P_RANGE3_LIST1 VALUES ((21,21),(22,22)),
      SUBPARTITION P_RANGE3_LIST2 VALUES ((25,25),(26,26)),
      SUBPARTITION P_RANGE3_LIST3 VALUES ((28,28),(29,29))
   )
);
```

- Specifying partitions in a subpartitioned table using DML statements
  - INSERT

    ```
    -- Insert data to a specified partition.
    gaussdb=# INSERT INTO tbl_range_list PARTITION(p_201901) VALUES('201901', '01001', '0001', 75000 );

    -- The actual partition is inconsistent with the specified partition. An error is reported.
    gaussdb=# INSERT INTO tbl_range_list PARTITION(p_201902) VALUES('201901', '01001', '0002', 6000);
    ERROR:  inserted partition key does not map to the table partition
    DETAIL:  N/A.
    ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Insert data to a specified subpartition.
gaussdb=# INSERT INTO tbl_range_list SUBPARTITION(p_201902_01001) VALUES('201902',
'01001', '0002', 8000);
```

- SELECT

```
-- Query data in a specified partition.
gaussdb=# SELECT * FROM tbl_range_list PARTITION(p_201902);
 sal_date | area_id | emp_id | sales_amt
----------+---------+--------+-----------
 201902   | 01001   | 0002   |      8000
(1 row)

gaussdb=# SELECT * FROM tbl_range_list SUBPARTITION(p_201901_01001);
 sal_date | area_id | emp_id | sales_amt
----------+---------+--------+-----------
 201901   | 01001   | 0001   |     75000
(1 row)
```

- UPDATE

```
-- Update data in a specified partition.
gaussdb=# UPDATE tbl_range_list PARTITION(p_201901) SET sales_amt = 7000;

gaussdb=# SELECT * FROM tbl_range_list;
 sal_date | area_id | emp_id | sales_amt
----------+---------+--------+-----------
 201901   | 01001   | 0001   |      7000
 201902   | 01001   | 0002   |      8000
(2 rows)

gaussdb=# UPDATE tbl_range_list SUBPARTITION FOR('201902','01001') SET sales_amt = 6000;

gaussdb=# SELECT * FROM tbl_range_list;
 sal_date | area_id | emp_id | sales_amt
----------+---------+--------+-----------
 201901   | 01001   | 0001   |      7000
 201902   | 01001   | 0002   |      6000
(2 rows)
```

- DELETE

```
-- Delete data from a specified partition.
gaussdb=# DELETE FROM tbl_range_list PARTITION (p_201901);
DELETE 1

gaussdb=# DELETE FROM tbl_range_list SUBPARTITION (p_201902_01001);
DELETE 1

gaussdb=# DELETE FROM tbl_range_list SUBPARTITION FOR ('201901','01002');
DELETE 0

-- Specify multiple partitions for deleting data, which is possible when sql_compatibility is
'MYSQL'.
gaussdb=# CREATE DATABASE db DBCOMPATIBILITY 'MYSQL';
gaussdb=# \c db
db=# CREATE TABLE range_list
(
    month_code VARCHAR2 ( 30 ) NOT NULL ,
    dept_code  VARCHAR2 ( 30 ) NOT NULL ,
    user_no    VARCHAR2 ( 30 ) NOT NULL ,
    sales_amt  int
)
PARTITION BY RANGE (month_code) SUBPARTITION BY LIST (dept_code)
(
  PARTITION p_201901 VALUES LESS THAN( '201903' )
  (
    SUBPARTITION p_201901_a VALUES ('1'),
    SUBPARTITION p_201901_b VALUES ('2')
  ),
  PARTITION p_201902 VALUES LESS THAN( '201910' )
  (
    SUBPARTITION p_201902_a VALUES ('1'),
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  SUBPARTITION p_201902_b VALUES ('2')
 )
);

db=# DELETE FROM range_list AS t PARTITION (p_201901_a, p_201901);
DELETE 0

-- Drop the database (replace the database name with the actual one).
db=# \c postgres
gaussdb=# DROP DATABASE db;

-- Drop tables.
gaussdb=# DROP TABLE tbl_list_list;
gaussdb=# DROP TABLE tbl_range_list;
gaussdb=# DROP TABLE multi_range_list;
```

Helpful Links

ALTER TABLE SUBPARTITION

1.13.9.50 CREATE TEXT SEARCH CONFIGURATION

Description

Creates a custom text search configuration, which specifies a text search parser and a list of dictionaries.

NOTICE

This functionality is designed for internal use only. Avoid direct use by users.

Precautions

- When only **PARSER** is declared, the text search configuration created will not have any mapping between string types and dictionaries. In this situation, you need to run the **ALTER TEXT SEARCH CONFIGURATION** command to create a dictionary mapping for the configuration to take effect. If you declare **COPY**, the text search configuration created will automatically copy information such as the parser, mappings, and options from the specified text search configuration.
- If a schema name is specified, the text search configuration is created in the specified schema. Otherwise, it is created in the current schema.
- The user who creates a custom text search configuration will automatically be designated as its owner.
- The **PARSER** and **COPY** options are mutually exclusive because copying an existing text search configuration also copies its text search parser.

Syntax

```
CREATE TEXT SEARCH CONFIGURATION name
    ( {PARSER = parser_name | COPY = source_config} )
    [ WITH ( {configuration_option = value} [, ...] )];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **name**

  Specifies the name of the text search configuration to be created. The name can be schema-qualified.

- **parser_name**

  Specifies the name of the text search parser used by the text search configuration.

- **source_config**

  Specifies the name of an existing text search configuration to copy.

- **configuration_option**

  Specifies the options of the text search configuration.

  Value range: default and ngram parsers along with their parameters, as described in **Table 1-399**.

  Table 1-399 Parameters specific to different parsers

Parser	Parameter	Description	Value Range
default	-	-	-
ngram	gram_size	Token length.	Positive integer from 1 to 4 Default value: **2**
	punctuation_ignore	Specifies whether to ignore punctuations.	• **true** (default): ignored. • **false**: not ignored.
	grapsymbol_ignore	Specifies whether to ignore graphical characters.	• **true**: ignored. • **false** (default): not ignored.

Examples

```
-- Create a text search configuration named ngram2.
gaussdb=# CREATE TEXT SEARCH CONFIGURATION ngram2 (PARSER=ngram) WITH (GRAM_SIZE = 2,
GRAPSYMBOL_IGNORE = false);

-- Create a text search configuration named ngram3.
gaussdb=# CREATE TEXT SEARCH CONFIGURATION ngram3 (COPY=ngram2) WITH (GRAM_SIZE = 2,
GRAPSYMBOL_IGNORE = false);

-- Add a type mapping.
gaussdb=# ALTER TEXT SEARCH CONFIGURATION ngram2 ADD MAPPING FOR multisymbol WITH simple;

-- Create user joe.
gaussdb=# CREATE USER joe IDENTIFIED BY '********';

-- Change the owner of a text search configuration.
gaussdb=# ALTER TEXT SEARCH CONFIGURATION ngram2 OWNER TO joe;

-- Change the schema of a text search configuration.
gaussdb=# ALTER TEXT SEARCH CONFIGURATION ngram2 SET SCHEMA joe;

-- Rename a text search configuration.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# ALTER TEXT SEARCH CONFIGURATION joe.ngram2 RENAME TO ngram_2;

-- Drop a type mapping.
gaussdb=# ALTER TEXT SEARCH CONFIGURATION joe.ngram_2 DROP MAPPING IF EXISTS FOR
multisymbol;

-- Drop a text search configuration.
gaussdb=# DROP TEXT SEARCH CONFIGURATION joe.ngram_2;
gaussdb=# DROP TEXT SEARCH CONFIGURATION ngram3;

-- Drop a schema and user joe.
gaussdb=# DROP SCHEMA IF EXISTS joe CASCADE;
gaussdb=# DROP ROLE IF EXISTS joe;
```

1.13.9.51 CREATE TEXT SEARCH DICTIONARY

Description

Creates a full-text search dictionary. A dictionary is used to identify and process particular words during full-text search.

NOTICE

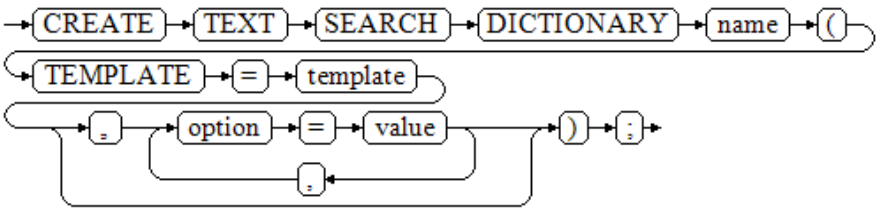
This functionality is designed for internal use only. Avoid direct use by users.

Precautions

- A user with the SYSADMIN permission can create a dictionary. Then, the user automatically becomes the owner of the dictionary.
- To create dictionaries, you will need predefined templates from the pg_ts_template system catalog. Currently, you can create various types of dictionaries, including Simple, Ispell, Synonym, Thesaurus, Snowball, and Tokenweight. Each type serves a distinct purpose.
- Dictionaries cannot be created in the temporary schema pg_temp.
- If a dictionary's **FILEPATH** requires further modifications after the dictionary is created or modified, use the ALTER statement to update **FILEPATH**.

Syntax

```
CREATE TEXT SEARCH DICTIONARY name (
    TEMPLATE = template
    [, { option = value } [, ... ]]
);
```



Parameters

- **name**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the name of the dictionary to be created. The name can be schema-qualified. If no schema is qualified, the dictionary will be created in the current schema.

Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

- **template**

  Specifies the template used to create the dictionary.

  Value range: templates predefined in the pg_ts_template system catalog, including:

  - Simple: Simple dictionary/stop word dictionary. This dictionary template operates by converting the input token to lowercase and checking it against a file of stop words. If it is found in the file, then an empty array is returned, causing the token to be discarded. If not, the lower-cased form of the word is returned as the normalized lexeme. Alternatively, the Simple dictionary can be configured to report non-stop-words as unrecognized, allowing them to be passed on to the next dictionary in the list.
  - Synonym: This dictionary template is used to create dictionaries that replace a word with a synonym.
  - Thesaurus: A Thesaurus dictionary (sometimes abbreviated as TZ) is a collection of words that includes information about the relationships of words and phrases, i.e., broader terms (BT), narrower terms (NT), preferred terms, non-preferred terms, related terms, etc. Basically, a Thesaurus dictionary replaces all non-preferred terms by one preferred term and, optionally, preserves the original terms for indexing as well.
  - Ispell: The Ispell dictionary template supports morphological dictionaries, which can normalize many different linguistic forms of a word into the same lexeme. For example, an English Ispell dictionary can match all declensions and conjugations of the search word **bank**, e.g., **banking**, **banked**, **banks**, **banks'**, and **bank's**.
  - Snowball: A Snowball dictionary provides stemming algorithms for many languages. Each algorithm understands how to reduce common variant forms of words to a base, or stem, spelling within its language. A Snowball dictionary requires a **LANGUAGE** parameter to identify which stemmer to use, and optionally can specify a **STOPWORDS** file that gives a list of words to eliminate.
  - Tokenweight: A Tokenweight dictionary can be used to adjust the weights of specific tokens in natural language processing or information retrieval. It relies on manual intervention to optimize algorithms and determine the importance of keywords.

- **option**

  Specifies an option name. The list of options varies depending on **template**.

  - Options for a Simple dictionary:

    - STOPWORDS

      Specifies the name of the stop-words file. The default file name extension is .stop.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The file format is simply a list of words, one per line. Blank lines and trailing spaces are ignored, and uppercase is folded to lowercase, but no other processing is done on the file contents.

- ACCEPT

  Specifies whether to accept a non-stop word as recognized. The default value is **true**.

  With the default setting of **ACCEPT=true**, it is only useful to place a Simple dictionary at the end of a dictionary list, since it will never pass on any token to a following dictionary. Conversely, setting **ACCEPT=false** will cause the Simple dictionary to pass on the token to the following dictionary. However, if the Simple dictionary is placed at the end of a dictionary list, **ACCEPT** will not be operational. In this scenario, it is not advisable to place the Simple dictionary at the end of the dictionary list.

- FILEPATH

  Specifies the directory storing the Simple dictionary definition file. It can be specified as a local directory in **file://***absolute_path* format. The default value is the directory where predefined dictionary files are located. If any of the **FILEPATH** and **STOPWORDS** parameters is specified, the other one must also be specified.

– Options for a Synonym dictionary:

- SYNONYM

  Specifies the name of the Synonym dictionary definition file. The default file name extension is .syn.

  The file format is simply a list of synonyms, with each line structured in *token synonym* format (each token and its corresponding synonym are separated by a space).

- CASESENSITIVE

  Specifies whether tokens and synonyms are case-sensitive. With the default value of **false**, tokens and synonyms in the synonym file are folded to lowercase. When it is **true**, words and tokens are not folded to lowercase.

- FILEPATH

  Specifies the directory storing the Synonym dictionary definition file. It can be specified as a local directory in **file://***absolute_path* format. The default value is the directory where predefined dictionary files are located.

– Options for a Thesaurus dictionary:

- DICTFILE

  Specifies the name of the Thesaurus dictionary definition file. The default file name extension is .ths.

  The file format is simply a list of synonyms, with each line structured in *sample words : indexed words* format, where the colon (:) acts as a delimiter between a phrase and its replacement. The Thesaurus dictionary chooses the longest match if there are multiple sample words matching the input.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DICTIONARY

  Specifies the name of a subdictionary used to normalize the input text. This parameter is required, and it is only possible to select one subdictionary. The specified subdictionary must exist. It is used to identify and normalize input text before phrase matching.

  An error is reported if the subdictionary fails to recognize a word. In that case, you should remove the use of the word or teach the subdictionary about it. You can place an asterisk (*) at the beginning of an indexed word to skip applying the subdictionary to it, but all sample words must be known to the subdictionary.

  If sample words in the Thesaurus dictionary definition file (**DICTFILE**) include any stop word defined in the subdictionary; use question marks (?) to replace the stop word. For example, assuming that "a" and "the" are stop words according to the subdictionary:

  ```
  ? one ? two : swsw
  ```

  matches "a one the two" and "the one a two"; both would be replaced by "swsw".

- FILEPATH

  Specifies the directory storing the Thesaurus dictionary definition file. It can be specified as a local directory in **file://***absolute_path* format. The default value is the directory where predefined dictionary files are located.

– Options for an Ispell dictionary:

- DICTFILE

  Specifies the name of the dictionary definition file. The default file name extension is .dict.

- AFFFILE

  Specifies the name of an .affix file. The default file name extension is .affix.

- STOPWORDS

  Specifies the name of the stop-words file. The default file name extension is .stop. The stop-words file has the same format explained above for the Simple dictionary type.

- FILEPATH

  Specifies the directory storing the Ispell dictionary definition file. It can be specified as a local directory in **file://***absolute_path* format. The default value is the directory where predefined dictionary files are located.

– Options for a Snowball dictionary:

- LANGUAGE

  Specifies the name of a language whose stemming algorithm will be used. According to spelling rules in the language, the algorithm normalizes the variants of an input word into a basic word or a stem.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- STOPWORDS

  Specifies the name of the stop-words file. The default file name extension is .stop. The stop-words file has the same format explained above for the Simple dictionary type.

- FILEPATH

  Specifies the directory storing the Snowball dictionary definition file. It can be specified as a local directory in **file://***absolute_path* format. The default value is the directory where predefined dictionary files are located. If any of the **FILEPATH** and **STOPWORDS** parameters is specified, the other one must also be specified.

– Options for a Tokenweight dictionary:

- TOKENFILE

  Specifies the name of the Tokenweight dictionary definition file. The default file name extension is .tokw.

  The file format is simply a list of token weights and synonym similarities, with each line representing a token weight or synonym similarity. When processing a specific file, the dictionary will ignore empty lines, comments starting with "#", and spaces at the beginning or end of each line.

  ○ Token weight format: *<token words> [: <token weight>]*. *token weight* can be a decimal or an integer, with a default value of **3**. The colon (:) acts as a delimiter in the middle. Increasing the token weight enhances the probability of the character string being segmented into a single word.

  ○ Synonym similarity format: *<token words 1> :: <token words 2> [: <similarity>]*, indicating that **token words 2** is a synonym of **token words 1**. The value of **similarity** ranges from 0 to 1, with a default of **0.8**. Note that synonym similarities only denote one-way synonymy; to represent two-way synonymy, two lines of definitions with reversed word positions are necessary.

- STOPFILE

  Specifies the name of the stop-words file. The default file name extension is .stop. The stop-words file has the same format explained above for the Simple dictionary type.

- FILEPATH

  Specifies the directory storing the Tokenweight dictionary definition file. It can be specified as a local directory in **file://***absolute_path* format. The default value is the directory where predefined dictionary files are located. The **FILEPATH** parameter must be specified together with parameter **STOPFILE** or **TOKENFILE**.

NOTE

The name of the dictionary definition file can contain only lowercase letters, digits, and underscores (_).

- **value**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the option value. If the value is of the character string type, it must be enclosed in single quotation marks (identifiers and numbers can also be enclosed in single quotation marks).

Examples

Text search configuration specifies the following components required for converting a document into a tsvector:

- A parser, used for breaking a text into tokens.
- A list of dictionaries, used for converting each token into a lexeme.

Each time when the to_tsvector or to_tsquery function is called, a text search configuration is required to specify a processing procedure. The GUC parameter **default_text_search_config** specifies the default text search configuration, which will be used if the text search function does not explicitly specify a text search configuration.

GaussDB provides some predefined text search configurations. You can also create user-defined text search configurations. In addition, to facilitate the management of text search objects, multiple gsql meta-commands are provided to display information about text search objects. (For details, see **Meta-Command Reference**.)

Procedure

Step 1 Create a text search configuration **ts_conf** by copying the predefined text search configuration **english**.

```
gaussdb=# CREATE TEXT SEARCH CONFIGURATION ts_conf ( COPY = pg_catalog.english );
CREATE TEXT SEARCH CONFIGURATION
```

Step 2 Create a Synonym dictionary.

Assume that the definition file **gs_dict.syn** of the Synonym dictionary contains the following contents:

```
gaussdb    gs
gauss      gs
```

Create the Synonym dictionary.

```
gaussdb=# CREATE TEXT SEARCH DICTIONARY gs_dict (
    TEMPLATE = synonym,
    SYNONYMS = gs_dict,
    FILEPATH = 'file:///home/dicts'
);
```

Step 3 Create an Ispell dictionary named **english_ispell** (the dictionary definition file is from the open source dictionary).

```
gaussdb=# CREATE TEXT SEARCH DICTIONARY english_ispell (
   TEMPLATE = ispell,
   DictFile = english,
   AffFile = english,
   StopWords = english,
   FILEPATH = 'file:///home/dicts'
);
```

Step 4 Alter the text search configuration **ts_conf** and change the dictionary list for tokens of certain types.

```
gaussdb=# ALTER TEXT SEARCH CONFIGURATION ts_conf
   ALTER MAPPING FOR asciiword, asciihword, hword_asciipart,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
                word, hword, hword_part
  WITH gs_dict, english_ispell, english_stem;
```

Step 5 In the text search configuration, set no indexing or searching for tokens of certain types.

```
gaussdb=# ALTER TEXT SEARCH CONFIGURATION ts_conf
   DROP MAPPING FOR email, url, url_path, sfloat, float;
```

Step 6 Use the text search debugging function ts_debug() to test the text search configuration **ts_conf**.

```
gaussdb=# SELECT * FROM ts_debug('ts_conf', '
GaussDB, the highly scalable, SQL compliant, open source object-relational
database management system, is now undergoing beta testing of the next
version of our software.
');
```

Step 7 You can set the default text search configuration of the current session to **ts_conf**. This setting is valid only in the current session.

```
gaussdb=# \dF+ ts_conf
     Text search configuration "public.ts_conf"
Parser: "pg_catalog.default"
      Token      |            Dictionaries
-----------------+------------------------------------
 asciihword      | gs_dict,english_ispell,english_stem
 asciiword       | gs_dict,english_ispell,english_stem
 file            | simple
 host            | simple
 hword           | gs_dict,english_ispell,english_stem
 hword_asciipart | gs_dict,english_ispell,english_stem
 hword_numpart   | simple
 hword_part      | gs_dict,english_ispell,english_stem
 int             | simple
 numhword        | simple
 numword         | simple
 uint            | simple
 version         | simple
 word            | gs_dict,english_ispell,english_stem

gaussdb=# SET default_text_search_config = 'public.ts_conf';
SET
gaussdb=# SHOW default_text_search_config;
 default_text_search_config
----------------------------
 public.ts_conf
(1 row)
```

----End

1.13.9.52 CREATE TRIGGER

Description

CREATE TRIGGER is used to create a trigger. A trigger is a specialized stored procedure used for complex service rules and requirements to ensure reference integrity. It is associated with a specified table or view and executes a specified function under specific conditions. Triggers help preserve predefined relationships within a table when rows are added, updated, or deleted.

Precautions

- Currently, triggers can be created only on ordinary tables and views, instead of on temporary tables or unlogged tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If multiple triggers of the same kind are defined for the same event, they will be fired in alphabetical order by name.
- Triggers are usually used for data synchronization between multiple tables, which impact SQL execution performance greatly. Therefore, you are advised not to use them when a large amount of data needs to be synchronized and performance requirements are high.
- When a trigger meets the following conditions, the trigger statement and trigger itself can be pushed together down to a DN for execution, improving the trigger execution performance:
  - The GUC parameters **enable_trigger_shipping** and **enable_fast_query_shipping** are enabled.
  - The trigger function used by the source table is a PL/pgSQL function (recommended).
  - The source and target tables have the same type and number of distribution keys, are both row-store tables, and belong to the same node group.
  - The **INSERT**, **UPDATE**, or **DELETE** statement on the source table contains an expression about equality comparison between all the distribution keys and the *NEW* or *OLD* variable.
  - The INSERT, UPDATE, or DELETE statement on the source table can be used to query shipping without a trigger.
  - There are only six types of triggers on the source table: **BEFORE INSERT FOR EACH ROW**, **AFTER INSERT FOR EACH ROW**, **BEFORE UPDATE FOR EACH ROW**, **AFTER UPDATE FOR EACH ROW**, **BEFORE DELETE FOR EACH ROW**, and **AFTER DELETE FOR EACH ROW**. All triggers can be pushed down.
- When a trigger statement is executed, the permission is determined by the trigger creator.
- To create a trigger, you must have the TRIGGER permission on the specified table or have the CREATE ANY TRIGGER permission.
- A row-level trigger function triggered by BEFORE can return a **NULL** value, indicating that operations on the row are ignored. Subsequent triggers will not be executed and no INSERT, UPDATE, or DELETE action will be generated on the row.

  The return value of the trigger function triggered by AFTER is not affected.
- In the case of a BEFORE DELETE trigger, the return value **NEW** of the trigger function indicates **NULL**. In the case of a BEFORE INSERT trigger, the return value **OLD** of the trigger function indicates **NULL**. In the case of a BEFORE UPDATE trigger, the return value of the trigger function is **NULL** only when it is displayed as **NULL**.
- For a trigger function whose event is INSERT or UPDATE, the normal return value is **NEW**. For a row returning a value other than **NULL**, a BEFORE trigger modifies the inserted or updated row, while an AFTER trigger does not. For a trigger function whose event is DELETE, the normal return value is **OLD**.
- INSTEAD OF triggers can only work on views. Their trigger functions can also return **NULL**, indicating that subsequent triggers will not be executed.
- Package functions cannot be used as trigger functions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

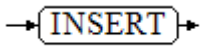
- If there are overloaded functions, the implicit variables in triggers, such as *old* and *new*, cannot be used as input parameters of the overloaded functions.
- In a distributed PGXC plan, if a BEFORE ROW trigger modifies the same row in the target table, the modification of the trigger does not take effect.
- The MERGE INTO statement does not support triggers.
- The REPLACE INTO statement does not support ROW triggers. For STATEMENT triggers, only INSERT STATEMENT is supported, and DELETE STATEMENT is not supported.
- The INSERT ... ON DUPLICATE KEY UPDATE | ON CONFLICT statement supports STATEMENT triggers but does not support ROW triggers.
- In a trigger function or anonymous block, **cursor%RowType** is considered as a composite type (record), and the specific type of the cursor cannot be parsed. The corresponding column information is obtained only when this type is used to define variables and assign values.

Syntax

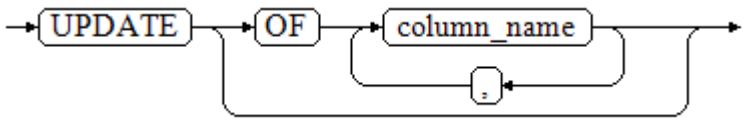
```
CREATE [OR REPLACE] [ CONSTRAINT ] TRIGGER name { BEFORE | AFTER | INSTEAD OF } { event [ OR
event ] [ ... ] }
   ON table_name
   [ FROM referenced_table_name ]
   [ NOT DEFERRABLE [ INITIALLY IMMEDIATE ] | [ DEFERRABLE ] [ INITIALLY IMMEDIATE | INITIALLY
DEFERRED ] ]
   [ FOR [ EACH ] { ROW | STATEMENT } ]
   [ WHEN ( condition ) ]
   [ EXECUTE PROCEDURE function_name ( arguments ) | AnonyBlock ];
```

Events include:

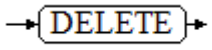
```
INSERT
```



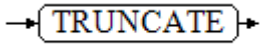

```
UPDATE [ OF column_name [, ... ] ]
```




```
DELETE
```




```
TRUNCATE
```



Parameters

- **OR REPLACE**

  (Optional) If this parameter is specified, the existing trigger will be modified.

  Constraint triggers and internal triggers cannot be created or modified using the OR REPLACE syntax. A constraint trigger is a trigger created using CREATE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

CONSTRAINT TRIGGER. An internal trigger is a trigger implicitly created by some SQL statements. For example, if a foreign key constraint is added to a table, four triggers are implicitly created. The four triggers are internal triggers.

- **CONSTRAINT**

  (Optional) Creates a constraint trigger. That is, the trigger is used as a constraint. This is the same as a regular trigger except that the timing of the trigger firing can be adjusted using SET CONSTRAINTS. Constraint triggers must be AFTER ROW triggers.

- **name**

  Specifies the name of the trigger to be created. This must be distinct from the name of any other trigger for the same table. The name cannot be schema-qualified because the trigger inherits the schema of its table. For a constraint trigger, this is also the name to use when modifying the trigger's behavior using **SET CONSTRAINTS**.

  Value range: a string, which complies with **Identifier Naming Conventions**. A value can contain a maximum of 63 characters.

- **BEFORE**

  Specifies that the function is called before the event.

- **AFTER**

  Specifies that the function is called after the event. A constraint trigger can only be specified as **AFTER**.

- **INSTEAD OF**

  Specifies that the function is called instead of the event.

- **event**

  Specifies the event that will fire the trigger. Values are **INSERT**, **UPDATE**, **DELETE**, and **TRUNCATE**. Multiple events can be specified using **OR**.

  For UPDATE events, it is possible to specify a list of columns using this syntax:

  ```
  UPDATE OF column_name1 [, column_name2 ... ]
  ```

  The trigger will only fire if at least one of the listed columns is mentioned as a target of the UPDATE statement. INSTEAD OF UPDATE events do not allow a list of columns. If the column specified by UPDATE OF contains a generated column, the trigger is also fired when the column on which the generated column depends is the target column of the UPDATE statement.

- **table_name**

  Specifies the name of the table for which the trigger is created.

  Value range: existing table names in the database.

- **referenced_table_name**

  Specifies the name of another table referenced by the constraint. This option is used for foreign-key constraints. It can only be specified for constraint triggers. Because foreign keys are not supported currently, this option is not recommended for general use.

  Value range: existing table names in the database.

- **DEFERRABLE | NOT DEFERRABLE**

  Specifies the start time of the trigger. It can only be specified for constraint triggers. Determines whether the constraint can be deferred.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

For details, see **CREATE TABLE**.

- **INITIALLY IMMEDIATE | INITIALLY DEFERRED**

  If the constraint is deferrable, the two clauses specify the default time to check the constraint. It can only be specified for constraint triggers.

  For details, see **CREATE TABLE**.

- **FOR EACH ROW | FOR EACH STATEMENT**

  Specifies the frequency of firing the trigger.

  - **FOR EACH ROW** indicates that the trigger should be fired once for every row affected by the trigger event.
  - **FOR EACH STATEMENT** indicates that the trigger should be fired just once per SQL statement.

  If not specified, **FOR EACH STATEMENT** is the default. Constraint triggers can only be specified as **FOR EACH ROW**.

- **condition**

  Specifies whether the trigger function will actually be executed. If WHEN is specified, the function will be called only when **condition** returns **true**.

  In FOR EACH ROW triggers, the WHEN condition can refer to columns of the old and/or new row values by writing **OLD.***column_name* or **NEW.***column_name* respectively. In addition, INSERT triggers cannot refer to OLD, and DELETE triggers cannot refer to NEW.

  INSTEAD OF triggers do not support WHEN conditions.

  Currently, WHEN expressions cannot contain subqueries.

  Note that for constraint triggers, evaluation of the WHEN condition is not deferred, but occurs immediately after the row update operation is performed. If the condition does not evaluate to **true**, then the trigger is not queued for deferred execution.

- **function_name**

  Specifies a user-defined function, which must be declared as taking no parameters and returning type trigger. This is executed when a trigger fires.

- **arguments**

  Specifies an optional comma-separated list of parameters to be provided to the function when the trigger is executed. The parameters are literal string constants. Simple names and numeric constants can also be written here, but they will all be converted to strings. Check the description of the implementation language of the trigger function to find out how these parameters can be accessed within the function.

  NOTE

  The following details trigger types:

  - INSTEAD OF triggers must be marked as FOR EACH ROW and can be defined only on views.
  - BEFORE and AFTER triggers on a view must be marked as FOR EACH STATEMENT.
  - TRUNCATE triggers must be marked as FOR EACH STATEMENT.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 1-400 Types of triggers supported on tables and views

When	Event	Row-Level	Statement-Level
BEFORE	INSERT/UPDATE/DELETE	Tables	Tables and views
	TRUNCATE	Not supported.	Tables
AFTER	INSERT/UPDATE/DELETE	Tables	Tables and views
	TRUNCATE	Not supported.	Tables
INSTEAD OF	INSERT/UPDATE/DELETE	Views	Not supported.
	TRUNCATE	Not supported.	Not supported.

Table 1-401 Special variables in PL/pgSQL functions

Variable	Description
NEW	New tuple for INSERT and UPDATE operations. This variable is **NULL** for **DELETE** operations and statement-level triggers.
OLD	Old tuple for UPDATE and DELETE operations. This variable is **NULL** for **INSERT** operations and statement-level triggers.
TG_NAME	Trigger name.
TG_WHEN	Trigger timing (**BEFORE**, **AFTER**, or **INSTEAD OF**).
TG_LEVEL	Trigger frequency (**ROW** or **STATEMENT**).
TG_OP	Trigger event (**INSERT**, **UPDATE**, **DELETE**, or **TRUNCATE**).
TG_RELID	OID of the table where the trigger resides.
TG_RELNAME	Name of the table where the trigger is located. (Currently, it is replaced by *TG_TABLE_NAME*, which has the same function. You are advised to use *TG_TABLE_NAME*.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Variable	Description
TG_TABLE_NAME	Name of the table where the trigger resides.
TG_TABLE_SCHEMA	Schema of the table where the trigger resides.
TG_NARGS	Number of parameters for the trigger function.
TG_ARGV[]	List of parameters for the trigger function.

Examples

```
-- Create a source table and a destination table.
gaussdb=# CREATE TABLE test_trigger_src_tbl(id1 INT, id2 INT, id3 INT);
gaussdb=# CREATE TABLE test_trigger_des_tbl(id1 INT, id2 INT, id3 INT);
```

Using the INSERT trigger

```
-- Create an INSERT trigger function.
gaussdb=# CREATE OR REPLACE FUNCTION tri_insert_func() RETURNS TRIGGER AS
        $$
        DECLARE
        BEGIN
                INSERT INTO test_trigger_des_tbl VALUES(NEW.id1, NEW.id2, NEW.id3);
                RETURN NEW;
        END
        $$ LANGUAGE plpgsql;

-- Create an INSERT trigger.
gaussdb=# CREATE TRIGGER insert_trigger
        BEFORE INSERT ON test_trigger_src_tbl
        FOR EACH ROW
        EXECUTE PROCEDURE tri_insert_func();

-- Execute the INSERT event and check the trigger results.
gaussdb=# INSERT INTO test_trigger_src_tbl VALUES(100,200,300);

gaussdb=# SELECT * FROM test_trigger_src_tbl;
 id1 | id2 | id3
-----+-----+-----
 100 | 200 | 300
(1 row)

gaussdb=# SELECT * FROM test_trigger_des_tbl; // Check whether the trigger operation takes effect.
 id1 | id2 | id3
-----+-----+-----
 100 | 200 | 300
(1 row)
```

Using the UPDATE trigger

```
-- Create an UPDATE trigger function.
gaussdb=# CREATE OR REPLACE FUNCTION tri_update_func() RETURNS TRIGGER AS
$$
DECLARE
BEGIN
UPDATE test_trigger_des_tbl SET id3 = NEW.id3 WHERE id1=OLD.id1;
RETURN OLD;
END
$$ LANGUAGE plpgsql;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create an UPDATE trigger.
gaussdb=# CREATE TRIGGER update_trigger
        AFTER UPDATE ON test_trigger_src_tbl
        FOR EACH ROW
        EXECUTE PROCEDURE tri_update_func();

-- Execute the UPDATE event and check the trigger results.
gaussdb=# UPDATE test_trigger_src_tbl SET id3=400 WHERE id1=100;

gaussdb=# SELECT * FROM test_trigger_src_tbl;
 id1 | id2 | id3
-----+-----+-----
 100 | 200 | 400
(1 row)

gaussdb=# SELECT * FROM test_trigger_des_tbl; // Check whether the trigger operation takes effect.
 id1 | id2 | id3
-----+-----+-----
 100 | 200 | 400
(1 row)
```

Using the DELETE trigger

```
-- Create a DELETE trigger function.
gaussdb=# CREATE OR REPLACE FUNCTION TRI_DELETE_FUNC() RETURNS TRIGGER AS
        $$
        DECLARE
        BEGIN
                DELETE FROM test_trigger_des_tbl WHERE id1=OLD.id1;
                RETURN OLD;
        END
        $$ LANGUAGE plpgsql;

-- Create a DELETE trigger.
gaussdb=# CREATE TRIGGER delete_trigger BEFORE DELETE ON test_trigger_src_tbl FOR EACH ROW
EXECUTE PROCEDURE tri_delete_func();

-- Execute the DELETE event and check the trigger results.
gaussdb=# DELETE FROM test_trigger_src_tbl WHERE id1=100;

gaussdb=# SELECT * FROM test_trigger_src_tbl;
 id1 | id2 | id3
-----+-----+-----
(0 rows)

gaussdb=# SELECT * FROM test_trigger_des_tbl; // Check whether the trigger operation takes effect.
 id1 | id2 | id3
-----+-----+-----
(0 rows)
```

Renaming a trigger

```
-- Rename a trigger.
gaussdb=# ALTER TRIGGER delete_trigger ON test_trigger_src_tbl RENAME TO delete_trigger_renamed;
```

Disabling triggers

```
-- Disable insert_trigger.
gaussdb=# ALTER TABLE test_trigger_src_tbl DISABLE TRIGGER insert_trigger;

gaussdb=# INSERT INTO test_trigger_src_tbl VALUES(100,200,300);

gaussdb=# SELECT * FROM test_trigger_src_tbl;
 id1 | id2 | id3
-----+-----+-----
 100 | 200 | 300
(1 row)

gaussdb=# SELECT * FROM test_trigger_des_tbl; // View that the trigger does not take effect.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 id1 | id2 | id3
-----+-----+-----
(0 rows)

-- Disable all triggers on the current table.
gaussdb=# ALTER TABLE test_trigger_src_tbl DISABLE TRIGGER ALL;
```

Dropping triggers

```
gaussdb=# DROP TRIGGER insert_trigger ON test_trigger_src_tbl;

gaussdb=# DROP TRIGGER update_trigger ON test_trigger_src_tbl;

gaussdb=# DROP TRIGGER delete_trigger_renamed ON test_trigger_src_tbl;
```

Dropping functions

```
gaussdb=# DROP FUNCTION tri_insert_func;

gaussdb=# DROP FUNCTION tri_update_func;

gaussdb=# DROP FUNCTION tri_delete_func;
-- Drop the source table and destination table.
gaussdb=# DROP TABLE test_trigger_src_tbl;
gaussdb=# DROP TABLE test_trigger_des_tbl;
```

Helpful Links

ALTER TRIGGER, **DROP TRIGGER**, and **ALTER TABLE**

1.13.9.53 CREATE TYPE

Description

Defines a new data type for use in the current database.

Currently, only these types are available: composite type, base type, shell type, enumerated (enum) type, and set type. Here are the details for each type:

- Composite type

  A composite type is specified by a list of attribute names and data types. If the data type of an attribute is collatable, the attribute's collation can also be specified. This is essentially the same as the row type of a table, but using CREATE TYPE avoids the need to create an actual table when all that is wanted is to define a type. A stand-alone composite type is useful as the parameter or return type of a function.

  To create a composite type, you must have the USAGE permission on all of its attribute types.

- Base type

  You can create a base type (scalar type).

- Shell type

  A shell type is simply a placeholder for a type to be defined later; it is created by issuing CREATE TYPE with no parameters except for the type name. Shell types are needed as forward references when base types are created.

- Enumerated type

  An enumerated type is a list of one or more quoted labels, each of which must be 1 to 64 bytes long.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Set type

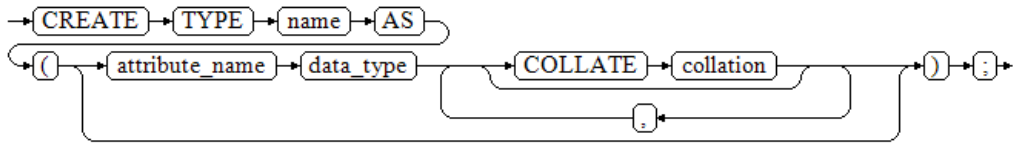
  It is similar to an array but has no length limit. It is mainly used in stored procedures.

Precautions

- If a schema name is specified, the type is created in the specified schema. Otherwise, it is created in the current schema.
- The type name must be distinct from the name of any existing type or table in the same schema. (Because tables have associated data types, the name of the new type must also be distinct from the name of any existing table in the same schema.)
- The user who defines a type becomes its owner.
- Users with the CREATE ANY TYPE permission can create types in the PUBLIC and user schemas.
- Types are designed only for row-store tables.
- When creating a non-system type by associating a function, the user needs to understand the definition of the new type and the associated function. If the operation is improper, permissions may be abused due to the associated function.

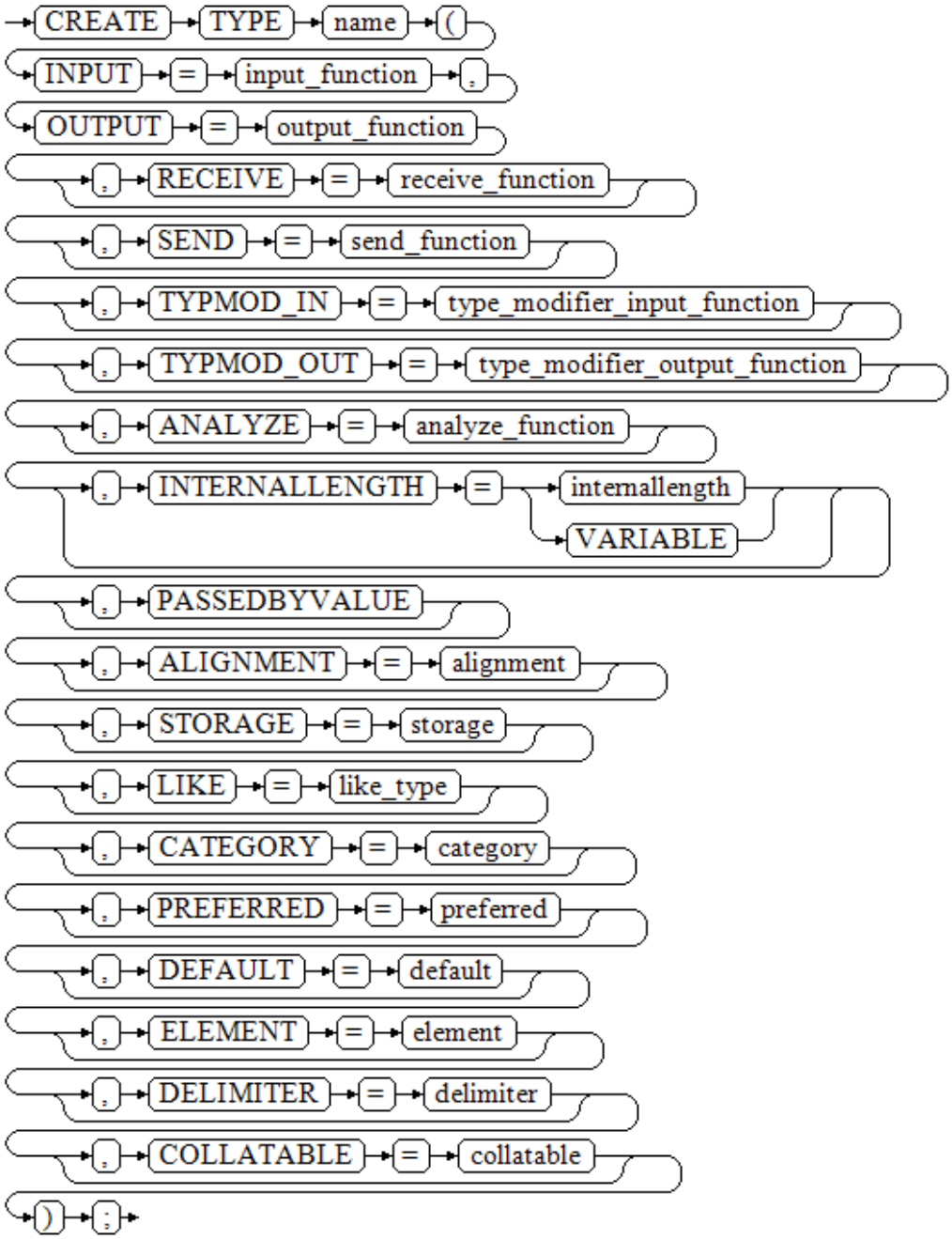
Syntax

```
CREATE TYPE name AS
    ( [ attribute_name data_type [ COLLATE collation ] [, ... ] ] );
```

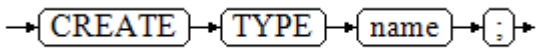



```
CREATE TYPE name (
    INPUT = input_function,
    OUTPUT = output_function
    [ , RECEIVE = receive_function ]
    [ , SEND = send_function ]
    [ , TYPMOD_IN = type_modifier_input_function ]
    [ , TYPMOD_OUT = type_modifier_output_function ]
    [ , ANALYZE = analyze_function ]
    [ , INTERNALLENGTH = { internallength | VARIABLE } ]
    [ , PASSEDBYVALUE ]
    [ , ALIGNMENT = alignment ]
    [ , STORAGE = storage ]
    [ , LIKE = like_type ]
    [ , CATEGORY = category ]
    [ , PREFERRED = preferred ]
    [ , DEFAULT = default ]
    [ , ELEMENT = element ]
    [ , DELIMITER = delimiter ]
    [ , COLLATABLE = collatable ]
);
```

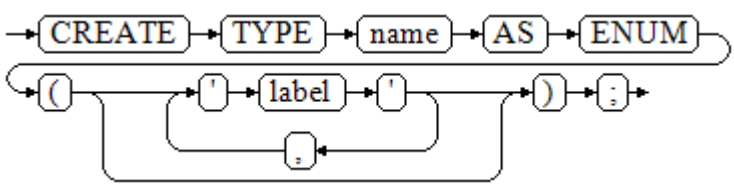
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.




```
CREATE TYPE name;
```

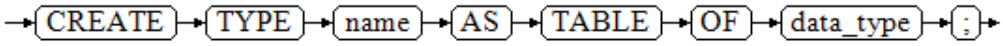



```
CREATE TYPE name AS ENUM
    ( [ {'label'} [, ... ] ] );
```




```
CREATE TYPE name AS TABLE OF data_type;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- Composite type
  - **name**

    Specifies the name (optionally schema-qualified) of the type to be created.
  - **attribute_name**

    Specifies the name of an attribute (column) for the composite type.
  - **data_type**

    Specifies the name of an existing data type to become a column of the composite type. You can use **%RowType** to indirectly reference the type of a table, or **%Type** to indirectly reference the type of a column in a table or a composite type.
  - **collation**

    Specifies the name of an existing collation to be associated with a column of the composite type. You can run the **SELECT * FROM pg_collation;** command to query collations from the pg_collation system catalog. The row starting with **default** in the query result represents the default collation.

    NOTE

    The constructor of the composite type can use the => to assign values. The restrictions are as follows:

    - The => function is used when the constructor of a composite type assigns values. This function is supported only in an ORA-compatible database (**sql_compatibility** set to **'ORA'**).
    - The constructor of the composite type uses the => function to assign values. This function is supported only when => is consecutively used to assign values to input parameters, for example, **composite_name(elename1 => val1, elename2 => val2, elename3 => val3)** or **composite_name(val1, elename2 => val2, elename3 => val3)**. Discontinuous use of => or => is not supported. The last input parameter, for example, **composite_name(elename1 => val1, elename2 => val2, val3)** or **composite_name(val1, elename2 => val2, val3)**, is not assigned a value.
- Base type: When creating a base type, you can place parameters in any order. The **input_function** and **output_function** parameters are required, and other parameters are optional.
  - **input_function**

    Specifies the name of a function that converts data from the type's external textual form to its internal form.

    The input function can be declared as taking one parameter of the CSTRING type, or as taking three parameters of the CSTRING, OID, and INTEGER types.

    - The CSTRING-type parameter is the input text as a C string.
    - The OID-type parameter is the type's own OID (except for array types, which instead receive their element type's OID).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The INTEGER-type parameter is the typmod of the destination column, if known (**-1** will be passed if not).

The input function must return a value of the data type itself. Usually, an input function should be declared STRICT; if it is not, it will be called with a NULL first parameter when reading a NULL input value. The function must still return NULL in this case, unless it raises an error.

NOTE

- The input and output functions can be declared to have results or parameters of the new type, when they have to be created before the new type can be created. The type should first be defined as a shell type, which is a placeholder type that has no attributes except a name and an owner. This is done by issuing the **CREATE TYPE** *name* command, with no additional parameters. Then, the I/O functions written in C can be defined as referencing the shell type. Finally, CREATE TYPE with a full definition replaces the shell entry with a complete, valid type definition, after which the new type can be used normally.
- If the input and output functions are of the INTERNAL type and specified as internal system functions, they must share the same parameter types. Additionally, **INTERNALLENGTH** and **PASSEDBYVALUE** of the new type should also match the parameter types of the input and output functions.

– **output_function**

Specifies the name of a function that converts data from the type's internal form to its external textual form.

The output function must be declared as taking one parameter of the new data type. The output function must return type CSTRING. Output functions are not called for **NULL** values.

– **receive_function**

(Optional) Specifies the name of a function that converts data from the type's external binary form to its internal form.

If this function is not supplied, the type cannot participate in binary input. The binary representation should be chosen to be cheap to convert to internal form, while being reasonably portable. (For example, the standard integer data types use network byte order as the external binary representation, while the internal representation is in the machine's native byte order.) The receive function should perform adequate checking to ensure that the value is valid.

The receive function can be declared as taking one parameter of the INTERNAL type, or as taking three parameters of the INTERNAL, OID, and INTEGER types.

- The INTERNAL-type parameter is a pointer to a StringInfo buffer holding the received byte string.
- The OID- and INTEGER-type parameters are the same as for the text input function.

The receive function must return a value of the data type itself. Usually, a receive function should be declared STRICT; if it is not, it will be called with a NULL first parameter when reading a NULL input value. The function must still return NULL in this case, unless it raises an error.

– **send_function**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

(Optional) Specifies the name of a function that converts data from the type's internal form to its external binary form.

If this function is not supplied, the type cannot participate in binary output. The send function must be declared as taking one parameter of the new data type. The send function must return type BYTEA. Send functions are not called for **NULL** values.

- **type_modifier_input_function**

  (Optional) Specifies the name of a function that converts an array of modifiers for a type to its internal format.

- **type_modifier_output_function**

  (Optional) Specifies the name of a function that converts the internal format of modifiers for a type to its external text format.

  NOTE

  type_modifier_input_function and **type_modifier_output_function** are needed if the type supports modifiers, that is optional constraints attached to a type declaration, such as char(5) or numeric(30,2). GaussDB allows user-defined types to take one or more simple constants or identifiers as modifiers. However, this information must be capable of being packed into a single non-negative integer value for storage in the system catalogs. The **type_modifier_input_function** is passed the declared modifiers in the form of a CSTRING array. It must check the values for validity (throwing an error if they are wrong), and if they are correct, return a single non-negative integer value that will be stored as the column "typmod". Type modifiers will be rejected if the type does not have a **type_modifier_input_function**. The **type_modifier_output_function** converts the internal integer typmod value back to the correct form for user display. It must return a CSTRING value that is the exact string to append to the type name. For example, a numeric function may return (30,2). It is allowed to omit the **type_modifier_output_function**, in which case the default display format is just the stored typmod integer value enclosed in parentheses.

- **analyze_function**

  (Optional) Specifies the name of a function that performs type-specific statistics collection for columns of the data type.

  By default, ANALYZE will attempt to collect statistics using the type's "equals" and "less-than" operators, if there is a default B-tree operator class for the type. For non-scalar types, this behavior is likely to be unsuitable, so it can be overridden by specifying a custom analysis function. The analysis function must be declared to take a single parameter of the INTERNAL type and return a Boolean result.

- **internallength**

  (Optional) Specifies the length in bytes of the new type's internal representation. The default assumption is that it is variable-length.

  While the details of the new type's internal representation are only known to the I/O functions and other functions you create to work with the type, there are several attributes of the internal representation that must be declared to GaussDB. Foremost of these is **INTERNALLENGTH**. Base data types can be fixed-length, in which case **internallength** is a positive integer, or variable-length, indicated by setting **INTERNALLENGTH** to **VARIABLE**. (Internally, this is represented by setting **typlen** to **-1**.) The internal representation of all variable-length types must start with a 4-byte integer and follow the total length defined by **internallength**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **PASSEDBYVALUE**

  (Optional) Indicates that values of this data type are passed by value, rather than by reference. Types passed by value must be fixed-length, and their internal representation cannot be larger than the size of the Datum type (4 bytes on some machines, 8 bytes on others).

- **alignment**

  (Optional) Specifies the storage alignment requirement of the data type. If specified, it must be **CHAR**, **INT2**, **INT4**, or **DOUBLE**; the default is **INT4**.

  The allowed values equate to alignment on 1, 2, 4, or 8 byte boundaries. Note that variable-length types must have an alignment of at least 4, since they necessarily contain an INT4 as their first component.

- **storage**

  (Optional) Specifies the storage strategy for the data type.

  If specified, it must be **plain**, **external**, **extended**, or **main**; the default is **plain**.

  - **plain** specifies that data of the type will always be stored in-line and not compressed. (Only **plain** is allowed for fixed-length types.)
  - **extended** specifies that the system will first attempt to compress a long data value, and will move the value out of the main table row if it is still too long.
  - **external** allows the value to be moved out of the main table, but the system will not attempt to compress it.
  - **main** allows compression, but discourages moving the value out of the main table. (Data items with this storage strategy might still be moved out of the main table if there is no other way to make a row fit, but they will be kept in the main table preferentially over **extended** and **external** items.)

  All **storage** values other than **plain** imply that the functions of the data type can handle values that have been toasted. The specific other value given merely determines the default TOAST storage strategy for columns of a TOAST data type; users can pick other strategies for individual columns using ALTER TABLE SET STORAGE.

- **like_type**

  (Optional) Specifies the name of an existing data type that the new type will have the same representation as. The values of **internallength**, **passedbyvalue**, **alignment**, and **storage** are copied from that type, unless overridden by explicit specification elsewhere in this **CREATE TYPE** statement.

  Specifying representation this way is especially useful when the low-level implementation of the new type "piggybacks" on an existing type in some fashion.

- **category**

  (Optional) Specifies the category code (a single ASCII character) for this type. The default is **U** for a user-defined type. You may also choose other ASCII characters to create custom categories.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **preferred**

  (Optional) Specifies whether a type is preferred within its type category. If it is, the value will be **true**, else **false**. The default is **false**. Be very careful about creating a preferred type within an existing type category, as this could cause surprising changes in behavior.

  NOTE

  The **category** and **preferred** parameters can be used to help control which implicit cast will be applied in ambiguous situations. Each data type belongs to a category named by a single ASCII character, and each type is either preferred or not within its category. The parser will prefer casting to preferred types (but only from other types within the same category) when this rule is helpful in resolving overloaded functions or operators. For types that have no implicit casts to or from any other types, it is sufficient to leave these settings at the defaults. However, for a group of related types that have implicit casts, it is often helpful to mark them all as belonging to a category and select one or two of the most general types as being preferred within the category. The **category** parameter is especially useful when adding a user-defined type to an existing built-in category, such as the numeric or string types. However, it is also possible to create entirely-user-defined type categories. Select any ASCII character other than an uppercase letter to name such a category.

- **default**

  (Optional) Specifies the default value for the data type. If this is omitted, the default is null.

  A default value can be specified, in case a user wants columns of the data type to default to something other than the null value. Specify the default with the DEFAULT keyword. (Such a default can be overridden by an explicit DEFAULT clause attached to a particular column.)

- **element**

  (Optional) Specifies the type of array elements when an array type is created. For example, to define an array of 4-byte integers (INT4), specify **ELEMENT = INT4**.

- **delimiter**

  (Optional) Specifies the delimiter character to be used between values in arrays made of this type.

  delimiter can be set to a specific character. The default delimiter is the comma (,). Note that the delimiter is associated with the array element type, not the array type itself.

- **collatable**

  (Optional) Specifies whether this type's operations can use collation information. If they can, the value will be **true**. The default is **false**.

  If **collatable** is **true**, column definitions and expressions of the type may carry collation information through use of the COLLATE clause. It is up to the implementations of the functions operating on the type to actually make use of the collation information; this does not happen automatically merely by marking the type collatable.

- **label**

  (Optional) Represents the textual label associated with one value of an enumerated type. It is a string of 1 to 63 characters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

Whenever a user-defined type is created, GaussDB automatically creates an associated array type whose name consists of the element type's name prepended with an underscore (_).

Examples

```
-- Create a composite type, create a table, insert data, and make a query.
gaussdb=# CREATE TYPE compfoo AS (f1 int, f2 text);
gaussdb=# CREATE TABLE t1_compfoo(a int, b compfoo);
gaussdb=# CREATE TABLE t2_compfoo(a int, b compfoo);
gaussdb=# INSERT INTO t1_compfoo values(1,(1,'demo'));
gaussdb=# INSERT INTO t2_compfoo SELECT * FROM t1_compfoo;
gaussdb=# SELECT (b).f1 FROM t1_compfoo;
gaussdb=# SELECT * FROM t1_compfoo t1 JOIN t2_compfoo t2 ON (t1.b).f1=(t1.b).f1;

-- Rename the data type.
gaussdb=# ALTER TYPE compfoo RENAME TO compfoo1;

-- Change the owner of the user-defined type compfoo1 to usr1.
gaussdb=# CREATE USER usr1 PASSWORD '********';
gaussdb=# ALTER TYPE compfoo1 OWNER TO usr1;

-- Change the schema of the user-defined type compfoo1 to usr1.
gaussdb=# ALTER TYPE compfoo1 SET SCHEMA usr1;

-- Add a new attribute to the data type.
gaussdb=# ALTER TYPE usr1.compfoo1 ADD ATTRIBUTE f3 int;

-- Drop the compfoo1 type.
gaussdb=# DROP TYPE usr1.compfoo1 CASCADE;

-- Drop related tables and users.
gaussdb=# DROP TABLE t1_compfoo;
gaussdb=# DROP TABLE t2_compfoo;
gaussdb=# DROP SCHEMA usr1;
gaussdb=# DROP USER usr1;

-- Create an enumerated type.
gaussdb=# CREATE TYPE bugstatus AS ENUM ('create', 'modify', 'closed');

-- Add a label.
gaussdb=# ALTER TYPE bugstatus ADD VALUE IF NOT EXISTS 'regress' BEFORE 'closed';

-- Rename a label.
gaussdb=# ALTER TYPE bugstatus RENAME VALUE 'create' TO 'new';

-- Create a set type.
gaussdb=# CREATE TYPE bugstatus_table AS TABLE OF bugstatus;

-- Drop a set type.
gaussdb=# DROP TYPE bugstatus_table;
-- Drop an enumerated type.
gaussdb=# DROP TYPE bugstatus CASCADE;
-- Create an ORA-compatible database and switch to this database.
gaussdb=#  CREATE DATABASE ora_compatible_db DBCOMPATIBILITY 'ORA';
CREATE DATABASE
gaussdb=#  \c ora_compatible_db;
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "ora_compatible_db" as user "omm".
-- Create a composite type.
ora_compatible_db=# CREATE TYPE person_type AS  (name VARCHAR2(50), age NUMBER, gender
VARCHAR2(10));
CREATE TYPE
ora_compatible_db=# CREATE TYPE address_type AS  (street VARCHAR2(50), city VARCHAR2(50), zip_code
VARCHAR2(10));
CREATE TYPE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ora_compatible_db=# CREATE TYPE customer_type AS  (id NUMBER, person_info person_type, address_info
address_type);
CREATE TYPE
-- When assigning values, the constructor of the composite type uses => for all input parameters. The
execution is successful.
ora_compatible_db=# DECLARE
  v_customer customer_type;
 BEGIN
  v_customer := customer_type(
    id => 123,
    person_info => person_type(name => 'John', age => 30, gender => 'Male'),
    address_info => address_type(street => '123 Main St', city => 'Anytown', zip_code => '12345')
  );
 END;
 /
ANONYMOUS BLOCK EXECUTE
-- When assigning values to input parameters, the constructor of the composite type uses => consecutively
up to the last input parameter. The execution is successful.
ora_compatible_db=# DECLARE
  v_person person_type;
 BEGIN
  v_person := person_type('John', age => 30, gender => 'Male');
 END;
 /
ANONYMOUS BLOCK EXECUTE
-- When assigning values to input parameters, the constructor of the composite type does not use =>
consecutively up to the last input parameter. An error is reported.
ora_compatible_db=# DECLARE
  v_person person_type;
 BEGIN
  v_person := person_type(name => 'John', age => 30, 'Male');
 END;
 /
ERROR:  Positional parameters cannot follow named parameters
LINE 1: SELECT person_type(name => 'John', age => 30, 'Male')
                                                   ^
QUERY:  SELECT person_type(name => 'John', age => 30, 'Male')
CONTEXT:  referenced column: person_type
PL/pgSQL function inline_code_block line 4 at assignment
-- Drop composite types.
ora_compatible_db=# DROP TYPE customer_type;
DROP TYPE
ora_compatible_db=# DROP TYPE address_type;
DROP TYPE
ora_compatible_db=# DROP TYPE person_type;
DROP TYPE
-- Drop the created database and switch back to the original database.
ora_compatible_db=# \c postgres;
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "postgres" as user "omm".
gaussdb=# DROP DATABASE ora_compatible_db;
DROP DATABASE
```

Helpful Links

ALTER TYPE and **DROP TYPE**

1.13.9.54 CREATE USER

Description

Creates a user with a specified password.

A user is a basic element for GaussDB authentication. A user can use the correct username and password to log in to the GaussDB and grant different permissions to different users so that different users can perform different operations.

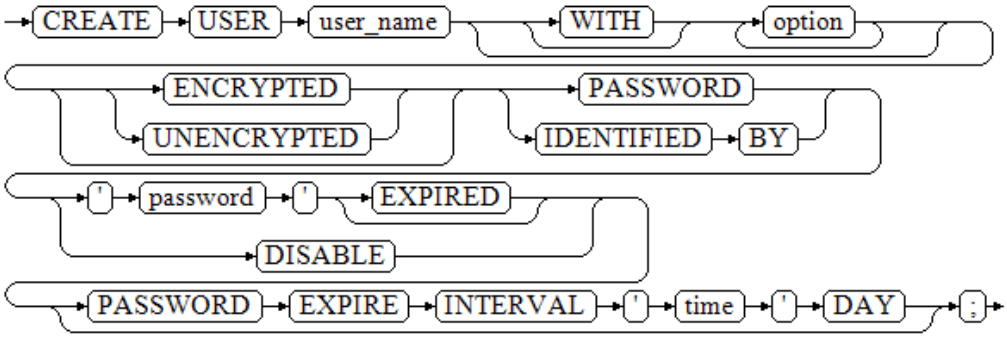
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

- A user created using the CREATE USER statement has the LOGIN permission by default.
- When you run the **CREATE USER** command to create a user, the system creates a schema with the same name as the user in the database where the command is executed.
- If a system administrator creates an object in the schema with the same name as a common user, the ownership of the object will be assigned to the common user rather than the system administrator.

Syntax

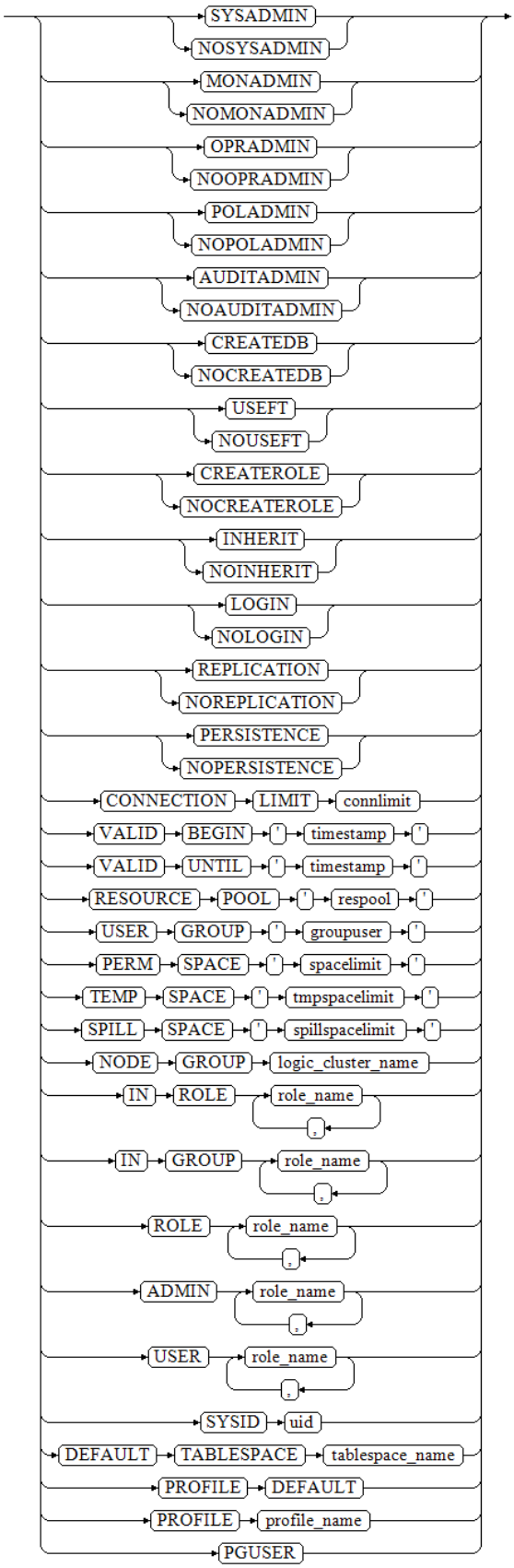
```
CREATE USER user_name [ [ WITH ] option [ ... ] ] [ ENCRYPTED | UNENCRYPTED ] { PASSWORD |
IDENTIFIED BY } { 'password' [EXPIRED] | DISABLE }[ PASSWORD EXPIRE INTERVAL 'time' DAY ];
```



The option clause is used to configure information, including permissions and properties.

```
{
   {SYSADMIN | NOSYSADMIN}
   | {MONADMIN | NOMONADMIN}
   | {OPRADMIN | NOOPRADMIN}
   | {POLADMIN | NOPOLADMIN}
   | {AUDITADMIN | NOAUDITADMIN}
   | {CREATEDB | NOCREATEDB}
   | {USEFT | NOUSEFT}
   | {CREATEROLE | NOCREATEROLE}
   | {INHERIT | NOINHERIT}
   | {LOGIN | NOLOGIN}
   | {REPLICATION | NOREPLICATION}
   | {PERSISTENCE | NOPERSISTENCE}
   | CONNECTION LIMIT connlimit
   | VALID BEGIN 'timestamp'
   | VALID UNTIL 'timestamp'
   | RESOURCE POOL 'respool'
   | USER GROUP 'groupuser'
   | PERM SPACE 'spacelimit'
   | TEMP SPACE 'tmpspacelimit'
   | SPILL SPACE 'spillspacelimit'
   | NODE GROUP logic_cluster_name
   | IN ROLE role_name [, ...]
   | IN GROUP role_name [, ...]
   | ROLE role_name [, ...]
   | ADMIN role_name [, ...]
   | USER role_name [, ...]
   | SYSID uid
   | DEFAULT TABLESPACE tablespace_name
   | PROFILE DEFAULT
   | PROFILE profile_name
   | PGUSER
}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **user_name**

  Name of the user to be created.

  Value range: character strings complying with **Identifier Naming Conventions** and not exceeding 63 characters.

  If the username entered exceeds 63 characters, the database will truncate it and keep the first 63 characters.

  If a username contains uppercase letters, the database automatically converts the uppercase letters into lowercase letters. To create a username that contains uppercase letters, declare the username with double quotation marks ("").

  NOTICE

  redisuser is used for managing redistributed resources. It is currently not possible to create a user with **user_name** set as **redisuser**.

- **password**

  Specifies the login password.

  Value range: character strings.

  Password rules:

  – Contain at least eight characters. This is the default length.

  – Differ from the username or the username spelled backward.

  – Contain at least three of the following character types: uppercase characters, lowercase characters, digits, and special characters (limited to ~!@#$%^&*()-_=+\|[{}];:,<.>/?). If the password contains characters other than the preceding characters, an error will be reported during statement execution.

  – The password can also be a ciphertext character string that meets the format requirements. It is mainly used to import user data. You are advised not to use it directly. The database does not verify the complexity of the ciphertext password. Using the ciphertext password may cause data security risks. If you use the ciphertext password, you need to know the plaintext corresponding to the ciphertext password and ensure that the password meets the complexity requirements.

  – When creating a user, use single or double quotation marks to declare the user password.

- For details about other parameters, see **CREATE ROLE**.

Examples

```
-- Create user jim whose login password is ********.
gaussdb=# CREATE USER jim PASSWORD '********';

-- Create user kim whose login password is ********.
gaussdb=# CREATE USER kim IDENTIFIED BY '********';

-- Create user tom.
gaussdb=# CREATE USER TOM PASSWORD '********';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create user TOM.
gaussdb=# CREATE USER "TOM" PASSWORD '********';

-- To create a user with the CREATEDB permission, add the CREATEDB keyword.
gaussdb=# CREATE USER dim CREATEDB PASSWORD '********';

-- Query the permissions of the dim user.
gaussdb=#  \du dim
          List of roles
 Role name | Attributes | Member of
-----------+------------+-----------
 dim       | Create DB  | {}
(You can see that the dim user has the CREATEDB permission.)

-- Change the password of user jim.
gaussdb=# ALTER USER jim IDENTIFIED BY '**********' REPLACE '********';

-- Add the CREATEROLE permission to jim.
gaussdb=# ALTER USER jim CREATEROLE;

-- View the CREATEROLE permission added to user jim.
gaussdb=# \du jim
           List of roles
 Role name | Attributes  | Member of
-----------+-------------+-----------
 jim       | Create role | {}

-- Lock jim.
gaussdb=# ALTER USER jim ACCOUNT LOCK;

-- Unlock jim.
gaussdb=# ALTER USER jim ACCOUNT UNLOCK;

-- Change the user password.
gaussdb=# ALTER USER dim WITH PASSWORD '********';

-- Change the username.
gaussdb=# ALTER USER dim RENAME TO lisa;

-- Create user user1 with the OPRADMIN permission and user user2 with the SYSADMIN permission.
gaussdb=# CREATE USER user1 WITH OPRADMIN PASSWORD '********';
gaussdb=# CREATE USER user2 WITH SYSADMIN PASSWORD '********';

-- Query permissions.
gaussdb=# \du user1
                  List of roles
 Role name |         Attributes          | Member of
-----------+-----------------------------+-----------
 user1     |        Operatoradmin        | {}

gaussdb=# \du user2
                List of roles
 Role name |       Attributes       | Member of
-----------+------------------------+-----------
 user2     |       Sysadmin         | {}

-- Create role user3 with the CREATEDB permission.
gaussdb=# CREATE USER user3 CREATEDB PASSWORD '********';
CREATE ROLE

-- Query permissions.
gaussdb=# \du user3
                List of roles
 Role name |       Attributes        | Member of
-----------+-------------------------+-----------
 user3     |       Create DB         | {}

-- Create role user4 using the ADMIN clause.
gaussdb=# CREATE USER user4 WITH  CREATEDB  ADMIN user3  PASSWORD '********';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
CREATE ROLE
-- Query permissions.
gaussdb=# \du user3
                List of roles
 Role name |       Attributes        | Member of
-----------+-------------------------+-----------
 user3     |       Create DB         | {user4}

gaussdb=# \du user4
                List of roles
 Role name |       Attributes        | Member of
-----------+-------------------------+-----------
 user4     |       Create DB         | {}

-- Drop the roles.
gaussdb=# DROP USER user1;
gaussdb=# DROP USER user2;
gaussdb=# DROP USER user3;
gaussdb=# DROP USER user4;

-- Separation-of-duties configuration: If enableSeparationOfDuty is set to on, the separation of duties is enabled to restrict the permissions of system administrators. System administrators are not allowed to create or modify user configurations.

-- Check whether separation of duties is enabled.
gaussdb=# SHOW enableSeparationOfDuty;
 enableSeparationOfDuty
------------------------
 off
(1 row)
-- Switch to database user omm and run the following statement to set parameters.
gs_guc set -Z coordinator -Z datanode -N all -I all -c "enableSeparationOfDuty = on"

-- After the setting is successful, the setting takes effect after a restart.
gs_om -t stop
gs_om -t start

-- Check the separation of duties again.
gaussdb=# SHOW enableSeparationOfDuty;
 enableSeparationOfDuty
------------------------
 on
(1 row)

-- Drop users.
gaussdb=# DROP USER kim CASCADE;
gaussdb=# DROP USER jim CASCADE;
gaussdb=# DROP USER lisa CASCADE;
gaussdb=# DROP USER TOM CASCADE;
gaussdb=# DROP USER "TOM" CASCADE;

-- Set the validity period of the user password.
gaussdb=# CREATE USER user1 PASSWORD '********' PASSWORD EXPIRE INTERVAL 10 DAY;
```

Helpful Links

ALTER USER, **CREATE ROLE**, and **DROP USER**

1.13.9.55 CREATE VIEW

Description

Creates a view. A view is a virtual table, not a base table. A database stores only the view definition but not the view data. The view data is stored in a base table. If data in the base table changes, the data in the view changes accordingly. In this

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

sense, a view is like a window through which users can know their interested data and data changes in the database.

- You can define frequently used data as views to encapsulate complex query SQL statements, which simplifies operations.
- You can query only data defined in a view, which is secure. Columns in a base table are hidden to protect the data structure of the database.
- User permission management is simplified. Only the permission to use views is granted to users.

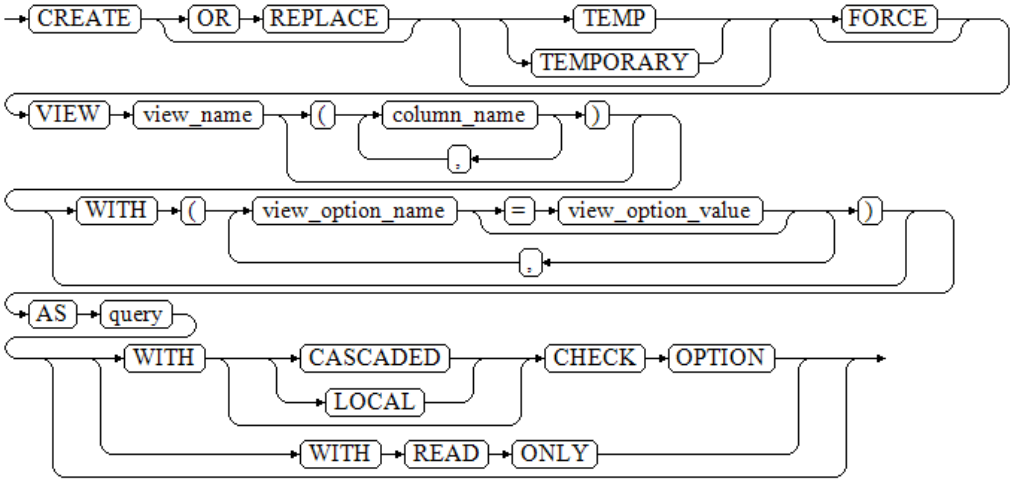
Precautions

A user with the CREATE ANY TABLE permission can create views in the public and user schemas.

Non-initial users are not allowed to perform the CREATE OR REPLACE operation on system views in the SYS, DBE_PERF, and INFORMATION_SCHEMA schemas.

Syntax

```
CREATE [ OR REPLACE ] [ TEMP | TEMPORARY ] [ FORCE ] VIEW view_name [ ( column_name [, ...] ) ]
    [ WITH ( {view_option_name [= view_option_value]} [, ... ] ) ]
    AS query
    [ WITH [ CASCADED | LOCAL ] CHECK OPTION  | WITH READ ONLY ];
```



NOTE

- You can use WITH(security_barrier) to create a relatively secure view. This prevents attackers from printing base table data by using the RAISE statement of low-cost functions.
- Modifying column names or types, or dropping columns from a base view will automatically invalidate dependent views, regardless of the **enable_view_invalidation** setting.

Parameters

- **OR REPLACE**

  (Optional) Redefines the view if it already exists.

- **TEMP | TEMPORARY**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

(Optional) Creates a temporary view. The view is automatically deleted when the current session ends. If any table referenced by a view is a temporary table, the view is created as a temporary view (regardless of whether **TEMP|TEMPORARY** is specified in the SQL statement).

NOTE

- Temporary views do not support automatic recompilation.
- To recompile an invalid view, refer to "Feature Usage Guidelines > Recompilation upon Invalidation" in *Developer Guide*.

- **FORCE**

  (Optional) Successfully creates the view even if the objects it depends on are absent or the required query permissions are missing.

- **view_name**

  Specifies the name of the view to be created. It is optionally schema-qualified.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **column_name**

  Specifies an optional list of names to be used for columns of the view. If not given, the column names are deduced from the query.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **view_option_name [= view_option_value]**

  Specifies an optional parameter for a view.

  Currently, view_option_name supports only **security_barrier** and **check_option**.

  security_barrier: This parameter is used when the view attempts to provide row-level security. Value range: Boolean (**true** or **false**).

  check_option: controls the behavior of updating a view. Value range: **CASCADED** and **LOCAL**.

- **query**

  Specifies a SELECT or VALUES statement that will provide the columns and rows of the view.

NOTICE

If **query** contains a clause specifying the partition of a partitioned table, the OID of the specified partition is directly written to the system catalog when the view is created. However, using any DDL syntax that changes the OID of the specified partition—for example, DROP, SPLIT, or MERGE PARTITION—will render the view invalid. (Refer to "Feature Usage Guidelines > Recompilation upon Invalidation" in *Developer Guide*.)

- **WITH [CASCADED | LOCAL] CHECK OPTION**

  Controls the behavior of updatable views. INSERT and UPDATE on the view will be checked to ensure that new rows meet the view-defining condition, that is, the new rows are visible through the view. If the check fails, the update is rejected. If this option is not included, INSERT and UPDATE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

operations on the view are permitted to create rows that are not visible through the view. When DELETE is performed on a view, if the base table from which data is to be deleted is duplicate in a multi-table join view and the duplicate base tables are not all key-reserved tables in the view, data cannot be deleted when **WITH CHECK OPTION** is specified. The **WITH CHECK OPTION** option can be specified as **CASCADED** or **LOCAL**.

CASCADED: New rows are checked against the conditions of the view and all underlying views. If **CHECK OPTION** is specified, and neither **LOCAL** nor **CASCADED** is specified, then **CASCADED** is used by default.

LOCAL: Only the conditions directly defined by the view itself are checked. If **CHECK OPTION** is also defined in the underlying views, the conditions defined in the underlying views are checked.

- **WITH READ ONLY**

  A read-only view is defined. You cannot insert, update, or delete data in the view.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The concepts involved in the constraints on updating, inserting, and deleting views are described as follows:
  - Join view: view created with multiple tables using JOIN.
  - Key-preserved table: Insertion, update, and deletion of multi-table join views are restricted by key-preserved tables. In a multi-table view, if each row in the source table corresponds to each row in the view, and a row in the source table does not correspond to multiple rows in the view after the JOIN, the source table is a key-preserved table.
  - Currently, you can determine whether a target table is a key-preserved table only by using the JOIN condition. The equivalence class (that is, combining JOIN and WHERE) cannot be used to determine whether a target table is a key-preserved table.
  - Relationship between the top layer and the bottom layer: A view may be nested at multiple layers. For example, a view consists of one or more views or subqueries. The view that is directly operated by the DML is called the top layer. The tables and views that form the view and the tables and views in the WITH clause are called the corresponding bottom layer.
  - Columns that can be updated: Columns that are not system columns or whole-row references and directly reference user columns in the base table can be updated. You can query the **is_updatable** column in **information_schema.columns** to check whether a column in a view or table can be updated.
  - Updatable view: view that can be inserted, updated, or deleted. Updatable views do not contain DISTINCT, GROUP BY, HAVING, LIMIT, OFFSET, UNPIVOT, FOR UPDATE, FETCH and START WITH CONNECT BY clauses, set operations (UNION, INTERSECT, and EXCEPT), aggregate functions, window functions, and return set functions (array_agg, json_agg, and generate_series), and flashback. **WITH CHECK OPTION** can be defined only on updatable views. To check whether a view can be updated, query the **is_updatable** and **is_insertable_into** columns in **information_schema.views** or the **information_schema.tables.is_insertable_into** column. In **information_schema.views**, **is_updatable** specifies whether a view can be updated or deleted, **is_insertable_into** specifies whether data can be inserted into a view, and **information_schema.tables.is_insertable_into** specifies whether data can be inserted into a relationship.
- If an updatable view has an INSTEAD OF trigger or INSTEAD rule, using CHECK OPTION does not check the conditions on that view.
- If an underlying view of an updatable view has an INSTEAD OF trigger and the updatable view defines the **CHECK OPTION** option of the CASCADED type, in non-ORA-compatible mode, the conditions of the underlying view with the INSTEAD OF trigger are recursively checked, and the conditions of the underlying view with the trigger are not checked. In ORA-compatible mode, the **CHECK OPTION** option of all views is invalid and no check is performed.
- If an updatable view or any of its underlying relationships has an INSTEAD rule that causes INSERT or UPDATE rewriting, specifying **CHECK OPTION** will not check the conditions on all views.
- If **CASCADED CHECK OPTION** is defined in the upper-layer view and **LOCAL CHECK OPTION** is defined in the lower-layer view, **LOCAL CHECK OPTION** in the lower-layer view is overwritten by **CASCADED CHECK OPTION** in the upper-layer view.
- If **CHECK OPTION** is specified, insert and update operations cannot be performed on join columns in multi-table join views or multi-table join subqueries.
- If **CHECK OPTION** is specified, duplicate base tables exist in a multi-table join view or multi-table join subquery, and the duplicate base tables are not all key-preserved tables, the view or subquery cannot be deleted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

- Ordinary view

```
-- Create the test_tb1 table and insert 100 data records into the table.
gaussdb=# CREATE TABLE test_tb1(col1 int, col2 int);
gaussdb=# INSERT INTO test_tb1 VALUES (generate_series(1,100),generate_series(1,100));
-- Create a view whose value of col1 is less than 3.
gaussdb=# CREATE VIEW test_v1 AS SELECT * FROM test_tb1 WHERE col1 < 3;
-- Query a view.
gaussdb=# SELECT * FROM test_v1;
 col1 | col2
------+------
    1 |    1
    2 |    2
(2 rows)

-- Drop the table and view.
gaussdb=# DROP VIEW test_v1;
gaussdb=# DROP TABLE test_tb1;
```

- Read-only view

```
-- Create the test_tb table.
gaussdb=# CREATE TABLE test_tb(c1 int, c2 int);

-- Create the test_v read-only view.
gaussdb=# CREATE  VIEW test_v AS SELECT * FROM test_tb WITH READ ONLY;

-- Write data to the view to check whether the view is read-only.
gaussdb=# INSERT INTO test_v VALUES (1, 1);
ERROR:  DML operation cannot be performed on a read-only view.

-- Drop the table and view.
gaussdb=# DROP VIEW test_v;
gaussdb=# DROP TABLE test_tb;
```

- REPLACE application

```
-- Create tables a and b.
gaussdb=# CREATE TABLE a(c1 int, c2 int);
gaussdb=# CREATE TABLE b(c1 int, c2 int, c3 int);

-- Insert data into a and b.
gaussdb=# INSERT INTO a VALUES (1, 1);
gaussdb=# INSERT INTO b VALUES (1, 1, 1);

-- Query.
gaussdb=# SELECT * FROM a;
 c1 | c2
----+----
  1 |  1
(1 row)

gaussdb=# SELECT * FROM b;
 c1 | c2 | c3
----+----+----
  1 |  1 |  1
(1 row)

-- Create view a_view. The view does not exist here, which is equivalent to creating a view.
gaussdb=# CREATE OR REPLACE VIEW a_view AS SELECT * FROM a;

-- Query view a_view.
gaussdb=# SELECT * FROM a_view;
 c1 | c2
----+----
  1 |  1
(1 row)

-- Create view a_view. The view exists here, which is equivalent to redefining the view.
gaussdb=# CREATE OR REPLACE VIEW a_view AS SELECT * FROM b;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Query view a_view again.
gaussdb=# SELECT * FROM a_view;
 c1 | c2 | c3
----+----+----
  1 |  1 |  1
(1 row)

-- Rebuild a_view and drop column c1.
gaussdb=# CREATE OR REPLACE VIEW a_view AS SELECT c2, c3 FROM b;

-- Query view a_view again.
gaussdb=# SELECT * FROM a_view;
 c2 | c3
----+----
  1 |  1
(1 row)

-- Rebuild a_view and modify the type of column c2.
gaussdb=# CREATE OR REPLACE VIEW a_view AS SELECT c2::TEXT, c3 FROM b;

-- Query view a_view again.
gaussdb=# SELECT * FROM a_view;
 c2 | c3
----+----
 1  |  1
(1 row)

-- Rebuild a_view and rename column c2.
gaussdb=# CREATE OR REPLACE VIEW a_view AS SELECT c2 AS c2_new, c3 FROM b;

-- Query view a_view again.
gaussdb=# SELECT * FROM a_view;
 c2_new | c3
--------+----
      1 |  1
(1 row)

-- Drop the view and the tables.
gaussdb=# DROP VIEW a_view;
gaussdb=# DROP TABLE a;
gaussdb=# DROP TABLE b;
```

- View import

```
-- Create a view. The dependency base table test_tb1 does not exist.
gaussdb=# CREATE FORCE VIEW test_v1 AS SELECT * FROM test_tb1 WHERE col1 < 3;
WARNING:  View "test_v1" created with compilation errors.
DETAIL:  Relation "test_tb1" does not exist on datanode.
WARNING:  View "test_v1" created with compilation errors.
DETAIL:  Relation "test_tb1" does not exist on datanode.
CREATE VIEW
-- Import the view. The operation is successful, but the view status is invalid.
gaussdb=# SELECT c.relname, o.valid FROM pg_object o JOIN pg_class c ON c.oid = o.object_oid
WHERE c.relname = 'test_v1';
 relname | valid
---------+-------
 test_v1 | f
(1 row)

-- Create table test_tb1 and recompile view test_v1. The operations are successful.
gaussdb=# CREATE TABLE test_tb1(col1 int, col2 int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'x' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# ALTER VIEW test_v1 COMPILE;
ALTER VIEW
gaussdb=# SELECT c.relname, o.valid FROM pg_object o JOIN pg_class c ON c.oid = o.object_oid
WHERE c.relname = 'test_v1';
 relname | valid
---------+-------
 test_v1 | t
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)

-- Query the view.
gaussdb=# SELECT * FROM test_v1;
 col1 | col2
------+------
(0 rows)

-- Drop the table and view.
gaussdb=# DROP VIEW test_v1;
DROP VIEW
gaussdb=# DROP TABLE test_tb1;
DROP TABLE
```

- Temporary view

```
-- Create a table and a temporary view.
gaussdb=# CREATE TABLE test_tb2(c1 int, c2 int);
gaussdb=# CREATE TEMP VIEW test_v2 AS SELECT * FROM test_tb2;
-- Query the table and view information. (The temporary table belongs to a schema starting with
pg_temp instead of public.)
gaussdb=# \dv
                    List of relations
         Schema          |  Name   | Type | Owner | Storage
-------------------------+---------+------+-------+---------
 pg_temp_cn_5001_1_2_1742 | test_v2 | view | Mike  |
(1 row)

-- After the session ends, log in to the system again and check whether the temporary view is
dropped.
gaussdb=# \dv
No relations found.

-- Drop the view and the table.
gaussdb=# DROP TABLE test_tb2;
```

- Insert, update, and drop views.

```
-- Create a single table view.
gaussdb=# CREATE TABLE t_view_iud1 (x int, y int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'x' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# INSERT INTO t_view_iud1 VALUES (11, 11);
INSERT 0 1
gaussdb=# CREATE VIEW vt AS SELECT * FROM t_view_iud1;
CREATE VIEW
gaussdb=# CREATE VIEW vt_wco AS SELECT * FROM t_view_iud1 WHERE x > 5 WITH CHECK OPTION;
CREATE VIEW

-- Use information_schema.columns to check whether columns in a view can be updated.
gaussdb=# SELECT table_schema, table_name, column_name, is_updatable FROM
information_schema.columns WHERE table_schema = current_schema AND table_name = 'vt';
 table_schema | table_name | column_name | is_updatable
--------------+------------+-------------+--------------
 public       | vt         | y           | YES
 public       | vt         | x           | YES
(2 rows)

-- Insert and update data in a view. If WITH CHECK OPTION is specified when a view is created, data
is checked before the view is updated.
gaussdb=# INSERT INTO vt VALUES (1, 1);
INSERT 0 1
gaussdb=# INSERT INTO vt_wco VALUES (1, 1);
ERROR:  New row violates WITH CHECK OPTION for view "vt_wco".
DETAIL:  Failing row contains (1, 1).
gaussdb=# UPDATE vt SET y = 121 WHERE y = 1;
UPDATE 1
gaussdb=# UPDATE vt_wco SET y = 6 WHERE y = 11;
UPDATE 1
gaussdb=# DELETE FROM vt WHERE y =11;
DELETE 0
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a multi-layer nested view and specify WITH CHECK OPTION in the bottom-layer relationship. Check the corresponding conditions when DML operations are performed on the top-layer view.
gaussdb=# CREATE VIEW vvtt AS SELECT * FROM vt_wco;
CREATE VIEW
gaussdb=# INSERT INTO vvtt VALUES (1, 1), (2, 2);
ERROR:  New row violates WITH CHECK OPTION for view "vt_wco".
DETAIL:  Failing row contains (1, 1).

-- View information_schema.tables and information_schema.views and check whether they can be inserted and updated.
gaussdb=# SELECT table_schema, table_name, is_insertable_into
          FROM information_schema.tables
          WHERE table_schema = current_schema AND table_name = 'vvtt';
 table_schema | table_name | is_insertable_into
--------------+------------+--------------------
 public       | vvtt       | YES
(1 row)

gaussdb=# SELECT table_name, is_updatable, check_option
          FROM information_schema.views
          WHERE table_schema = current_schema AND table_name = 'vvtt';
 table_name | is_updatable | check_option
------------+--------------+--------------
 vvtt       | YES          | NONE
(1 row)

gaussdb=# SELECT table_name, is_updatable, check_option FROM information_schema.views WHERE table_schema = current_schema AND table_name = 'vt_wco';
 table_name | is_updatable | check_option
------------+--------------+--------------
 vt_wco     | YES          | CASCADED
(1 row)

-- Drop the view and table.
gaussdb=# DROP VIEW vvtt, vt, vt_wco CASCADE;
DROP VIEW
gaussdb=# DROP TABLE t_view_iud1;
DROP TABLE
```

- The equivalence class (that is, combining JOIN and WHERE condition) cannot be used to determine whether a target table is a key-preserved table.

```
-- Create a test table.
gaussdb=# CREATE TABLE A(col1 int, col2 int, col3 int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'col1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# CREATE TABLE B(col1 int, col2 int, col3 int unique);
NOTICE:  CREATE TABLE / UNIQUE will create implicit index "b_col3_key" for table "b"
CREATE TABLE
gaussdb=# CREATE UNIQUE INDEX index_1 ON A(col1, col2);
CREATE INDEX

-- Execute the update statement. Because A.col1 = B.col1 cannot determine that table A is a key-preserved table, the update fails.
gaussdb=# UPDATE (
gaussdb(#     SELECT A.col1, A.col2, A.col3
gaussdb(#     FROM A INNER JOIN B ON A.col1 = B.col1
gaussdb(#     WHERE B.col3 = 3
gaussdb(# )
gaussdb-# SET col1 = 1, col2 = 2, col3 = 3;
ERROR:  Failed to modify a column which maps to a non key-preserved table.

-- Drop the test table.
gaussdb=# DROP TABLE A;
DROP TABLE
gaussdb=# DROP TABLE B;
DROP TABLE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

ALTER VIEW and **DROP VIEW**

1.13.9.56 CREATE WEAK PASSWORD DICTIONARY

Description

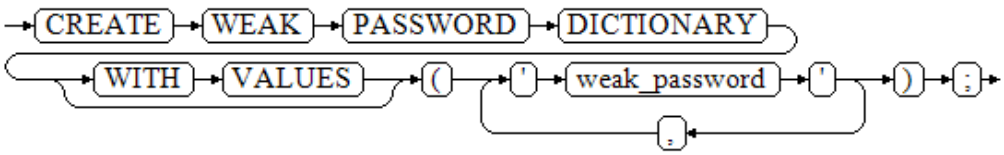
Creates a weak password dictionary. The weak password dictionary is empty by default. You can use this syntax to add one or more weak passwords to the gs_global_config system catalog.

Precautions

- Only the initial user, system administrators, and security administrators can execute it.
- Passwords in the weak password dictionary are stored in the gs_global_config system catalog.
- The weak password dictionary is empty by default. You can use this syntax to add one or more weak passwords to the gs_global_config system catalog.
- When a user attempts to execute this syntax to insert a weak password that already exists in the gs_global_config catalog, only one weak password is retained in the table.

Syntax

```
CREATE WEAK PASSWORD DICTIONARY
    [WITH VALUES] ( {'weak_password'} [, ...] );
```



Parameters

weak_password

Specifies a weak password.

Value range: character strings.

Examples

```
-- Insert a single weak password into the gs_global_config system catalog.
gaussdb=# CREATE WEAK PASSWORD DICTIONARY WITH VALUES ('********');

-- Check weak passwords in the gs_global_config system catalog.
gaussdb=# SELECT * FROM gs_global_config WHERE NAME LIKE 'weak_password';
     name       |    value
----------------+--------------
 weak_password   | ********
(1 rows)

-- Insert multiple weak passwords into the gs_global_config system catalog.
gaussdb=# CREATE WEAK PASSWORD DICTIONARY WITH VALUES ('********'),('********');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Check the weak password in the gs_global_config system catalog again. The asterisk (*) in a weak
password does not represent the password content.
gaussdb=# SELECT * FROM gs_global_config WHERE NAME LIKE 'weak_password';
     name       |     value
----------------+----------------
 weak_password   | ********
 weak_password   | ********
 weak_password   | ********
(3 rows)

-- Clear all weak passwords in the gs_global_config system catalog.
gaussdb=# DROP WEAK PASSWORD DICTIONARY;

-- View existing weak passwords.
gaussdb=# SELECT * FROM gs_global_config WHERE NAME LIKE 'weak_password';
 name | value
------+-------
(0 rows)
```

Helpful Links

DROP WEAK PASSWORD DICTIONARY

1.13.9.57 CURSOR

Description

Creates a cursor to retrieve specified rows of data from a query.

To process SQL statements, the stored procedure thread assigns a memory segment to store context association. Cursors are handles or pointers pointing to context areas. With cursors, stored procedures can control alterations in context areas.

Precautions

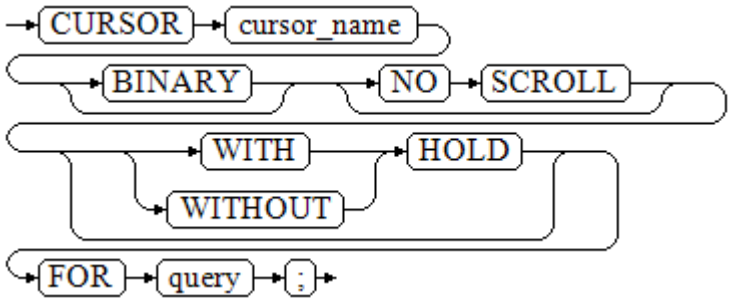
- CURSOR can only be in transaction blocks.
- Generally, CURSOR and SELECT both have text returns. Since data is stored in binary format in the system, the system needs to convert the data from the binary format to the text format. If data is returned in text format, client applications need to convert the data back to the binary format for processing. FETCH implements conversion between binary data and text data.
- Be cautious when using binary cursors. Text usually occupies larger space than binary data. A binary cursor returns internal binary data, which is easier to operate. A text cursor returns text, which is easier to retrieve and therefore reduces workload on the client. As an example, if a query returns a value of one from an integer column, you would get a string of 1 with a default cursor, whereas with a binary cursor you would get a 4-byte value containing the internal representation of the value (in big-endian byte order).
- Parameters can be defined for static cursors. The parameters support default values. However, the default value of a parameter cannot be a variable outside the package. You can define a variable in the package, assign the variable outside the package to the defined variable, and use the variable as the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The definition of a cursor generates a query object. Currently, packages or stored procedures do not support the creation of dependencies on query objects.

Syntax

```
CURSOR cursor_name
    [ BINARY ]  [ NO SCROLL ]  [ { WITH | WITHOUT } HOLD ]
    FOR query;
```



Parameters

- **cursor_name**

  Specifies the name of the cursor to be created.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **BINARY**

  Specifies that the cursor returns data in binary format rather than text format.

- **NO SCROLL**

  Specifies how the cursor retrieves rows.

  - **NO SCROLL**: specifies that the cursor cannot be used to retrieve rows in a nonsequential fashion.
  - Unspecified: Based on the query's execution plan, the system automatically determines whether the cursor can be used to retrieve rows in a nonsequential fashion.

- **WITH HOLD | WITHOUT HOLD**

  Specifies whether the cursor can continue to be used after the transaction that created it successfully commits. If neither **WITH HOLD** nor **WITHOUT HOLD** is specified, **WITHOUT HOLD** is the default.

  - **WITH HOLD**: The cursor can continue to be used after the transaction that created it successfully commits.
  - **WITHOUT HOLD**: The cursor cannot be used outside of the transaction that created it. It will be automatically closed when the transaction ends.

  Cross-node transactions (like those created with DDL statements in a cluster deployed with multiple Coordinators) do not support **WITH HOLD**.

- **query**

  Uses the SELECT or VALUES clause to specify the rows to be returned by a cursor.

  Value range: SELECT or VALUES clause

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

Example 1:

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE tbl_test(c1 int);
gaussdb=# INSERT INTO tbl_test VALUES (generate_series(1,20));

-- Set up cursor1.
gaussdb=# BEGIN;
gaussdb=# CURSOR cursor1 FOR SELECT * FROM tbl_test ORDER BY 1;

-- Run the FETCH command to retrieve three rows of data.
gaussdb=# FETCH FORWARD 3 FROM cursor1;
 c1
----
  1
  2
  3
(3 rows)

-- Run the MOVE command to move the cursor backwards by two rows. No result is returned.
gaussdb=# MOVE FORWARD 2 FROM cursor1;
MOVE 2

-- Run the FETCH command to retrieve two rows of data.
gaussdb=# FETCH FORWARD 2 FROM cursor1;
 c1
----
  6
  7
(2 rows)

-- Close the cursor and end the transaction.
gaussdb=# CLOSE cursor1;
gaussdb=# END;
```

Example 2: Cursor with the WITH HOLD attribute

```
-- Set up a WITH HOLD cursor named cursor2.
gaussdb=# BEGIN;
gaussdb=# CURSOR cursor2 WITH HOLD FOR SELECT * FROM tbl_test ORDER BY 1;

-- Run the FETCH command to retrieve three rows of data.
gaussdb=# FETCH FORWARD 3 FROM cursor2;
 c1
----
  1
  2
  3
(3 rows)

-- Different from common cursors, cursors with the WITH HOLD attribute can still be used after a
transaction ends.
gaussdb=# END;
gaussdb=# FETCH FORWARD 2 FROM cursor2;
 c1
----
  4
  5
(2 rows)

-- Close the cursor.
gaussdb=# CLOSE cursor2;

-- Drop.
gaussdb=# DROP TABLE tbl_test;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

FETCH, **CLOSE**, and **MOVE**

1.13.10 D

1.13.10.1 DEALLOCATE

Description

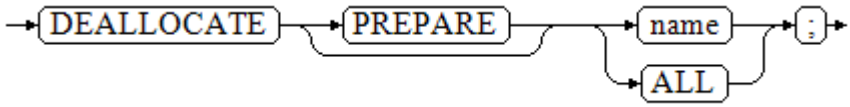
Deallocates a prepared statement from the database.

Precautions

You can use DEALLOCATE to manually deallocate prepared statements created using PREPARE. If not explicitly deallocated, these statements will be automatically cleared when the session ends.

Syntax

```
DEALLOCATE [ PREPARE ] { name | ALL };
```



Parameters

- **PREPARE**

  (Optional) It is provided solely for syntax compatibility and will be ignored during actual execution.

- **name**

  Specifies the name of the prepared statement to be deallocated.

- **ALL**

  Deallocates all prepared statements.

Examples

```
-- View existing prepared statements.
gaussdb=# SELECT name, statement, parameter_types FROM pg_prepared_statements;
 name | statement | parameter_types
------+-----------+-----------------
(0 rows)

-- Create four prepared statements q1, q2, q3, and q4.
gaussdb=# PREPARE q1 AS SELECT 1 AS a;
PREPARE
gaussdb=# PREPARE q2 AS SELECT 1 AS a;
PREPARE
gaussdb=# PREPARE q3 AS SELECT 1 AS a;
PREPARE
gaussdb=# PREPARE q4 AS SELECT 1 AS a;
PREPARE

-- View existing prepared statements again.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT name, statement, parameter_types FROM pg_prepared_statements;
 name |          statement          | parameter_types
------+-----------------------------+-----------------
 q1   | PREPARE q1 AS SELECT 1 AS a; | {}
 q4   | PREPARE q4 AS SELECT 1 AS a; | {}
 q3   | PREPARE q3 AS SELECT 1 AS a; | {}
 q2   | PREPARE q2 AS SELECT 1 AS a; | {}
(4 rows)

-- Deallocate the prepared statement q4 and view the remaining prepared statements.
gaussdb=# DEALLOCATE q4;
gaussdb=# SELECT name, statement, parameter_types FROM pg_prepared_statements;
 name |          statement          | parameter_types
------+-----------------------------+-----------------
 q1   | PREPARE q1 AS SELECT 1 AS a; | {}
 q3   | PREPARE q3 AS SELECT 1 AS a; | {}
 q2   | PREPARE q2 AS SELECT 1 AS a; | {}
(3 rows)

-- Deallocate all prepared statements and view the remaining prepared statements.
gaussdb=# DEALLOCATE ALL;
DEALLOCATE ALL
gaussdb=# SELECT name, statement, parameter_types FROM pg_prepared_statements;
 name | statement | parameter_types
------+-----------+-----------------
(0 rows)
```

1.13.10.2 DECLARE

Description

Defines a cursor to retrieve a small number of rows at a time out of a larger query or declares an anonymous block (DECLARE must lead the statement of the anonymous block).

This section describes usage of cursors. The usage of anonymous blocks is available in **BEGIN**.

To process SQL statements, the stored procedure thread allocates a memory area to store context relationships. A cursor is a handle or pointer that points to the context area. With cursors, stored procedures can control alterations in context areas.

Normal cursors return data in text format, the same as a SELECT would produce. Since data is stored in binary format in the system, the system needs to convert the data from the binary format to the text format. If data is returned in text format, client applications need to convert the data back to the binary format for processing. Cursors can return data in either text or binary format using FETCH.

Precautions

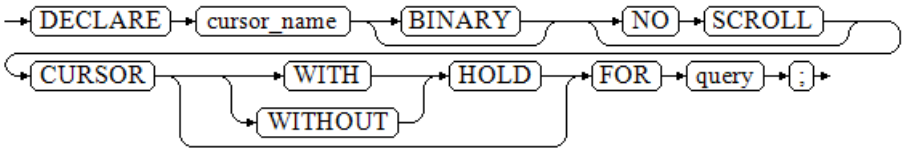
- CURSOR is used only in transaction blocks.
- Be cautious when using binary cursors. Text usually occupies larger space than binary data. A binary cursor returns internal binary data, which is easier to operate. A text cursor returns text, which is easier to retrieve and therefore reduces workload on the client. As an example, if a query returns a value of one from an integer column, you would get a string of 1 with a default cursor, whereas with a binary cursor you would get a 4-byte value containing the internal representation of the value (in big-endian byte order).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax

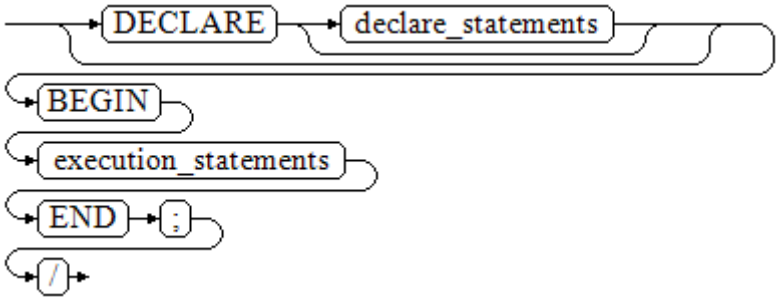
- Define a cursor.

```
DECLARE cursor_name [ BINARY ] [ NO SCROLL ]
    CURSOR [ { WITH | WITHOUT } HOLD ] FOR query;
```



- Enable an anonymous block.

```
[DECLARE [declare_statements]]
BEGIN
execution_statements
END;
/
```



Parameters

- **cursor_name**

  Specifies the name of the cursor to be created.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **BINARY**

  Specifies that the cursor returns data in binary rather than text format.

- **NO SCROLL**

  Specifies how the cursor retrieves rows.

  – **NO SCROLL**: The cursor cannot be used to retrieve rows in a nonsequential fashion.

  – Unspecified: Based on the query's execution plan, the system automatically determines whether the cursor can be used to retrieve rows in inverse order.

- **WITH HOLD | WITHOUT HOLD**

  Specifies whether the cursor can continue to be used after the transaction that created it successfully commits. If neither **WITH HOLD** nor **WITHOUT HOLD** is specified, **WITHOUT HOLD** is the default.

  – **WITH HOLD**: The cursor can continue to be used after the transaction that created it successfully commits.

  – **WITHOUT HOLD**: The cursor cannot be used outside of the transaction that created it. It will be automatically closed when the transaction commits.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

For a cursor declared as **WITH HOLD**, all its data is cached when the transaction ends. If the cursor has a large amount of data, this caching process may take a long time.

- **query**

  Uses the SELECT or VALUES clause to specify the rows to be returned by a cursor.

  Value range: SELECT or VALUES clause.

- **declare_statements**

  Declares a variable, including its name and type, for example, **sales_cnt int**.

- **execution_statements**

  Specifies the statement to be executed in an anonymous block.

  Value range: names of existing functions.

Examples

- Define a cursor.

```
-- Create the test table and insert 10 data records into the table.
gaussdb=# CREATE TABLE test(c1 int, c2 int);
gaussdb=# INSERT INTO test VALUES (generate_series(1,10),generate_series(1,10));

-- Start a transaction and create a cursor named cursor1.
gaussdb=# START TRANSACTION;
gaussdb=# DECLARE cursor1 CURSOR FOR SELECT * FROM test ORDER BY 1;

-- The cursor retrieves three rows of data from the associated position.
gaussdb=# FETCH FORWARD 3 FROM cursor1;
 c1 | c2
----+----
  1 |  1
  2 |  2
  3 |  3
(3 rows)

-- Close the cursor and commit the transaction.
gaussdb=# CLOSE cursor1;
gaussdb=# END;

-- Drop a table.
gaussdb=# DROP TABLE test;
```

- Enable an anonymous block.

```
-- Declare and output a variable.
gaussdb=# DECLARE num INTEGER := 10;
BEGIN
    dbe_output.print_line(num);
END;
/
```

Helpful Links

BEGIN and **FETCH**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.10.3 DELETE

Description

Deletes rows that satisfy the WHERE clause from the specified table. If the WHERE clause is absent, the effect is to delete all rows in the table. The result is a valid, but an empty table.

Precautions

- The owner of a table, users granted the DELETE permission on the table or the DELETE ANY TABLE permission can delete data from the table. When separation of duties is disabled, system administrators have this permission by default. To perform the DELETE operation, you must have the SELECT permission on the table referenced by the USING clause and the table read in the condition.
- For row-store tables, DELETE can be used if the tables have primary key constraints or the execution plan can be pushed down.
- For a DELETE statement with a subquery executed as a STREAM plan, UPDATE cannot be performed on the deleted row data.

Syntax

```
[ WITH [ RECURSIVE ] with_query [, ...] ]
DELETE [/*+ plan_hint */] [FROM] [ ONLY ] {table_name [ * ] subquery [ [ AS ] alias ] | view_name [ [ AS ] alias ]}
    [ USING using_list ]
    [ WHERE condition | WHERE CURRENT OF cursor_name ]
    [ ORDER BY {expression [  ASC | DESC | USING operator ] } ]
    [ LIMIT row_count ]
    [ RETURNING { * | { output_expr [ [ AS ] output_name ] } [, ...] } ];
```

- Format of **with_query**:

  ```
  with_query_name [ ( column_name [, ...] ) ] AS [ [ NOT ] MATERIALIZED ]
  ( {select | values | insert | update | delete} )
  ```

- Format of **subquery**:

  ```
  [ WITH [ RECURSIVE ] with_query [, ...] ]
  SELECT [/*+ plan_hint */] [ ALL ]
  { * | {expression [ [ AS ] output_name ]} [, ...] }
  [ into_option ]
  [ FROM from_item [, ...] ]
  [ WHERE condition ]
  [ [ START WITH condition ] CONNECT BY [NOCYCLE] condition [ ORDER SIBLINGS BY expression ] ]
  [ ORDER BY {expression [ [ ASC | DESC | USING operator ] | nlssort_expression_clause ] [ NULLS
  { FIRST | LAST } ]} [, ...] ]
  [ OFFSET start [ ROW | ROWS ] ]
  [ FETCH { FIRST | NEXT } [ count ] { ROW | ROWS } ONLY ]
  [ into_option ];
  ```

  The specified subquery source **from_item** is as follows:

  ```
  {[ ONLY ] {table_name | view_name} [ * ] [ partition_clause ] [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
  |( select ) [ AS ] alias [ ( column_alias [, ...] ) ]
  |with_query_name [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
  |from_item [ NATURAL ] join_type from_item [ ON join_condition | USING ( join_column [, ...] ) ]}
  ```

Parameters

- **WITH [RECURSIVE] with_query [, ...]**

  Specifies one or more subqueries that can be referenced by name in the primary query. The subqueries effectively act as temporary tables. If

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

RECURSIVE is specified, it allows a SELECT subquery to reference itself by name.

- **with_query_name** specifies the name of the result set generated by the subquery. The name can be used to access the result set of the subquery in the query.
- **column_name** specifies the column names displayed in the subquery result set.
- Each subquery can be a SELECT, VALUES, INSERT, UPDATE, or DELETE statement.
- You can use MATERIALIZED or NOT MATERIALIZED to modify the CTE.
  - If **MATERIALIZED** is specified, the WITH query will be materialized, and a copy of the subquery result set is generated. The copy is directly queried at the reference point. Therefore, the WITH subquery cannot be jointly optimized with the SELECT statement trunk (for example, predicate pushdown and equivalence class transfer). In this scenario, you can use **NOT MATERIALIZED** for modification. If the WITH query can be executed as a subquery inline, the preceding optimization can be performed.
  - If the user does not explicitly specify the materialized attribute, comply with the following rules: If the CTE is referenced only once in the trunk statement to which it belongs and semantically supports inline execution, it will be rewritten as subquery inline execution. Otherwise, the materialized execution will be performed in CTE Scan mode.

- **plan_hint**

  Follows the DELETE keyword in the /*+ *plan_hint* */ format. It is used to optimize the plan of a DELETE statement block using hints. For details, see "Hint-based Optimization" in *SQL Optimization Guide*. Only the first /*+ *plan_hint* */ comment block in each statement takes effect as a hint. Multiple hints can be written in the comment block.

- **ONLY**

  If **ONLY** is specified, only that table is deleted. If **ONLY** is not specified, that table and all its subtables are deleted.

- **table_name**

  Specifies the name (optionally schema-qualified) of the target table.

  Value range: existing table names.

  NOTE

  You can use database links to perform operations on remote tables. For details, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.

- **subquery**

  The object to be deleted can be a subquery. When data in a subquery is deleted, the subquery is regarded as a temporary view. The **CHECK OPTION** option can be added to the end of the subquery.

  If there is only one table in the subquery, the table is deleted. If there are multiple tables or nested tables in the subquery, the system determines whether the table can be deleted by checking whether there is a key-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

preserved table. For details about key-preserved tables and **WITH CHECK OPTION**, see **CREATE VIEW**.

- **view_name**

  Specifies name of the target view.

  NOTE

  The restrictions on deleting views and subqueries are as follows:

  - The DELETE operation can be performed only on columns that directly reference user columns in the base table.
  - A subquery or view must contain at least one updatable column. For details about updatable columns, see **CREATE VIEW**.
  - The view or subquery must not contain the DISTINCT, GROUP BY, HAVING, LIMIT, or OFFSET clause at the top layer.
  - Views and subqueries that contain set operations (UNION, INTERSECT, EXCEPT, and MINUS) at the top layer are not supported.
  - Views and subqueries whose target lists contain aggregate functions, window functions, or return set functions (such as array_agg, json_agg, and generate_series) are not supported.
  - Views with BEFORE or AFTER triggers but without INSTEAD OF triggers or INSTEAD rules are not supported.
  - Table types supported in views and subqueries include ordinary tables, temporary tables, global temporary tables, partitioned tables, subpartitioned tables, Ustore tables, and Astore tables.
  - In a join view or subquery, only data in key-preserved tables can be deleted. If there is only one key-preserved table, the data in the table is deleted. If there are multiple key-preserved tables, only the data in the first key-preserved table following FROM is deleted.
  - If the **CHECK OPTION** option is specified in the join view or subquery, the base table is duplicate, and the duplicate base table has attributes inconsistent with the key-preserved table in the view or subquery, rows cannot be deleted from the join view or subquery. For details about the key-preserved table, see **CREATE VIEW**.
  - The DELETE operation cannot be performed on the system view.
  - Deleting multiple tables is not supported.

- **alias**

  Specifies the alias of the target table.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **using_list**

  Specifies the USING clause.

- **condition**

  Specifies an expression that returns a Boolean value, which is used to determine the row to be deleted. You are advised not to use numeric types such as INT as conditions because such types can be implicitly converted to Boolean values (non-zero values are implicitly converted to **true** and **0** is implicitly converted to **false**), which may cause unexpected results.

- **WHERE CURRENT OF cursor_name**

  When the cursor points to a row in a table, you can use this syntax to delete the row. For details about restrictions, see **UPDATE**.

- **ORDER BY**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Refer to **SELECT**.

- **LIMIT**

  Refer to **SELECT**.

  NOTE

  If you want to perform DELETE on a replication table, the LIMIT keyword cannot be used to specify the table.

- **output_expr**

  Specifies an expression to be computed and returned by the DELETE statement after each row is deleted. The expression can use any column names of the table. Write * to return all columns.

- **output_name**

  Specifies a name to use for a returned column.

  Value range: character strings complying with **Identifier Naming Conventions**.

Examples

- Deleting some data records

```
-- Create a table.
gaussdb=# CREATE TABLE test_t1(col1 INT,col2 INT);
gaussdb=# INSERT INTO test_t1 VALUES (1, 1), (2, 2), (3, 3), (4, 4), (4, 6);

-- Delete some data records from the table.
gaussdb=# DELETE FROM test_t1 WHERE col1 = 4;

-- Query.
gaussdb=# SELECT * FROM test_t1;
 col1 | col2
------+------
    1 |    1
    2 |    2
    3 |    3
(3 rows)
```

- Deleting all data

```
-- Delete all data.
gaussdb=# DELETE FROM test_t1;

-- Query.
gaussdb=# SELECT * FROM test_t1;
 col1 | col2
------+------
(0 rows)

-- Drop the table.
gaussdb=# DROP TABLE test_t1;
```

- WITH [RECURSIVE] with_query [, ...]

```
-- Create a student table.
gaussdb=# CREATE TABLE student(id INT,name varchar(50));

-- Create a score table.
gaussdb=# CREATE TABLE grade(id INT,score CHAR);

gaussdb=# INSERT INTO student VALUES (1, 'tom'), (2, 'jerry'), (3, 'david');
gaussdb=# INSERT INTO grade VALUES (1, 'A'), (2, 'B'), (3, 'b');

-- Delete the data of the student whose id is 2 from the student table and the grade table.
gaussdb=# WITH del_stu AS(DELETE FROM student WHERE id = 2 RETURNING id)
    DELETE FROM grade WHERE id = (SELECT id FROM del_stu);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Query data.
gaussdb=# SELECT * FROM student;
 id | name
----+-------
  1 | tom
  3 | david
(2 rows)

gaussdb=# SELECT * FROM grade;
 id | score
----+-------
  1 | A
  3 | b
(2 rows)

-- Drop the table.
gaussdb=# DROP TABLE grade;
gaussdb=# DROP TABLE student;
```

- Deleting a subquery

```
-- Create a schema.
gaussdb=# CREATE SCHEMA del_subqry;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = 'del_subqry';
SET

-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE t1 (x1 int, y1 int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'x1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# CREATE TABLE t2 (x2 int PRIMARY KEY, y2 int);
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "t2_pkey" for table "t2"
CREATE TABLE
gaussdb=# CREATE TABLE tdata (x INT PRIMARY KEY, y INT);
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "tdata_pkey" for table "tdata"
CREATE TABLE
gaussdb=# CREATE TABLE tinfo (z INT PRIMARY KEY, comm VARCHAR2(20));
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "tinfo_pkey" for table "tinfo"
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES (1, 1), (2, 2), (3, 3), (5, 5);
INSERT 0 4
gaussdb=# INSERT INTO t2 VALUES (1, 1), (2, 2), (3, 3), (5, 5);
INSERT 0 4
gaussdb=# INSERT INTO tdata VALUES (1, 1), (2, 2), (3, 3);
INSERT 0 3
gaussdb=# INSERT INTO tinfo VALUES (1,'one'), (2, 'two'), (3, 'three'), (5, 'wrong three');
INSERT 0 4

-- Delete data from t1 using a subquery.
gaussdb=# DELETE FROM (SELECT * FROM t1) where y1 = 3;
DELETE 1

-- The subquery is read-only, and data cannot be deleted.
gaussdb=# DELETE FROM (SELECT * FROM t1 WITH READ ONLY) WHERE y1 = 1;
ERROR:  DML operation cannot be performed on a read-only view.

-- Delete a subquery of a multi-table join.
gaussdb=# SELECT * FROM t1, t2 WHERE x1 = x2;
 x1 | y1 | x2 | y2
----+----+----+----
  1 |  1 |  1 |  1
  2 |  2 |  2 |  2
  5 |  5 |  5 |  5
(3 rows)

gaussdb=# DELETE FROM (SELECT * FROM t1, t2 WHERE x1 = x2) WHERE y2 = 5;
DELETE 1
gaussdb=# SELECT * FROM t1, t2 WHERE x1 = x2;
 x1 | y1 | x2 | y2
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
----+----+----+----
 1 | 1 | 1 | 1
 2 | 2 | 2 | 2
(2 rows)

-- The subquery contains CHECK OPTION, and the tdata table is duplicate. td1 is not a key-preserved table, and td2 is a key-preserved table.
gaussdb=# DELETE FROM (SELECT td1.x x1, td1.y y1, td2.x x2, td2.y y2 FROM tdata td1, tdata td2, tinfo WHERE td2.y=tinfo.z AND td1.x=td2.y WITH CHECK OPTION) WHERE y1 = 2;
ERROR:  Failed to delete from view with no key-preserved table.

-- If CHECK OPTION is not specified, a subquery with the same structure is created and deleted successfully.
gaussdb=# DELETE FROM (SELECT td1.x x1, td1.y y1, td2.x x2, td2.y y2 FROM tdata td1, tdata td2, tinfo WHERE td2.y=tinfo.z AND td1.x=td2.y) WHERE y1 = 2;
DELETE 1
-- Drop a schema.
gaussdb=# RESET CURRENT_SCHEMA;
RESET
gaussdb=# DROP SCHEMA del_subqry CASCADE;
NOTICE:  drop cascades to 5 other objects
DETAIL:  drop cascades to table del_subqry.t1
drop cascades to table del_subqry.t2
drop cascades to table del_subqry.tdata
drop cascades to table del_subqry.tinfo
drop cascades to view del_subqry.vv_wco
DROP SCHEMA
```

- Dropping a view

```
-- Create a schema.
gaussdb=# CREATE SCHEMA del_view;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = 'del_view';
SET

-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE t1 (x1 int, y1 int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'x1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# CREATE TABLE t2 (x2 int PRIMARY KEY, y2 int);
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "t2_pkey" for table "t2"
CREATE TABLE
gaussdb=# CREATE TABLE tdata (x INT PRIMARY KEY, y INT);
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "tdata_pkey" for table "tdata"
CREATE TABLE
gaussdb=# CREATE TABLE tinfo (z INT PRIMARY KEY, comm VARCHAR2(20));
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "tinfo_pkey" for table "tinfo"
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES (1, 1), (2, 2), (3, 3), (5, 5);
INSERT 0 4
gaussdb=# INSERT INTO t2 VALUES (1, 1), (2, 2), (3, 3), (5, 5);
INSERT 0 4
gaussdb=# INSERT INTO tdata VALUES (1, 1), (2, 2), (3, 3);
INSERT 0 3
gaussdb=# INSERT INTO tinfo VALUES (1,'one'), (2, 'two'), (3, 'three'), (5, 'wrong three');
INSERT 0 4

-- Create a single table view.
gaussdb=# CREATE VIEW v_del1 AS SELECT * FROM t1;
CREATE VIEW
gaussdb=# CREATE VIEW v_del_read AS SELECT * FROM t1 WITH READ ONLY;
CREATE VIEW

-- Delete data from t1 using a view.
gaussdb=# DELETE FROM v_del1 where y1 = 3;
DELETE 1

-- The view is read-only, and data cannot be deleted.
gaussdb=# DELETE FROM v_del_read WHERE y1 = 1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ERROR:  DML operation cannot be performed on a read-only view.

-- Create a multi-table join view.
gaussdb=# CREATE VIEW vvt1t2 AS SELECT * FROM t1, t2 WHERE x1 = x2;
CREATE VIEW
gaussdb=# CREATE VIEW vv_dup AS SELECT td1.x x1, td1.y y1, td2.x x2, td2.y y2 FROM tdata td1,
tdata td2, tinfo WHERE td2.y=tinfo.z AND td1.x=td2.y;
CREATE VIEW
gaussdb=# CREATE VIEW vv_dup_wco AS SELECT td1.x x1, td1.y y1, td2.x x2, td2.y y2 FROM tdata td1,
tdata td2, tinfo WHERE td2.y=tinfo.z AND td1.x=td2.y WITH CHECK OPTION;
CREATE VIEW

-- Delete the multi-table join view.
gaussdb=# SELECT * FROM vvt1t2;
 x1 | y1 | x2 | y2
----+----+----+----
  1 |  1 |  1 |  1
  2 |  2 |  2 |  2
  5 |  5 |  5 |  5
(3 rows)

gaussdb=# DELETE FROM vvt1t2 WHERE y2 = 5;
DELETE 1

gaussdb=# SELECT * FROM vvt1t2;
 x1 | y1 | x2 | y2
----+----+----+----
  1 |  1 |  1 |  1
  2 |  2 |  2 |  2
(2 rows)

-- The view contains CHECK OPTION, and the tdata table is duplicate. td1 is not a key-preserved
table, and td2 is a key-preserved table.
gaussdb=# DELETE FROM vv_dup_wco WHERE y1 = 2;
ERROR:  Failed to delete from view with no key-preserved table.

-- If CHECK OPTION is not specified, a view with the same structure is created and deleted
successfully.
gaussdb=# DELETE FROM vv_dup WHERE y1 = 2;
DELETE 1

-- Drop a schema.
gaussdb=# RESET CURRENT_SCHEMA;
RESET
gaussdb=# DROP SCHEMA del_view CASCADE;
NOTICE:  drop cascades to 9 other objects
DETAIL:  drop cascades to table del_view.t1
drop cascades to table del_view.t2
drop cascades to table del_view.tdata
drop cascades to table del_view.tinfo
drop cascades to view del_view.v_del1
drop cascades to view del_view.v_del_read
drop cascades to view del_view.vvt1t2
drop cascades to view del_view.vv_dup
drop cascades to view del_view.vv_dup_wco
DROP SCHEMA
```

Suggestions

To delete all records in a table, use the **TRUNCATE** syntax.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.10.4 DO

Description

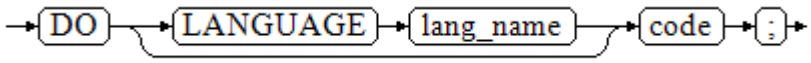
Executes an anonymous code block. A code block is regarded as a function body without parameters. The return type is VOID. It is parsed and executed at the same time.

Precautions

- Before using a procedural language, you must install it in the current database by running the **CREATE LANGUAGE** command. PL/pgSQL is installed by default. To install another language, you must specify it.
- To use an untrusted language, you must have the USAGE permission on the programming language or the SYSADMIN permission.

Syntax

```
DO [ LANGUAGE lang_name ] code;
```



Parameters

- **lang_name**

  Specifies the name of the programming language that is used to parse the code. If no language is specified, the default language is **plpgsql**.

- **code**

  Specifies the programming language code that can be executed. The value must be a character string.

Examples

```
-- Create the webuser user.
gaussdb=# CREATE USER webuser PASSWORD '********';

-- Grant all permissions on all views in the tpcds schema to the webuser user.
gaussdb=# DO $$DECLARE r record;
BEGIN
    FOR r IN SELECT c.relname,n.nspname FROM pg_class c,pg_namespace n
            WHERE c.relnamespace = n.oid AND n.nspname = 'tpcds' AND relkind IN ('r','v')
    LOOP
        EXECUTE 'GRANT ALL ON ' || quote_ident(r.table_schema) || '.' || quote_ident(r.table_name) || ' TO
webuser';
    END LOOP;
END$$;


-- Delete the webuser user.
gaussdb=# DROP USER webuser CASCADE;
```

1.13.10.5 DROP AGGREGATE

Description

Drops an aggregate function from the database.

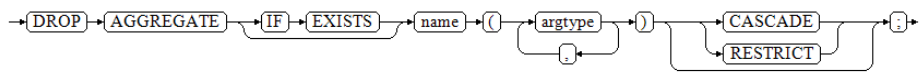
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

Only the owner of an aggregate function can perform the DROP AGGREGATE operation.

Syntax

```
DROP AGGREGATE [ IF EXISTS ] name ( argtype [ , ... ] ) [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified aggregate function exists.
  - If the aggregate function exists, it will be dropped successfully.
  - If the aggregate function does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the aggregate function does not exist.
- **name**

  Specifies the aggregate function to be dropped (optionally schema-qualified).
- **argtype**

  Specifies the input data type of the aggregate function. If the specified aggregate function is of the zero-parameter type, use an asterisk (*) instead of an input data type list.
- **CASCADE | RESTRICT**

  (Optional) Specifies whether to cascadingly drop objects that depend on the aggregate function. The default value is **RESTRICT**.
  - **CASCADE**: The database cascadingly drops objects that depend on the aggregate function.
  - **RESTRICT**: The database refuses to drop the aggregate function if any objects depend on it.

Examples

```
-- Create a user-defined function.
gaussdb=# CREATE OR REPLACE FUNCTION int_add(int,int)
 RETURNS int AS $BODY$
DECLARE
BEGIN
 RETURN $1 + $2;
END;
$BODY$ language plpgsql;

-- Create an aggregate function.
gaussdb=# CREATE AGGREGATE myavg (int)
(
    sfunc = int_add,
    stype = int,
    initcond = '0'
);

-- Drop the aggregate function myavg of the int type.
gaussdb=# DROP AGGREGATE myavg(int);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop the user-defined function.
gaussdb=# DROP FUNCTION int_add(int,int);
```

Helpful Links

ALTER AGGREGATE and **CREATE AGGREGATE**

Compatibility

The SQL standard does not provide the DROP AGGREGATE statement.

1.13.10.6 DROP AUDIT POLICY

Description

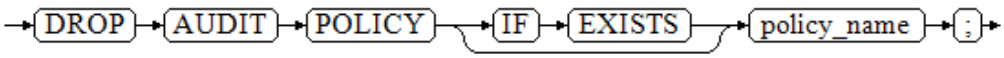
Drops an audit policy from the database.

Precautions

Only users with the POLADMIN or SYSADMIN permission, or the initial user, can perform this operation.

Syntax

```
DROP AUDIT POLICY [IF EXISTS] policy_name;
```



Parameters

- **policy_name**

  Specifies the name of an audit policy, which must be unique.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **IF EXISTS**

  Checks whether the specified audit policy exists.

  – If the audit policy exists, it will be dropped successfully.

  – If the audit policy does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the audit policy does not exist.

Examples

```
-- Create the adt1 policy.
gaussdb=# CREATE AUDIT POLICY adt1 PRIVILEGES CREATE;
CREATE AUDIT POLICY

-- Drop the audit policy adt1.
gaussdb=# DROP AUDIT POLICY adt1;
DROP AUDIT POLICY

-- When you drop the audit policy adt0 that does not exist, the system displays a message indicating that the operation fails because the audit policy does not exist.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# DROP AUDIT POLICY adt0;
ERROR:  adt0 policy does not exist, drop failed
```

Helpful Links

CREATE AUDIT POLICY and **ALTER AUDIT POLICY**

1.13.10.7 DROP CLIENT MASTER KEY

Description

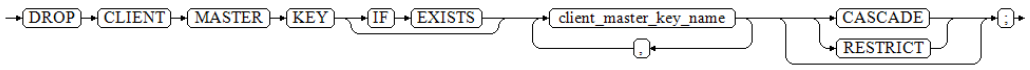
Drops a client master key (CMK).

Precautions

- Only the CMK owner and users with the DROP permission can perform this operation. System administrators have this permission by default.
- This command can only be used to drop the metadata information recorded in the system catalog of the database, but cannot be used to drop the CMK file. You need to use keytool to drop the CMK file.

Syntax

```
DROP CLIENT MASTER KEY [ IF EXISTS ] client_master_key_name [, ...] [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified CMK exists.

  - If the CMK exists, it will be dropped successfully.
  - If the CMK does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the CMK does not exist.

- **client_master_key_name**

  Specifies the name of a CMK to be dropped.

  Value range: names of existing CMKs (character strings).

- **CASCADE** | **RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the CMK when dropping the CMK itself. The default value is **RESTRICT**.

  - **CASCADE**: The database cascadingly drops objects that depend on the CMK.
  - **RESTRICT**: refuses to drop the CMK if any objects depend on it.

Helpful Links

CREATE CLIENT MASTER KEY

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.10.8 DROP COLUMN ENCRYPTION KEY

Description

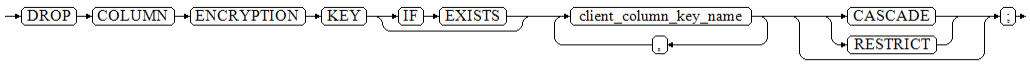
Drops a column encryption key (CEK).

Precautions

Only the CEK owner and users with the DROP permission can perform this operation. System administrators have this permission by default.

Syntax

```
DROP COLUMN ENCRYPTION KEY [ IF EXISTS ] client_column_key_name [, ...] [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified CEK exists.

  - If the CEK exists, it will be dropped successfully.
  - If the CEK does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the CEK does not exist.

- **client_column_key_name**

  Specifies the name of a CEK to be dropped.

  Value range: names of existing CEKs (character strings).

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on a CEK when dropping the CEK itself.

  For fully-encrypted databases, this syntax is high-risk operation. Actually, encrypted columns that depend on a CEK cannot be dropped.

Helpful Links

ALTER COLUMN ENCRYPTION KEY and **CREATE COLUMN ENCRYPTION KEY**

1.13.10.9 DROP DATABASE

Description

Drops a database.

Precautions

- Only the database owner and users with the DROP permission on the database can perform this operation. System administrators have this permission by default.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The default installed template and user databases (**template0**, **template1**, **templatea**, **templatem**, and **postgres**) cannot be dropped. To check databases in the current service, run the gsql command **\l**.
- If a user is currently accessing the database, the database cannot be dropped. You can check the current database connection in the DV_SESSIONS view.
- This operation cannot be performed in transaction blocks.
- Before dropping a database, run the **CLEAN CONNECTION TO ALL FORCE FOR DATABASE** *XXXX*; command to forcibly stop the existing user connections and background threads, preventing operation failures caused by running background threads.

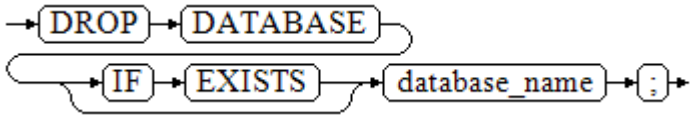
  NOTICE

  DROP DATABASE cannot be undone. Forcibly stopping background threads may cause data inconsistency in the current database. **CLEAN CONNECTION TO ALL FORCE FOR DATABASE** *XXXX*; is executed only when you are sure that the database needs to be dropped.

- If **DROP DATABASE** fails and is rolled back, run **DROP DATABASE IF EXISTS** again.

Syntax

```
DROP DATABASE [ IF EXISTS ] database_name;
```



Parameters

- **IF EXISTS**

  Checks whether the specified database exists. This parameter can be used to avoid errors.

  - If the database exists, it will be dropped successfully.
  - If the database does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the database does not exist.

- **database_name**

  Specifies the name of the database to be dropped.

  Value range: existing database names.

Examples

```
-- Create a database.
gaussdb=# CREATE DATABASE testdb1;

-- Drop a database.
gaussdb=# DROP DATABASE testdb1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

CREATE DATABASE

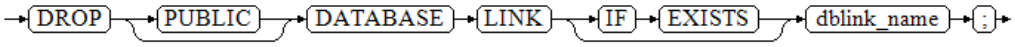
1.13.10.10 DROP DATABASE LINK

Description

Drops a database link.

Syntax

```
DROP [ PUBLIC ] DATABASE LINK [ IF EXISTS ] dblink_name;
```



Parameters

- **dblink_name**

  Specifies the name of a connection object.

- **IF EXISTS**

  Checks whether a specified database link exists.

  – If the database link exists, it will be dropped successfully.

  – If the database link does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the database link does not exist.

- **PUBLIC**

  Specifies the connection type. If **PUBLIC** is not specified, **PRIVATE** is the default.

Examples

```
-- Create an ORA-compatible database.
gaussdb=# CREATE DATABASE ora_test_db DBCOMPATIBILITY 'ORA';
CREATE DATABASE

-- Switch to another database.
gaussdb=# \c ora_test_db

-- Create a user with the SYSADMIN permission.
ora_test_db=# CREATE USER local_user WITH SYSADMIN PASSWORD '********';
CREATE ROLE
ora_test_db=# SET ROLE local_user PASSWORD '********';
SET

-- Create a private database link. host can be an IPv6 address.
ora_test_db=> CREATE DATABASE LINK private_dblink CONNECT TO 'remote_user' IDENTIFIED BY '********'
USING (host '***.***.***.***',port '*****',dbname 'remote_db');
CREATE DATABASE LINK

-- Drop the private database link.
ora_test_db=> DROP DATABASE LINK private_dblink;
DROP DATABASE LINK

-- Create a public database link. host can be an IPv6 address.
ora_test_db=> CREATE PUBLIC DATABASE LINK public_dblink CONNECT TO 'remote_user' IDENTIFIED BY
'********' USING (host '***.***.***.***',port '*****',dbname 'remote_db');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
CREATE DATABASE LINK

-- Drop the public database link.
ora_test_db=> DROP PUBLIC DATABASE LINK public_dblink;
DROP DATABASE LINK

-- Drop the created user.
ora_test_db=> RESET ROLE;
RESET
ora_test_db=# DROP USER local_user;
DROP ROLE

-- Switch back to the initial database and drop the test database. Replace postgres with the actual database name.
ora_test_db=# \c postgres
gaussdb=# DROP DATABASE ora_test_db;
DROP DATABASE
```

Helpful Links

ALTER DATABASE LINK and **CREATE DATABASE LINK**

1.13.10.11 DROP DIRECTORY

Description

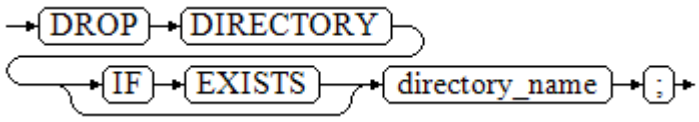
Drops a directory from the database.

Precautions

- When **enable_access_server_directory** is set to **off**, only the initial user can perform this operation.
- When **enable_access_server_directory** is set to **on**, users with the SYSADMIN permission, the directory object owner, users with the DROP permission on a directory, and users inheriting permissions from the built-in role gs_role_directory_drop can drop a directory object.

Syntax

```
DROP DIRECTORY [ IF EXISTS ] directory_name;
```



Parameters

- **directory_name**

  Specifies the directory name.

  Value range: names of existing directories.

- **IF EXISTS**

  Checks whether a specified directory exists.

  – If the directory exists, it will be dropped successfully.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - If the directory does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the directory does not exist.

Examples

```
-- Create a directory object.
gaussdb=# CREATE OR REPLACE DIRECTORY  dir  as '/tmp/';

-- Drop a directory object.
gaussdb=# DROP DIRECTORY dir;
```

Helpful Links

CREATE DIRECTORY and **ALTER DIRECTORY**

1.13.10.12 DROP EXTENSION

NOTICE

Extensions are for internal use only. Only advanced packages or extensions in the database can be dropped. Currently, only postgis, postgis_sfcgal, postgis_raster, postgis_topology, yukon_geomodel, yukon_geogridcoder, pkg_bpchar_opc, and tablefunc are supported.

Description

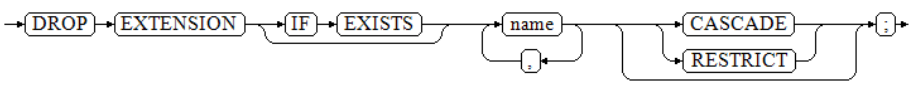
Drops an extension from the database.

Precautions

- Only the extension owner and initial user can perform this operation.
- When you drop an extension, the components that make up the extension are also dropped.

Syntax

```
DROP EXTENSION [ IF EXISTS ] name [, ...] [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified extension exists.

  - If the extension exists, it will be dropped successfully.
  - If the extension does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the extension does not exist.

- **name**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the name of an installed extension.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the extension when dropping the extension itself. If this parameter is not specified, the default value is **RESTRICT**.

  - **CASCADE**: The database cascadingly drops objects that depend on the extension.
  - **RESTRICT**: The database refuses to drop the extension if any objects depend on it (unless all member objects and other extension objects are simultaneously dropped in the same **DROP** command).

Helpful Links

CREATE EXTENSION

1.13.10.13 DROP FOREIGN DATA WRAPPER

Description

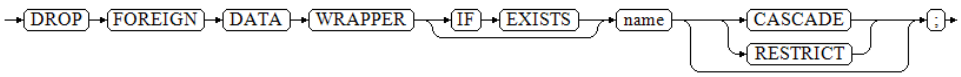
Drops a foreign data wrapper (FDW) from the database.

Precautions

- This operation can be successfully performed only when the GUC parameter **support_extended_features** is set to **on**.
- A foreign data wrapper can be dropped only during the internal initialization phase or within an upgrade script. In other scenarios, users cannot directly use this syntax.

Syntax

```
DROP FOREIGN DATA WRAPPER [ IF EXISTS ] name [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether the specified FDW exists.

  - If the FDW exists, it will be dropped successfully.
  - If the FDW does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the FDW does not exist.

- **name**

  Specifies the name of an existing FDW.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the FDW when dropping the FDW itself. If this parameter is not specified, the default value is **RESTRICT**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - **CASCADE**: The database cascadingly drops objects (such as servers) that depend on the FDW.
  - **RESTRICT**: The database refuses to drop the FDW if any objects that depend on it.

Examples

```
-- Create a FDW dbi.
gaussdb=# CREATE FOREIGN DATA WRAPPER dbi OPTIONS (bar 'true');

-- Drop the FDW dbi. (Prerequisite: The GUC parameter support_extended_features is set to on.)
gaussdb=# DROP FOREIGN DATA WRAPPER dbi;
```

Helpful Links

CREATE FOREIGN DATA WRAPPER and ALTER FOREIGN DATA WRAPPER

1.13.10.14 DROP FOREIGN TABLE

Description

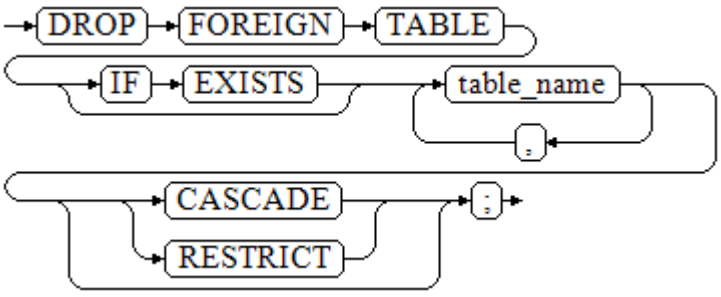
Drops a foreign table from the database.

Precautions

This operation forcibly drops the specified table, resulting in the removal of any dependent indexes. Functions and stored procedures that use this table will become unexecutable.

Syntax

```
DROP FOREIGN TABLE [ IF EXISTS ] table_name [, ...] [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified foreign table exists.
  - If the foreign table exists, it will be dropped successfully.
  - If the foreign table does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the foreign table does not exist.
- **table_name**

  Specifies the name of the table to be dropped.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: names of existing tables.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the foreign table when dropping the table itself. If this parameter is not specified, the default value is **RESTRICT**.

  - **CASCADE**: The database cascadingly drops the objects (such as views) that depend on the table.
  - **RESTRICT**: The database refuses to drop the table if any objects depend on it.

Examples

For details, see **CREATE FOREIGN TABLE** in **Examples**.

Helpful Links

CREATE FOREIGN TABLE

1.13.10.15 DROP FUNCTION

Description

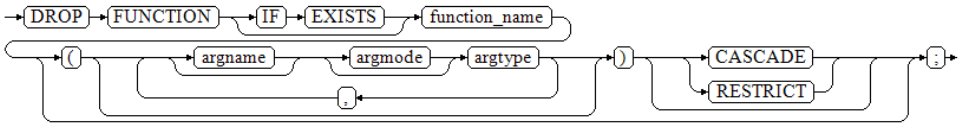
Drops an existing function from the database.

Precautions

- If a function involves operations on temporary tables, DROP FUNCTION cannot be used.
- Only the function owner and users with the DROP permission on the function can perform this operation. System administrators have this permission by default.
- The functions of the initial user can only be dropped by the initial user itself.

Syntax

```
DROP FUNCTION [ IF EXISTS ] function_name
[ ( [ { [ argname ] [ argmode ] argtype} [, ...] ] ) [ CASCADE | RESTRICT ] ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified function exists.

  - If the function exists, it will be dropped successfully.
  - If the function does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the function does not exist.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **function_name**

  Specifies the name of the function to be dropped.

  Value range: existing function names.

- **argmode**

  Specifies the parameter mode of the function.

- **argname**

  Specifies the parameter name of the function.

- **argtype**

  Specifies the data type of a function parameter.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the function when dropping the function itself. If this parameter is not specified, the default value is **RESTRICT**.

  – **CASCADE**: The database cascadingly drops objects that depend on the function.

  – **RESTRICT**: The database refuses to drop the function if any objects depend on it.

Examples

- Delete the list of parameters that can be omitted when a function.

```
-- Create a function.
gaussdb=# CREATE FUNCTION func_test(varchar) RETURN VARCHAR AS
BEGIN
    RETURN $1||_'test';
END;
/

-- Delete the function.
gaussdb=# DROP FUNCTION func_test;
```

- Delete a function with the same name.

  If a function with the same name exists, add a parameter list when deleting the function. Otherwise, an error is reported.

```
-- Create a function.
gaussdb=# CREATE FUNCTION func_add(int) RETURNS int AS $$
BEGIN
    RETURN $1+10;
END;
$$ LANGUAGE PLPGSQL;

-- func_add is an overloaded function.
gaussdb=# CREATE FUNCTION func_add(int,int) RETURNS int AS $$
BEGIN
    RETURN $1+$2;
END;
$$ LANGUAGE PLPGSQL;

-- Delete the function.
gaussdb=# DROP FUNCTION func_add(int);
gaussdb=# DROP FUNCTION func_add(int,int);
```

Helpful Links

ALTER FUNCTION and **CREATE FUNCTION**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.10.16 DROP GLOBAL CONFIGURATION

Description

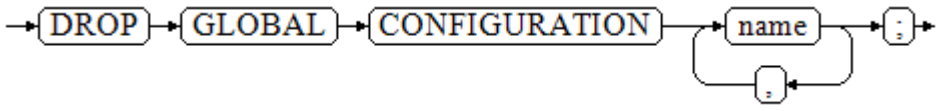
Drops parameter values from the gs_global_config system catalog.

Precautions

- Only the initial database user can run this command.
- The parameter name cannot be **weak_password** or **undostoragetype**.
- If the name of parameter for redistribution after scale-out is used, set **enable_cluster_resize** to **on**. The involved parameters are **lockwait_timeout**, **lockwait_interval**, **trylock_threshold**, **enable_cancel**, **last_catchup_threshold**, **catchup_times**, **write_error_mode**, **catchup_query_dop**, **parallel_catchup_threshold**, **parallel_reindex_jobs**, **redis_retry_times**, **redis_retry_interval**, **redis_bucket_expansion**, **redis_bucket_batchsize**, **redis_bucket_log_level**, **redis_bucket_lockhold_timeout**, **redis_bucket_timeout_penalty**, **redis_bucket_max_lockwait_timeout_retry**, **redis_bucket_max_lockhold_timeout_retry**, **redis_bucket_strategy_history_capacity**, **redis_bucket_strategy_recent_threshold**, **redis_bucket_workload_rule**, **redis_bucket_lsn_rule**, **redis_bucket_tx_bucket_min**, **redis_bucket_tx_bucket_max**, **redis_max_expected_tps**, **redis_bucket_sender_concurrency**, **redis_bucket_receiver_concurrency**, **redis_bucket_parse_redo_num**, **redis_bucket_workers_per_paser**, **redis_bucket_keep_bucketxlogs**, **redis_bucket_reindex_cbi_offline**, and **redis_bucket_reindex_parallel_workers**.

Syntax

```
DROP GLOBAL CONFIGURATION name [, ...];
```



Parameters

name

The parameter must exist in the gs_global_config system catalog. If you drop a parameter that does not exist, an error will be reported. If the name of parameter for redistribution after scale-out is used, set **enable_cluster_resize** to **on**.

Examples

For details, see **ALTER GLOBAL CONFIGURATION** in **Examples**.

Helpful Links

ALTER GLOBAL CONFIGURATION

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.10.17 DROP GROUP

Description

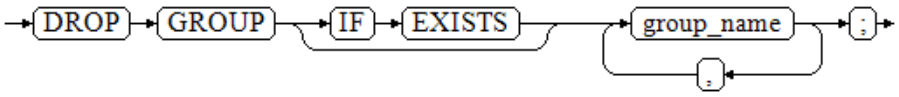
Drops a user group. DROP GROUP is synonymous with DROP ROLE.

Precautions

It is available only to users with the CREATE ROLE permission granted by the administrator.

Syntax

```
DROP GROUP [ IF EXISTS ] group_name [, ...];
```



Parameters

- **IF EXISTS**

  Checks whether a specified user group exists.

  - If the user group exists, it will be dropped successfully.
  - If the user group does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the user group does not exist.

- **group_name**

  Specifies the name of the user group to be dropped.

  Value range: existing user groups and roles.

Examples

For details, see **CREATE GROUP** in **Examples**.

Helpful Links

CREATE GROUP, **ALTER GROUP**, and **DROP ROLE**

1.13.10.18 DROP INDEX

Description

Drops an index.

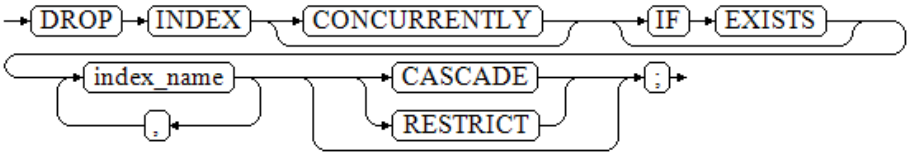
Precautions

Only the index owner, owner of the schema where the index resides, users with the INDEX permission on the table where the index resides, or users with the DROP ANY INDEX permission can perform this operation. When separation of duties is disabled, system administrators have this permission by default.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax

```
DROP INDEX [ CONCURRENTLY ] [ IF EXISTS ]
    index_name [, ...] [ CASCADE | RESTRICT ];
```



Parameters

- **CONCURRENTLY**

  Deletes an index (with a level-4 lock) online in a mode that does not block DML statements. When an index is deleted, other statements cannot access the table on which the index depends. With this option, the statement does not lock the table during index deletion.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When the **CONCURRENTLY** option is specified for dropping an index, only one index name can be specified and the **CASCADE** option is not supported.
- The **DROP INDEX** statement can be run within a transaction, but **DROP INDEX CONCURRENTLY** cannot.
- For temporary tables, you can use **CONCURRENTLY** to delete indexes. However, indexes are deleted in blocking mode because no other sessions concurrently access the temporary tables and the blocking mode is more cost-effective.
- Similar to online index creation, the online index deletion process also includes a phase where the system must wait for the end of other related transactions in the table corresponding to the index. As a result, the lock timeout may occur, causing online index deletion failures.
- Similar to online index creation, the online index deletion process also includes a phase where the system must wait for the end of other related transactions in the table corresponding to the index. As a result, deadlocks with other DDL statements may occur, causing online index deletion failures. The cause is as follows: The online index deletion process holds the level-4 lock and waits for the concurrent DDL transactions to end. The concurrent DDL transactions hold the transaction lock and wait for the online index deletion process to release the level-4 lock.
- When a fault such as lock timeout, deadlock, or database exception occurs or a user cancels an operation, online index dropping may fail, resulting in invalid residual indexes. The status of remaining indexes on CNs and DNs may be inconsistent. For example, indexes on CNs are invalid and those on DNs are valid, or indexes on some CNs are valid and those on some CNs are invalid and those on DNs are valid. Alternatively, remaining indexes exist on some nodes but do not exist on other nodes, but this does not cause inconsistency. To prevent occupying system resources, you need to clear the invalid indexes as soon as possible. You can continue to use **DROP INDEX CONCURRENTLY** or **DROP INDEX** to delete invalid indexes. If **DROP INDEX CONCURRENTLY** fails to be used to delete indexes continuously, you can use **DROP INDEX** instead.
- The online index deletion time is related to the concurrent service volume and concurrent service running time. If there is no concurrent service, the online index deletion is complete within seconds. If there are concurrent services, you need to wait until all concurrent services are complete at a certain time. Therefore, the online index deletion time needs to include the running time of these concurrent services.
- Only B-tree and UB-tree indexes can be dropped online in a distributed system. GSIs are not supported.
- During the execution of DROP INDEX CONCURRENTLY, an error may be reported for the stream plan involving the index. Before performing this operation, you are advised to disable the stream plan using an SQL patch or by adding hints to the relevant SQL statements, forcing the execution to fall back to the pgxc plan.
- If an online index drop operation is performed during the generation of a WDR snapshot, the error message "Could not open relation with OID %u." may be displayed, causing the WDR snapshot generation to fail. This is an expected behavior. After the index drop is complete, generate the WDR snapshot again.

- **IF EXISTS**

  Checks whether a specified index exists.

  – If the index exists, it will be dropped successfully.

  – If the index does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the index does not exist.

  If an index fails to be created or dropped online, its metadata may not be matched on some nodes. The index cannot be dropped using the DROP INDEX statement. Instead, it must be dropped using DROP INDEX IF EXISTS.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **index_name**

  Specifies the name of the index to be dropped.

  Value range: existing indexes.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the index when dropping the index itself. If this parameter is not specified, the default value is **RESTRICT**.

  – **CASCADE**: The database cascadingly drops objects that depend on the index.

  – **RESTRICT**: The database refuses to drop the index if any objects depend on it.

Examples

```
-- Create a table.
gaussdb=# CREATE TABLE test1_index (id INT, name VARCHAR(20));

-- Create an index.
gaussdb=# CREATE INDEX idx_test1 ON test1_index(id);

-- Drop an index.
gaussdb=# DROP INDEX IF EXISTS idx_test1 CASCADE;

-- Create an index online.
gaussdb=# CREATE INDEX CONCURRENTLY idx_test2 ON test1_index(name);

-- Drop an index online.
gaussdb=# DROP INDEX CONCURRENTLY idx_test2;

-- Drop a table.
gaussdb=# DROP TABLE test1_index;
```

Helpful Links

ALTER INDEX and **CREATE INDEX**

1.13.10.19 DROP LANGUAGE

Description

Drops a procedural language.

Precautions

This syntax is not supported currently.

1.13.10.20 DROP MASKING POLICY

Description

Drops a masking policy.

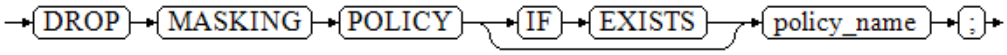
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

Only users with the POLADMIN or SYSADMIN permission, or the initial user can perform this operation.

Syntax

```
DROP MASKING POLICY [ IF EXISTS ] policy_name;
```



Parameters

policy_name

Specifies the masking policy name, which must be unique.

Value range: names of existing policies (character strings).

Example

```
-- Create table tb_for_masking.
gaussdb=# CREATE TABLE tb_for_masking(idx int, col1 text, col2 text, col3 text, col4 text, col5 text, col6 text, col7 text,col8 text);

-- Insert data into the tb_for_masking table.
gaussdb=# INSERT INTO tb_for_masking VALUES(1, '9876543210', 'usr321usr', 'abc@huawei.com', 'abc@huawei.com', '1234-4567-7890-0123', 'abcdef 123456 ui 323 jsfd321 j3k2l3', '4880-9898-4545-2525', 'this is a llt case');

-- Create a resource label for sensitive column col1.
gaussdb=# CREATE RESOURCE LABEL mask_lb1 ADD COLUMN(tb_for_masking.col1);

-- Create a resource label for sensitive column col2.
gaussdb=# CREATE RESOURCE LABEL mask_lb2 ADD COLUMN(tb_for_masking.col2);

-- Create a resource label for sensitive column col3.
gaussdb=# CREATE RESOURCE LABEL mask_lb3 ADD COLUMN(tb_for_masking.col3);

-- Create a data masking policy named maskpol1.
gaussdb=# CREATE MASKING POLICY maskpol1 maskall ON LABEL(mask_lb1);

-- Create a data masking policy named maskpol2.
gaussdb=# CREATE MASKING POLICY maskpol2 randommasking ON LABEL(mask_lb2);

-- Create a data masking policy named maskpol3.
gaussdb=# CREATE MASKING POLICY maskpol3 basicemailmasking ON LABEL(mask_lb3);

-- Drop a masking policy.
gaussdb=# DROP MASKING POLICY IF EXISTS maskpol1;

-- Drop a group of masking policies.
gaussdb=# DROP MASKING POLICY IF EXISTS maskpol2, maskpol3;

-- Drop resource labels.
gaussdb=# DROP RESOURCE LABEL mask_lb1;
gaussdb=# DROP RESOURCE LABEL mask_lb2;
gaussdb=# DROP RESOURCE LABEL mask_lb3;

-- Drop a table.
gaussdb=# DROP TABLE tb_for_masking;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

ALTER MASKING POLICY and **CREATE MASKING POLICY**

1.13.10.21 DROP MATERIALIZED VIEW

Description

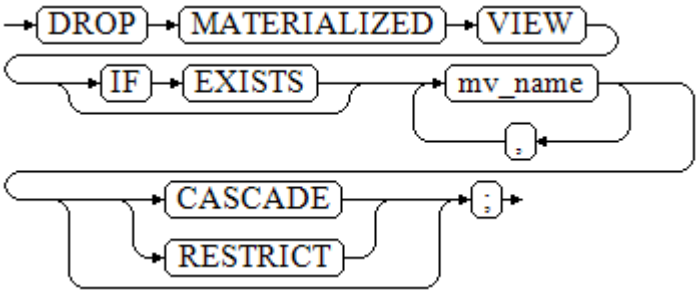
Drops an existing materialized view from the database.

Precautions

The owner of a materialized view, owner of the schema where the materialized view resides, users with the DROP MATERIALIZED VIEW permission, or users with the DROP ANY TABLE permission can perform this operation. System administrators have this permission by default.

Syntax

```
DROP MATERIALIZED VIEW [ IF EXISTS ] mv_name [, ...] [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified materialized view exists.

  - If the materialized view exists, it will be dropped successfully.
  - If the materialized view does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the materialized view does not exist.

- **mv_name**

  Specifies the name of the materialized view to be dropped.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the materialized view when dropping the materialized view itself. If this parameter is not specified, the default value is **RESTRICT**.

  - **CASCADE**: The database cascadingly drops the objects that depend on the materialized view.
  - **RESTRICT**: The database refuses to drop the materialized view if any objects depend on it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

```
-- Create a table.
gaussdb=# CREATE TABLE my_table (c1 int, c2 int)
WITH(STORAGE_TYPE=ASTORE);

-- Create a materialized view named my_mv.
gaussdb=# CREATE MATERIALIZED VIEW my_mv AS SELECT * FROM my_table;

-- Drop the materialized view named my_mv.
gaussdb=# DROP MATERIALIZED VIEW my_mv;

-- Drop a table.
gaussdb=# DROP TABLE my_table;
```

Helpful Links

ALTER MATERIALIZED VIEW, **CREATE INCREMENTAL MATERIALIZED VIEW**, **CREATE MATERIALIZED VIEW**, **CREATE TABLE**, **REFRESH INCREMENTAL MATERIALIZED VIEW**, and **REFRESH MATERIALIZED VIEW**

1.13.10.22 DROP MODEL

Description

Drops a model object that has been trained and saved.

Precautions

This syntax is not supported currently.

1.13.10.23 DROP NODE

Description

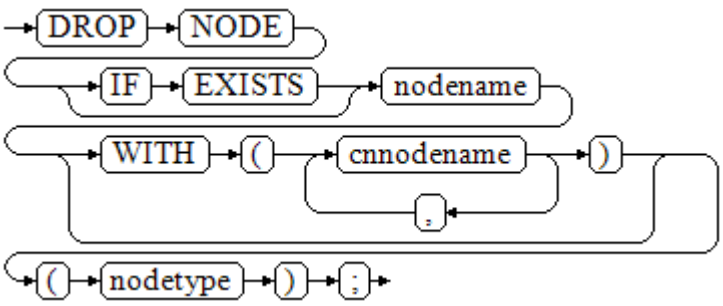
Drops nodes from a cluster.

Precautions

DROP NODE is an API of the cluster management tool and is used to manage clusters. Only administrators have the permission to use this API. Do not use this API directly. Otherwise, the cluster status may be affected and the database may be unavailable.

Syntax

```
DROP NODE [ IF EXISTS ] nodename [WITH ( cnnodename [, ... ] )] (nodetype);
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **IF EXISTS**

  Reports a notice instead of an error if the specified node does not exist.

- **nodename**

  Specifies the name of the node to be dropped.

  Value range: an existing node name.

- **cnnodename**

  Specifies the CN name. If this parameter is defined, the cluster nodes under the currently connected CN and the defined CN are dropped.

  Value range: an existing CN name.

- **nodetype**

  Type of the node to be changed, which is the value of **node_type** in the pgxc_node system catalog.

  Value range: **"C"** (CN)

NOTE

If there are multiple CNs, cluster nodes are created only on the currently connected CNs and will not be synchronized to other CNs. You need to create cluster nodes on all CNs. To drop a cluster node, you only need to connect to one CN. If cluster nodes are created on only one CN, you need to define the current CN name and drop the cluster node on the CN.

Examples

```
-- Create cluster nodes on all CNs.
gaussdb=# CREATE NODE datanode1 WITH(
   TYPE = datanode,
   PREFERRED = false
);

-- Connect to only one CN to drop cluster nodes.
gaussdb=# DROP NODE datanode1;
```

Helpful Links

CREATE NODE and **ALTER NODE**

1.13.10.24 DROP NODE GROUP

Description

Drops a node group.

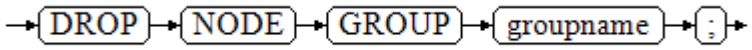
Precautions

- DROP NODE GROUP is an API of the cluster management tool.
- Only system administrators or users with the DROP permission on node groups can perform this operation.

Syntax

```
DROP NODE GROUP groupname;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

groupname

Specifies the name of the node group to be dropped.

Value range: names of existing node groups.

Examples

```
-- Query the initial cluster DN status.
gaussdb=# SELECT node_name, nodeis_preferred FROM pgxc_node WHERE node_type = 'D' ORDER BY 1;
 node_name | nodeis_preferred
-----------+------------------
 dn_6001_6002_6003 | f
 dn_6004_6005_6006 | f
 dn_6007_6008_6009 | f
(3 rows)

-- Create a node group and replace dn_6001_6002_6003 with the actual node name obtained in the previous step.
gaussdb=# CREATE NODE GROUP test_group WITH ( dn_6001_6002_6003 );

-- Drop the node group.
gaussdb=# DROP NODE GROUP test_group;
```

Helpful Links

ALTER NODE GROUP and **CREATE NODE GROUP**

1.13.10.25 DROP OWNED

Description

Deletes the database objects owned by a database role.

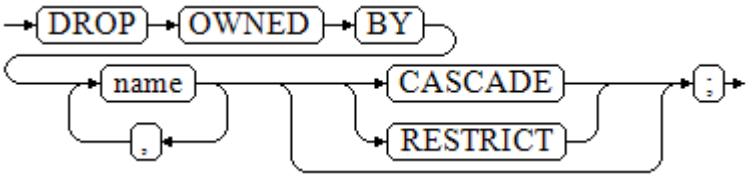
Precautions

- The role's permissions on all objects in the current database and shared objects (databases and tablespaces) will be revoked.
- DROP OWNED is usually considered as a preparation for removing one or more roles and affects only the objects in the current database. You need to run this statement in each database that contains the objects owned by the role to be removed.
- Using the **CASCADE** option may recursively delete objects owned by other users. Exercise caution when using this option.
- The databases and tablespaces owned by the role will not be removed.
- Private database links owned by a role can be dropped only after the **CASCADE** option is added.

Syntax

```
DROP OWNED BY name [, ...] [ CASCADE | RESTRICT ];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- **name**

  Specifies the role name.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to delete objects that depend on the role in cascading mode. The default value is **RESTRICT**.

  – **CASCADE**: cascadingly drops the objects that depend on the objects to be dropped.

  – **RESTRICT**: refuses to delete the role that has dependent objects.

Examples

```
-- Create user jim.
gaussdb=# CREATE USER jim PASSWORD '********';

-- Revoke user jim's permissions on all objects in the current database and on shared objects (databases and tablespaces).
gaussdb=# DROP OWNED BY jim;

-- Delete user jim.
gaussdb=# DROP USER jim;
```

Helpful Links

REASSIGN OWNED

1.13.10.26 DROP PACKAGE

Description

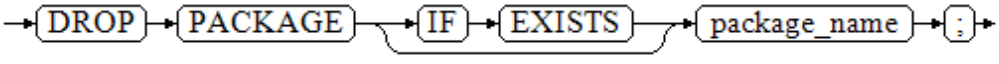
Drops an existing package or package body.

Precautions

- After a package body is dropped, the stored procedures and functions in the package become invalid.
- Only the initial user has the permission to perform the DROP operation on the packages of the initial user.

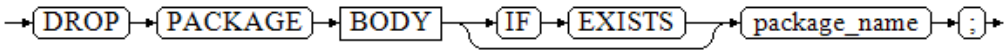
Syntax

```
DROP PACKAGE [ IF EXISTS ] package_name;
```




```
DROP PACKAGE BODY [ IF EXISTS ] package_name;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- **IF EXISTS**

  Checks whether a specified package or package body exists.

  - If the package or package body exists, it will be dropped successfully.
  - If the package or package body does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the package or package body does not exist.

- **package_name**

  Specifies the name of the package to be dropped.

  Value range: names of existing packages.

Examples

```
-- Create an ORA-compatible database.
gaussdb=# CREATE DATABASE ora_compat_db DBCOMPATIBILITY 'ORA';
CREATE DATABASE
gaussdb=# \c ora_compat_db

-- Create a package.
ora_compat_db=# CREATE OR REPLACE PACKAGE emp_bonus IS
var1 INT:=1;-- Public variable
var2 INT:=2;
PROCEDURE testpro1(var3 INT);-- Public stored procedure, which can be called by external systems.
END emp_bonus;
/

-- Drop a package.
ora_compat_db=# DROP PACKAGE emp_bonus;

-- Switch back to the initial database and drop the ORA-compatible database. Replace postgres with the actual database name.
ora_compat_db=# \c postgres
gaussdb=# DROP DATABASE ora_compat_db;
```

Helpful Links

ALTER PACKAGE and **CREATE PACKAGE**

1.13.10.27 DROP PROCEDURE

Description

Drops a stored procedure.

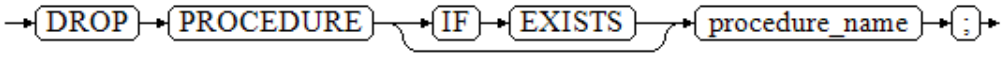
Precautions

Only the initial user has the permission to perform the DROP operation on the stored procedures of the initial user.

Syntax

```
DROP PROCEDURE [ IF EXISTS ] procedure_name;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- **IF EXISTS**

  Checks whether a specified stored procedure exists.
  - If the stored procedure exists, it will be dropped successfully.
  - If the stored procedure does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the stored procedure does not exist.
- **procedure_name**

  Specifies the name of the stored procedure to be dropped.

  Value range: names of existing stored procedures.

Examples

For details, see **CREATE PROCEDURE** in **Examples**.

Helpful Links

ALTER PROCEDURE and **CREATE PROCEDURE**

1.13.10.28 DROP RESOURCE LABEL

Description

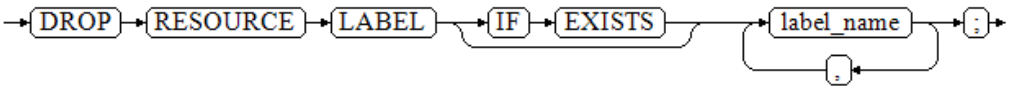
Drops a resource label.

Precautions

Only users with the POLADMIN or SYSADMIN permission, or the initial user, can perform this operation.

Syntax

```
DROP RESOURCE LABEL [IF EXISTS] label_name [, ...];
```



Parameters

label_name

Specifies the name of the resource label to be dropped.

Value range: character strings complying with **Identifier Naming Conventions**.

Examples

```
-- Create table tb_for_label.
gaussdb=# CREATE TABLE tb_for_label(col1 text, col2 text, col3 text);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a resource label based on a table.
gaussdb=# CREATE RESOURCE LABEL IF NOT EXISTS table_label add TABLE(public.tb_for_label);

-- Create an existing table resource label again and compare the differences between adding IF NOT EXISTS and not adding IF NOT EXISTS.
gaussdb=# CREATE RESOURCE LABEL IF NOT EXISTS table_label add TABLE(public.tb_for_label);
NOTICE:  table_label label already defined, skipping
CREATE RESOURCE LABEL
gaussdb=# CREATE RESOURCE LABEL table_label add TABLE(public.tb_for_label);
ERROR:  table_label label already defined

-- Create a resource label based on columns.
gaussdb=# CREATE RESOURCE LABEL IF NOT EXISTS column_label add COLUMN(public.tb_for_label.col1);

-- Create schema schema_for_label.
gaussdb=# CREATE SCHEMA schema_for_label;

-- Create a resource label based on a schema.
gaussdb=# CREATE RESOURCE LABEL IF NOT EXISTS schema_label add SCHEMA(schema_for_label);

-- Create view view_for_label.
gaussdb=# CREATE VIEW view_for_label AS SELECT 1;

-- Create a resource label based on a view.
gaussdb=# CREATE RESOURCE LABEL IF NOT EXISTS view_label add VIEW(view_for_label);

-- Create function func_for_label.
gaussdb=# CREATE FUNCTION func_for_label RETURNS TEXT AS $$ SELECT col1 FROM tb_for_label; $$ LANGUAGE SQL;

-- Create a resource label based on a function.
gaussdb=# CREATE RESOURCE LABEL IF NOT EXISTS func_label add FUNCTION(func_for_label);

-- Drop the table resource label table_label.
gaussdb=# DROP RESOURCE LABEL IF EXISTS table_label;

-- Drop the column resource label column_label.
gaussdb=# DROP RESOURCE LABEL IF EXISTS column_label;

-- Drop the function resource label func_for_label.
gaussdb=# DROP FUNCTION func_for_label;

-- Drop the view resource label view_for_label.
gaussdb=# DROP VIEW view_for_label;

-- Drop the schema resource label schema_for_label.
gaussdb=# DROP SCHEMA schema_for_label;

-- Drop the tb_for_label table.
gaussdb=# DROP TABLE tb_for_label;
```

Helpful Links

ALTER RESOURCE LABEL and **CREATE RESOURCE LABEL**

1.13.10.29 DROP RESOURCE POOL

Description

Drops a resource pool.

NOTE

The resource pool cannot be dropped if it is associated with a role.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

- Only users with the SYSADMIN permission or the initial user can perform this operation.
- This syntax is not supported during upgrade.

Syntax

```
DROP RESOURCE POOL [ IF EXISTS ] pool_name;
```



Parameters

- **IF EXISTS**

  Checks whether the specified resource pool exists.

  - If the specified resource pool exists, the deletion is successful.
  - If the specified resource pool does not exist, the deletion is not performed. Only a NOTICE message is generated, indicating that the specified resource pool does not exist.

- **pool_name**

  Specifies the name of a created resource pool.

  Value range: character strings complying with **Identifier Naming Conventions**.

NOTE

In a multi-tenant scenario, dropping a group resource pool also drops the related service resource pools. A resource pool can be dropped only when it is not associated with any users.

Examples

For details, see **CREATE RESOURCE POOL** in **Examples**.

Helpful Links

ALTER RESOURCE POOL and **CREATE RESOURCE POOL**

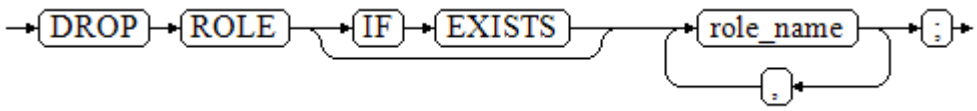
1.13.10.30 DROP ROLE

Description

Drops a role.

Syntax

```
DROP ROLE [ IF EXISTS ] role_name [, ...];
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **IF EXISTS**

  Checks whether a specified role exists.

  - If the role exists, it will be dropped successfully.
  - If the role does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the role does not exist.

- **role_name**

  Specifies the name of the role to be dropped.

  Value range: names of existing roles.

Examples

```
-- Create role role11.
gaussdb=# CREATE ROLE role11  PASSWORD '********';
CREATE ROLE

-- Drop role role11.
gaussdb=# DROP ROLE IF EXISTS role11;
DROP ROLE

-- Drop role role12 that does not exist.
gaussdb=# DROP ROLE IF EXISTS role12;
NOTICE:  role "role12" does not exist, skipping
DROP ROLE
```

Helpful Links

CREATE ROLE, **ALTER ROLE**, and **SET ROLE**

1.13.10.31 DROP ROW LEVEL SECURITY POLICY

Description

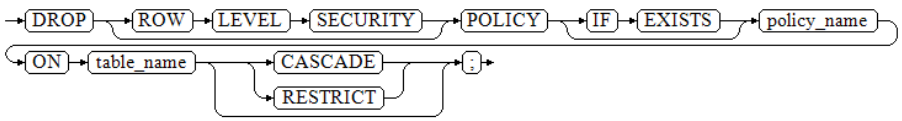
Drops a row-level security policy from a table.

Precautions

Only the table owner or administrators can drop a row-level security policy from the table.

Syntax

```
DROP [ ROW LEVEL SECURITY ] POLICY [ IF EXISTS ] policy_name ON table_name [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified row-level security policy exists.

  - If the row-level security policy exists, it will be dropped successfully.
  - If the row-level security policy does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the row-level security policy does not exist.
- **policy_name**

  Specifies the name of the row-level security policy to be dropped.
- **table_name**

  Specifies the name of the table containing the row-level security policy.
- **CASCADE | RESTRICT**

  Currently, no objects depend on row-level security policies. Therefore, CASCADE is equivalent to RESTRICT, and they are reserved to ensure backward compatibility.

Examples

```
-- Create the data table all_data.
gaussdb=# CREATE TABLE all_data(id int, role varchar(100), data varchar(100));

-- Create a row-level security policy.
gaussdb=# CREATE ROW LEVEL SECURITY POLICY all_data_rls ON all_data USING(role = CURRENT_USER);

-- Drop the row-level security policy.
gaussdb=# DROP ROW LEVEL SECURITY POLICY all_data_rls ON all_data;

-- Drop the all_data table.
gaussdb=# DROP TABLE all_data;
```

Helpful Links

ALTER ROW LEVEL SECURITY POLICY and **CREATE ROW LEVEL SECURITY POLICY**

1.13.10.32 DROP SCHEMA

Description

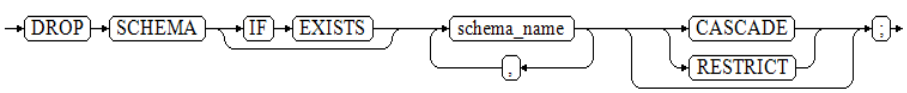
Drops a specified schema from the current database.

Precautions

- Only the schema owner or users with the DROP permission on the schema can execute the **DROP SCHEMA** command. When separation of duties is disabled, system administrators have this permission by default.
- You are not allowed to drop DBE_PLDEVELOPER when **allow_system_table_mods** is disabled.
- Only the initial user can perform the DROP operation on system schemas, except public schemas.

Syntax

```
DROP SCHEMA [ IF EXISTS ] schema_name [, ...] [ CASCADE | RESTRICT ];
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **IF EXISTS**

  Checks whether the specified schema exists. This parameter can be used to avoid errors.

  - If the specified schema exists, you can perform DROP directly.
  - If the specified schema does not exist, a notice will be issued instead of an error.

- **schema_name**

  Specifies a schema name.

  Value range: existing schema names.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the schema when dropping the schema itself. If not specified, the default is **RESTRICT**.

  - **CASCADE**: automatically drops all the objects contained in the schema.
  - **RESTRICT**: If the schema contains any objects, the schema will fail to be dropped.

NOTICE

- Do not drop schemas starting with pg_temp or pg_toast_temp. Such schemas are for internal use only, and dropping them may result in unexpected results.
- Do not directly drop the current schema. You can switch to another schema and then drop the current schema.

Examples

```
-- Create role role2.
gaussdb=# CREATE ROLE role2 IDENTIFIED BY '********';

-- Create a schema based on the name.
gaussdb=# CREATE SCHEMA schema1 AUTHORIZATION role2;
CREATE SCHEMA

-- Drop the schema.
gaussdb=# DROP SCHEMA schema1 CASCADE;

-- Drop a user.
gaussdb=# DROP USER role2 CASCADE;
```

Helpful Links

ALTER SCHEMA and **CREATE SCHEMA**

1.13.10.33 DROP SECURITY LABEL

Description

Drops a security label from the current database.

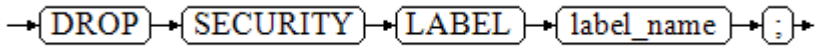
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

The initial user, a user with the SYSADMIN permission, or a user who inherits permissions of the built-in role gs_role_seclabel can drop security labels.

Syntax

```
DROP SECURITY LABEL label_name;
```



Parameters

label_name

Specifies the name of the security label to be dropped. The security label must exist in the database.

Examples

```
-- Create a security label sec_label.
gaussdb=# CREATE SECURITY LABEL sec_label 'L1:G4';

-- Drop the security label sec_label2 that does not exist.
gaussdb=# DROP SECURITY LABEL sec_label2;
ERROR:  security label "sec_label2" does not exist

-- Drop the existing security label sec_label.
gaussdb=# DROP SECURITY LABEL sec_label;
```

Helpful Links

CREATE SECURITY LABEL and **SECURITY LABEL ON**

1.13.10.34 DROP SEQUENCE

Description

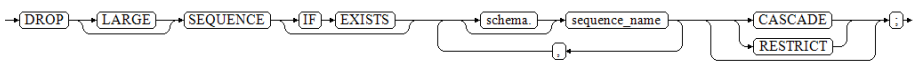
Drops a sequence from the current database.

Precautions

Only the owner of a sequence, the owner of the schema to which the sequence belongs, or users with the DROP permission on a sequence or users with the DROP ANY SEQUENCE permission can perform this operation. When separation of duties is disabled, system administrators have this permission by default.

Syntax

```
DROP [LARGE] SEQUENCE [ IF EXISTS ] { [schema.] sequence_name } [ , ... ] [ CASCADE | RESTRICT ];
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **LARGE**

  Specifies a LARGE flag. If a LARGE flag is used when the sequence is created, the LARGE flag must be specified when the sequence is dropped.

- **IF EXISTS**

  Checks whether a specified sequence exists.

  - If the sequence exists, it will be dropped successfully.
  - If the sequence does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the sequence does not exist.

- **sequence_name**

  Specifies a sequence name.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the sequence when dropping the sequence itself. If this parameter is not specified, the default value is **RESTRICT**.

  - **CASCADE**: The database cascadingly drops the objects that depend on the sequence.
  - **RESTRICT**: The database refuses to drop the sequence if any objects depend on it.

Examples

```
-- Create an ascending sequence named serial, starting from 101.
gaussdb=# CREATE SEQUENCE serial START 101;

-- Drop the sequence.
gaussdb=# DROP SEQUENCE serial;
```

Helpful Links

ALTER SEQUENCE and **DROP SEQUENCE**

1.13.10.35 DROP SERVER

Description

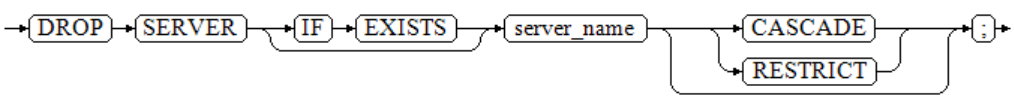
Drops an existing data server.

Precautions

Only the server owner or users with the DROP permission can perform this operation. System administrators have this permission by default.

Syntax

```
DROP SERVER [ IF EXISTS ] server_name [ CASCADE | RESTRICT ];
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **IF EXISTS**

  Checks whether a specified data server exists.

  - If the data server exists, it will be dropped successfully.
  - If the data server does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the data server does not exist.

- **server_name**

  Specifies the name of the data server to be dropped.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the data server when dropping the server itself. If this parameter is not specified, the default value is **RESTRICT**.

  - **CASCADE**: The database cascadingly drops the objects that depend on the data server.
  - **RESTRICT**: The database refuses to drop the data server if any objects depend on it.

Examples

```
-- Create a server.
gaussdb=# CREATE SERVER my_server FOREIGN DATA WRAPPER log_fdw;
CREATE SERVER

-- Drop a server.
gaussdb=# DROP SERVER my_server;
DROP SERVER
```

Helpful Links

ALTER SERVER and **CREATE SERVER**

1.13.10.36 DROP SYNONYM

Description

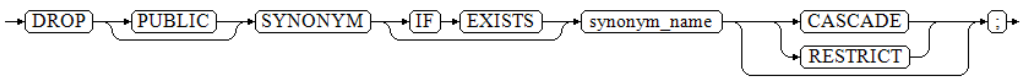
Drops a synonym.

Precautions

Only the synonym owner or users with the DROP ANY SYNONYM permission can perform this operation. System administrators have this permission by default.

Syntax

```
DROP [PUBLIC] SYNONYM [ IF EXISTS ] synonym_name [ CASCADE | RESTRICT ];
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **PUBLIC**

  (Optional) Drops PUBLIC synonyms.

- **IF EXISTS**

  Checks whether a specified synonym exists.

  - If the synonym exists, it will be dropped successfully.
  - If the synonym does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the synonym does not exist.

- **synonym_name**

  Specifies the synonym name (using a schema optionally).

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the synonym when dropping the synonym itself. If this parameter is not specified, the default value is **RESTRICT**.

  - **CASCADE**: The database cascadingly drops the objects (such as views) that depend on the synonym.
  - **RESTRICT**: The database refuses to drop the synonym if any objects depend on it.

Examples

```
-- Create a synonym.
gaussdb=# CREATE OR REPLACE SYNONYM test_tbl1 FOR test_tbl1;
-- Drop a synonym.
gaussdb=# DROP SYNONYM test_tbl1;
```

Helpful Links

ALTER SYNONYM and **CREATE SYNONYM**

1.13.10.37 DROP TABLE

Description

Drops a table.

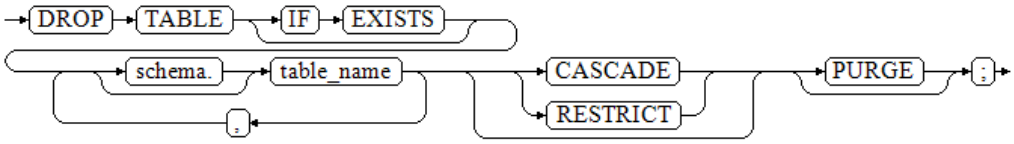
Precautions

- After the table is dropped, the indexes depending on the table are dropped, and the functions and stored procedures that need to use this table cannot be executed. Deleting a partitioned table also deletes all partitions in the table.
- Only the owner of a table, the owner of the schema of the table, users with the DROP permission on the table, or users with the DROP ANY TABLE permission can perform this operation. When separation of duties is disabled, system administrators have the permission by default.

Syntax

```
DROP TABLE [ IF EXISTS ]
   { [schema.]table_name } [, ...] [ CASCADE | RESTRICT ] [ PURGE ];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- **IF EXISTS**

  Checks whether a specified table exists.

  - If the table exists, it will be dropped successfully.
  - If the table does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the table does not exist.

- **schema**

  Specifies a schema name. The parameter is optional. It is used to specify restrictions for a table.

- **table_name**

  Table name.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the table when dropping the table itself. If this parameter is not specified, the default value is **RESTRICT**.

  - **CASCADE**: The database cascadingly drops the objects that depend on the table, such as views, triggers, and indexes. Note that joined table objects cannot be dropped cascadingly.
  - **RESTRICT**: The database refuses to drop the table if any objects depend on it.

- **PURGE**

  Specifies that even if the recycle bin function is enabled, the table is physically dropped instead of being moved to the recycle bin when you use DROP TABLE to drop tables.

Examples

```
-- Create the test table.
gaussdb=# CREATE TABLE test(c1 int, c2 int);
-- Drop the test table.
gaussdb=# DROP TABLE IF EXISTS test;

-- Create the test1 table.
gaussdb=# CREATE TABLE test1(c1 int, c2 int);
-- Create the v_test1 view.
gaussdb=# CREATE VIEW v_test1 AS SELECT * FROM test1 WHERE c1 < 20;
-- An error is reported when the table is dropped.
gaussdb=# DROP TABLE test1;
ERROR:  table test1 cannot be dropped because other objects depend on it.
DETAIL:  view v_test1 depends on table test1
HINT:  Use DROP ... CASCADE to drop the dependent objects.

-- Use the CASCADE parameter to drop the test1 table, which will cascadingly drop the views.
gaussdb=# DROP TABLE test1 CASCADE;
NOTICE:  drop cascades to view v_test1
DROP TABLE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

ALTER TABLE and **CREATE TABLE**

1.13.10.38 DROP TABLESPACE

Description

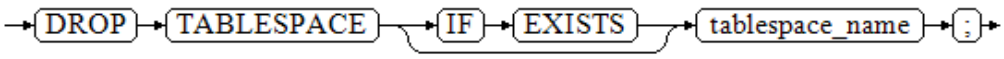
Drops a tablespace.

Precautions

- Only the tablespace owner or users with the DROP TABLESPACE permission can perform this operation. System administrators have this permission by default.
- The tablespace to be dropped should not contain any database objects. Otherwise, an error will be reported.
- After a tablespace is dropped, rollback is not supported. Therefore, the tablespace cannot be dropped within a transaction block.
- When you drop a tablespace, if another session is running **\db** for a query, the query may fail due to the tablespace transaction. In this case, run **\db** again.
- If DROP TABLESPACE fails to be executed, execute DROP TABLESPACE IF EXISTS.

Syntax

```
DROP TABLESPACE [ IF EXISTS ] tablespace_name;
```



Parameters

- **IF EXISTS**

  Checks whether a specified tablespace exists.

  - If the tablespace exists, it will be dropped successfully.
  - If the tablespace does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the tablespace does not exist.

- **tablespace_name**

  Specifies the tablespace name.

  Value range: names of existing tablespaces.

Examples

```
-- Create a tablespace.
gaussdb=# CREATE TABLESPACE tbs_location1 RELATIVE LOCATION 'test_tablespace/test_tablespace_1';

-- Drop the tablespace.
gaussdb=# DROP TABLESPACE tbs_location1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

ALTER TABLESPACE and **CREATE TABLESPACE**

1.13.10.39 DROP TEXT SEARCH CONFIGURATION

NOTICE

This function is for internal use only. You are advised not to use it.

Description

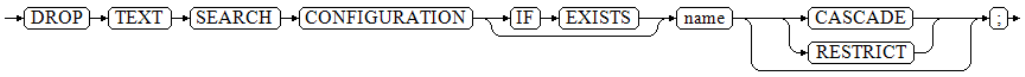
Drops an existing text search configuration.

Precautions

Only the owner of the text search configuration can perform this operation.

Syntax

```
DROP TEXT SEARCH CONFIGURATION [ IF EXISTS ] name [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified text search configuration exists.
  - If the text search configuration exists, it will be dropped successfully.
  - If the text search configuration does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the text search configuration does not exist.
- **name**

  Specifies the name of the text search configuration to be dropped. You can use a pattern to specify the name.
- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the text search configuration when dropping the text search configuration itself. If this parameter is not specified, the default value is **RESTRICT**.
  - **CASCADE**: The database cascadingly drops the objects that depend on the text search configuration.
  - **RESTRICT**: The database refuses to drop the text search configuration if any objects depend on it.

Examples

For details, see **CREATE TEXT SEARCH CONFIGURATION** in **Examples**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.10.40 DROP TEXT SEARCH DICTIONARY

NOTICE

This function is for internal use only. You are advised not to use it.

Description

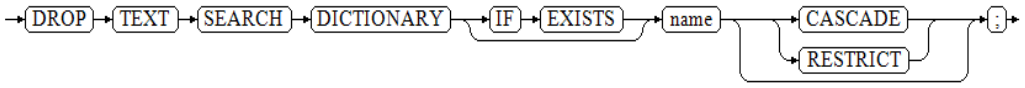
Drops an existing full-text search dictionary.

Precautions

- Predefined dictionary templates do not support the DROP operation.
- Only the owner of a dictionary or a system administrator has the permission to perform this operation.
- Exercise caution when specifying **CASCADE** explicitly with DROP. This operation will cascadingly drop the text search configuration that uses the dictionary.

Syntax

```
DROP TEXT SEARCH DICTIONARY [ IF EXISTS ] name [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified full-text search dictionary exists.
  - If the full-text search dictionary exists, it will be dropped successfully.
  - If the full-text search dictionary does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the full-text search dictionary does not exist.
- **name**

  Specifies the name (optionally schema-qualified) of the full-text search dictionary to be dropped. (If you do not specify a schema name, the dictionary will be dropped in the current schema by default.)

  Value range: names of existing dictionaries.
- **CASCADE** | **RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the dictionary when dropping the dictionary itself. If this parameter is not specified, the default value is **RESTRICT**.
  - **CASCADE**: The database cascadingly drops the objects that depend on the full-text search dictionary and other objects that depend on these objects.
  - **RESTRICT**: The database refuses to drop the dictionary if any objects depend on it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

```
-- Create the dictionary english.
gaussdb=# CREATE TEXT SEARCH DICTIONARY english (
    TEMPLATE = simple
);

-- Drop the english dictionary.
gaussdb=# DROP TEXT SEARCH DICTIONARY english;
```

Helpful Links

CREATE TEXT SEARCH DICTIONARY

1.13.10.41 DROP TRIGGER

Description

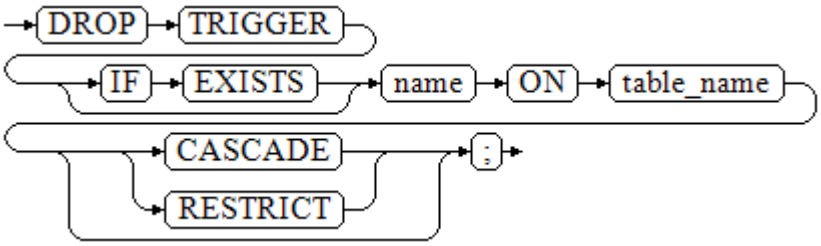
Drops a trigger.

Precautions

Only the trigger owner or users with the DROP ANY TRIGGER permission can perform this operation. System administrators have this permission by default.

Syntax

```
DROP TRIGGER [ IF EXISTS ] name ON table_name [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified trigger exists.

  - If the trigger exists, it will be dropped successfully.
  - If the trigger does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the trigger does not exist.

- **trigger_name**

  Specifies the name of the trigger to be deleted.

  Value range: an existing trigger name

- **table_name**

  Specifies the name of the table containing the trigger.

  Value range: name of the table containing the trigger

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the trigger when dropping the trigger itself. If this parameter is not specified, the default value is **RESTRICT**.

  – **CASCADE**: The database cascadingly drops the objects that depend on the trigger.

  – **RESTRICT**: The database refuses to drop the trigger if any objects depend on it.

Examples

```
-- Create a source table and a target table.
gaussdb=# CREATE TABLE test_trigger_src_tbl(id1 INT, id2 INT, id3 INT);
gaussdb=# CREATE TABLE test_trigger_des_tbl(id1 INT, id2 INT, id3 INT);
```

Use the INSERT trigger.

```
-- Create an INSERT trigger function.
gaussdb=# CREATE OR REPLACE FUNCTION tri_insert_func() RETURNS TRIGGER AS
        $$
        DECLARE
        BEGIN
                INSERT INTO test_trigger_des_tbl VALUES(NEW.id1, NEW.id2, NEW.id3);
                RETURN NEW;
        END
        $$ LANGUAGE plpgsql;

-- Create an INSERT trigger.
gaussdb=# CREATE TRIGGER insert_trigger
        BEFORE INSERT ON test_trigger_src_tbl
        FOR EACH ROW
        EXECUTE PROCEDURE tri_insert_func();

-- Execute the INSERT event and check the trigger results.
gaussdb=# INSERT INTO test_trigger_src_tbl VALUES(100,200,300);

gaussdb=# SELECT * FROM test_trigger_src_tbl;
 id1 | id2 | id3
-----+-----+-----
 100 | 200 | 300
(1 row)

gaussdb=# SELECT * FROM test_trigger_des_tbl;  // Check whether the trigger operation takes effect.
 id1 | id2 | id3
-----+-----+-----
 100 | 200 | 300
(1 row)
```

Use the UPDATE trigger.

```
-- Create an UPDATE trigger function.
gaussdb=# CREATE OR REPLACE FUNCTION tri_update_func() RETURNS TRIGGER AS
$$
DECLARE
BEGIN
UPDATE test_trigger_des_tbl SET id3 = NEW.id3 WHERE id1=OLD.id1;
RETURN OLD;
END
$$ LANGUAGE plpgsql;

-- Create an UPDATE trigger.
gaussdb=# CREATE TRIGGER update_trigger
        AFTER UPDATE ON test_trigger_src_tbl
        FOR EACH ROW
        EXECUTE PROCEDURE tri_update_func();
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Execute the UPDATE event and check the trigger results.
gaussdb=# UPDATE test_trigger_src_tbl SET id3=400 WHERE id1=100;

gaussdb=# SELECT * FROM test_trigger_src_tbl;
 id1 | id2 | id3
-----+-----+-----
 100 | 200 | 400
(1 row)

gaussdb=# SELECT * FROM test_trigger_des_tbl;  // Check whether the trigger operation takes effect.
 id1 | id2 | id3
-----+-----+-----
 100 | 200 | 400
(1 row)
```

Use the DELETE trigger.

```
-- Create a DELETE trigger function.
gaussdb=# CREATE OR REPLACE FUNCTION TRI_DELETE_FUNC() RETURNS TRIGGER AS
        $$
        DECLARE
        BEGIN
                DELETE FROM test_trigger_des_tbl WHERE id1=OLD.id1;
                RETURN OLD;
        END
        $$ LANGUAGE plpgsql;

-- Create a DELETE trigger.
gaussdb=# CREATE TRIGGER delete_trigger BEFORE DELETE ON test_trigger_src_tbl FOR EACH ROW
EXECUTE PROCEDURE tri_delete_func();

-- Execute the DELETE event and check the trigger results.
gaussdb=# DELETE FROM test_trigger_src_tbl WHERE id1=100;

gaussdb=# SELECT * FROM test_trigger_src_tbl;
 id1 | id2 | id3
-----+-----+-----
(0 rows)

gaussdb=# SELECT * FROM test_trigger_des_tbl;  // Check whether the trigger operation takes effect.
 id1 | id2 | id3
-----+-----+-----
(0 rows)
```

Rename a trigger.

```
-- Rename a trigger.
gaussdb=# ALTER TRIGGER delete_trigger ON test_trigger_src_tbl RENAME TO delete_trigger_renamed;
```

Disable a trigger.

```
-- Disable insert_trigger.
gaussdb=# ALTER TABLE test_trigger_src_tbl DISABLE TRIGGER insert_trigger;

gaussdb=# INSERT INTO test_trigger_src_tbl VALUES(100,200,300);

gaussdb=# SELECT * FROM test_trigger_src_tbl;
 id1 | id2 | id3
-----+-----+-----
 100 | 200 | 300
(1 row)

gaussdb=# SELECT * FROM test_trigger_des_tbl; // The trigger does not take effect.
 id1 | id2 | id3
-----+-----+-----
(0 rows)

-- Disable all triggers on the current table.
gaussdb=# ALTER TABLE test_trigger_src_tbl DISABLE TRIGGER ALL;
```

Deleting a trigger

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# DROP TRIGGER insert_trigger ON test_trigger_src_tbl;

gaussdb=# DROP TRIGGER update_trigger ON test_trigger_src_tbl;

gaussdb=# DROP TRIGGER delete_trigger_renamed ON test_trigger_src_tbl;
```

Deleting a function

```
gaussdb=# DROP FUNCTION tri_insert_func;

gaussdb=# DROP FUNCTION tri_update_func;

gaussdb=# DROP FUNCTION tri_delete_func;
-- Drop the source table and destination table.
gaussdb=# DROP TABLE test_trigger_src_tbl;
gaussdb=# DROP TABLE test_trigger_des_tbl;
```

Helpful Links

CREATE TRIGGER, **ALTER TRIGGER**, and **ALTER TABLE**

1.13.10.42 DROP TYPE

Description

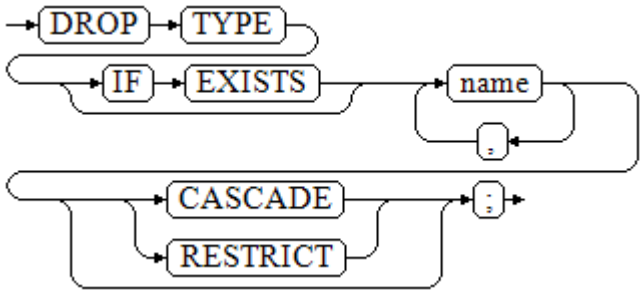
Drops a user-defined data type.

Precautions

Only the owner of a type, users with the DROP TYPE permission, or users with the DROP ANY TYPE permission can perform this operation. When separation of duties is disabled, system administrators have this permission by default.

Syntax

```
DROP TYPE [ IF EXISTS ] name [, ...] [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified type exists.

  - If the type exists, it will be dropped successfully.
  - If the type does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the type does not exist.

- **name**

Specifies the name of the type to be dropped. You can use a schema to specify the name.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the type when dropping the type itself. If this parameter is not specified, the default value is **RESTRICT**.

  - **CASCADE**: The database cascadingly drops the objects (such as columns, functions, and operators) that depend on the type.
  - **RESTRICT**: The database refuses to drop the type if any objects depend on it.

Examples

For details, see **CREATE TYPE** in **Examples**.

Helpful Links

CREATE TYPE and **ALTER TYPE**

1.13.10.43 DROP USER

Description

Drops a specified user.

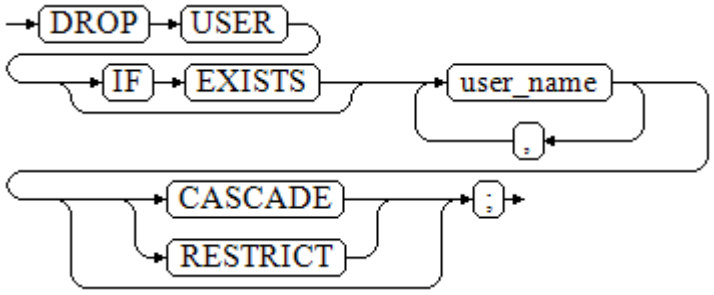
Precautions

- The DROP USER operation can be performed only when you have the permission to create a user. After a user is dropped, the schema with the same name is also dropped.
- Specifying CASCADE explicitly cannot drop locked objects that depend on a user when dropping the user itself unless the objects are unlocked or the threads locking the objects are terminated.
- If the dependent objects are other databases or reside in other databases, manually drop them before dropping the user from the current database. That is, DROP USER cannot drop objects across databases.
- Before dropping a user across databases, you must first drop the schema—which has the same name as the user—created by that user in the database where the user resides.
- Before dropping a user, you need to drop all the objects owned by the user and revoke the user's permissions on other objects. Alternatively, specify **CASCADE** to drop these objects and permissions.
- In multi-tenant scenarios, service users will also be dropped when you drop group users. If you specify CASCADE to drop group users, you also need to specify CASCADE explicitly when dropping service users.
- If the user has an error table that is specified during the creation of a GDS foreign table, the user cannot be dropped by specifying the CASCADE keyword in the DROP USER statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax

```
DROP USER [ IF EXISTS ] user_name [, ...] [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified user exists.

  - If the user exists, the user is dropped successfully.
  - If the user does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the user does not exist.

- **user_name**

  Specifies the name of the user to be dropped.

  Value range: existing usernames in the database.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop objects that depend on the user when dropping the user itself. If this parameter is not specified, the default value is **RESTRICT**.

  - **CASCADE**: automatically drops the objects that depend on the user and revokes permissions from the user.
  - **RESTRICT**: The database refuses to drop the user if any objects depend on it or if the user has been granted permissions on other objects.

  NOTE

  In GaussDB, configuring the **enable_kill_query** parameter in the **gaussdb.conf** file affects the cascading dropping of user objects.

  - If **enable_kill_query** is **on** and **CASCADE** is used, the statement automatically ends the threads locking dependent objects and directly drops the specified user.
  - When **enable_kill_query** is **off** and **CASCADE** is used, the statement waits until the threads locking the dependent objects end and then drops the specified user.

Examples

```
-- Create user jim whose login password is ********.
gaussdb=# CREATE USER jim PASSWORD '********';

-- Drop user tom that does not exist.
gaussdb=# DROP USER IF EXISTS tom;
NOTICE:  role "tom" does not exist, skipping
DROP ROLE

-- Drop the user.
gaussdb=# DROP USER jim CASCADE;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

ALTER USER and **CREATE USER**

1.13.10.44 DROP VIEW

Description

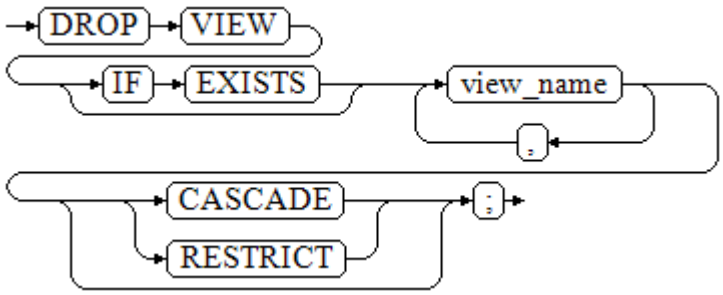
Drops a view from the database.

Precautions

Only the owner of a view, the owner of the schema of the view, users with the DROP permission on the view, or users with the DROP ANY TABLE permission can run this command. When separation of duties is disabled, system administrators have the permission by default.

Syntax

```
DROP VIEW [ IF EXISTS ] view_name [, ...] [ CASCADE | RESTRICT ];
```



Parameters

- **IF EXISTS**

  Checks whether a specified view exists.

  - If the view exists, it will be dropped successfully.
  - If the view does not exist, the DROP operation will not be performed. Only a NOTICE message will be generated, indicating that the view does not exist.

- **view_name**

  Specifies the name of the view to be dropped.

  Value range: existing view names.

- **CASCADE | RESTRICT**

  (Optional) Specifies whether to drop dependent objects along with the view. If not specified, **RESTRICT** is the default.

  - **CASCADE**: automatically drops the objects (such as other views) that depend on the view.
  - **RESTRICT**: refuses to drop the view if any objects depend on it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

```
-- Create the test_tb1 table and insert 100 data records into the table.
gaussdb=# CREATE TABLE test_tb1(col1 int, col2 int);
gaussdb=# INSERT INTO test_tb1 VALUES (generate_series(1,100),generate_series(1,100));

-- Create a view whose value of col1 is less than 3.
gaussdb=# CREATE VIEW test_v1 AS SELECT * FROM test_tb1 WHERE col1 < 3;

-- Query the view.
gaussdb=# SELECT * FROM test_v1;
 col1 | col2
------+------
    1 |    1
    2 |    2
(2 rows)

-- Drop the table and the view.
gaussdb=# DROP VIEW test_v1;
gaussdb=# DROP TABLE test_tb1;
```

Helpful Links

ALTER VIEW and **CREATE VIEW**

1.13.10.45 DROP WEAK PASSWORD DICTIONARY

Description

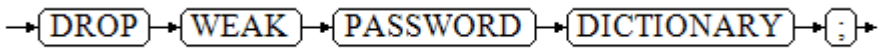
Clears the weak password dictionary.

Precautions

- Only the initial user, system administrators, and security administrators can perform this operation.
- Users can use this statement to clear all weak passwords in the gs_global_config system catalog.

Syntax

```
DROP WEAK PASSWORD DICTIONARY;
```



Examples

```
-- Insert a single weak password into the gs_global_config system catalog.
gaussdb=# CREATE WEAK PASSWORD DICTIONARY WITH VALUES ('********');

-- Check weak passwords in the gs_global_config system catalog.
gaussdb=# SELECT * FROM gs_global_config WHERE NAME LIKE 'weak_password';
      name       |    value
-----------------+--------------
 weak_password   | ********
(1 rows)

-- Clear all weak passwords in the gs_global_config system catalog.
gaussdb=# DROP WEAK PASSWORD DICTIONARY;

-- View existing weak passwords.
gaussdb=# SELECT * FROM gs_global_config WHERE NAME LIKE 'weak_password';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 name | value
------+-------
(0 rows)
```

Helpful Links

CREATE WEAK PASSWORD DICTIONARY

1.13.11 E

1.13.11.1 EXECUTE

Description

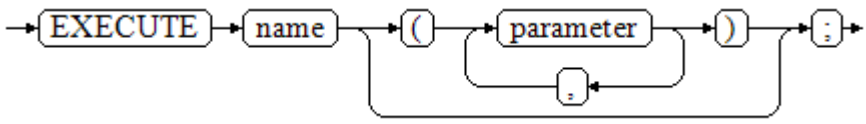
Executes a prepared statement. Because a prepared statement exists only in the lifetime of the session, the prepared statement must be created earlier in the current session by using the PREPARE statement.

Precautions

- If the PREPARE statement declares some parameters when the prepared statement is created, the parameter set passed to the EXECUTE statement must be compatible. Otherwise, an error occurs. If no parameter is defined in the prepared statement declaration, any valid parameter binding passed in subsequent EXECUTE statements will be silently ignored by the system.
- If the table definition on which the prepared statement depends changes (for example, a column definition is added, deleted, or modified), the expected result definition in the cached plan may not match the actual table definition. As a result, an unexpected result is returned when the statement is executed.

Syntax

```
EXECUTE name [ ( parameter [, ...] ) ];
```



Parameters

- **name**

  Specifies the name of the prepared statement to be executed.

- **parameter**

  Specifies a parameter of the prepared statement. It must be an expression that generates a value compatible with the data type of the parameter specified when the prepared statement was created. **ROWNUM** cannot be used as a parameter.

Examples

```
-- Create the reason table.
gaussdb=# CREATE TABLE reason (
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    CD_DEMO_SK          int NOT NULL,
    CD_GENDER           varchar(10),
    CD_MARITAL_STATUS   varchar(10)
);

-- Create a prepared statement for an INSERT statement and execute the prepared statement.
gaussdb=# PREPARE insert_reason(int,varchar(10),varchar(10)) AS INSERT INTO reason VALUES($1,$2,$3);
gaussdb=# EXECUTE insert_reason(52, 'AAAAAAAADD', 'reason 52');

-- Query data.
gaussdb=# SELECT * FROM reason;
 cd_demo_sk | cd_gender  | cd_marital_status
------------+------------+--------------------
         52 | AAAAAAAADD | reason 52
(1 row)

-- Drop the reason table.
gaussdb=# DROP TABLE reason;
```

Helpful Links

PREPARE and **DEALLOCATE**

1.13.11.2 EXECUTE DIRECT

Description

Executes an SQL statement on a specified node. Generally, the cluster automatically allocates an SQL statement to proper nodes. EXECUTE DIRECT is mainly used for database maintenance and testing.

Precautions

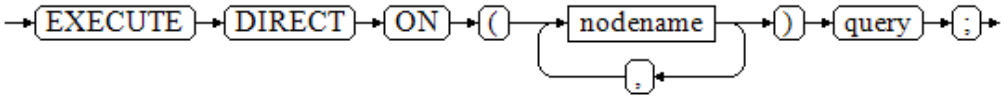
- When **enable_nonsysadmin_execute_direct** is **off**, only the SYSADMIN or MONADMIN has the EXECUTE DIRECT permission.
- When **enable_udf_execute_direct** is **off**, EXECUTE DIRECT cannot be used to execute user-defined functions, functions in the advanced packages DBE_SQL, PKG_SERVICE, and DBE_XMLGE, as well as DBE_UTILITY.EXEC_DDL_STATEMENT.
- To ensure data consistency across nodes, only the SELECT statement can be used. Transaction statements, DDL, and DML cannot be used.
- When the stddev aggregation calculation is performed on the specified DN using such statements, the result set is returned in triplet. For example, {3, 8, 30} indicates that the count result is 3, the sum result is 8, and the sum of squares is 30. When the AVG aggregation calculation is performed on the specified DN using such statements, the result set is returned in a binary tuple, for example, {4,2}. The result of count is 4, and that of sum is 2.
- When multiple nodes are specified, aggregate functions are not supported. If the query contains an aggregate function, the message "EXECUTE DIRECT on multinode not support agg functions." is returned.
- CN nodes do not store user table data. Therefore, do not execute SELECT for querying user tables on a CN.
- If the SQL statement to be executed is also EXECUTE DIRECT, do not nest it into EXECUTE DIRECT; instead, directly execute the inner EXECUTE DIRECT.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

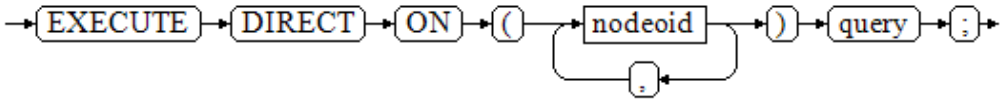
- The query result of the aggregate function is inconsistent with that on the CN. Multiple pieces of information are returned. The array_avg function is not supported.
- If the **nodeoid** or **nodeoid list** parameter is specified, this parameter takes effect only when the **enable_direct_standby_datanodes** parameter is successfully set in distributed mode. To query data on a specified DN, invalid or duplicate **nodeoid** values are not supported, and **nodeoid** and **nodename** cannot be used together.

Syntax

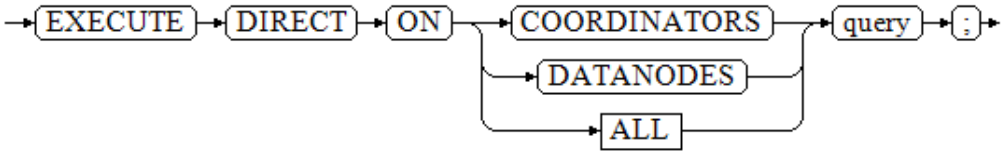
```
EXECUTE DIRECT ON ( nodename [, ... ] ) query ;
```




```
EXECUTE DIRECT ON ( nodeoid [, ... ] ) query;
```




```
EXECUTE DIRECT ON { COORDINATORS | DATANODES | ALL } query;
```



Parameters

- **nodename**

  Node name.

  Value range: an existing node name
- **nodeoid**

  Node OID.

  Value range: an existing DN OID, which can be obtained from the **PGXC_NODE** system catalog.
- **query**

  Specifies the query statement to be executed.
- **COORDINATORS**

  Runs the query statement on all CNs.
- **DATANODES**

  Runs the query statement on all DNs.
- **ALL**

  Runs the query statement on all CNs and DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

```
-- Query the name of the current node in the cluster.
gaussdb=# SELECT node_name,oid FROM pgxc_node WHERE node_type IN ('C','D') ORDER BY 1;
     node_name     |  oid
-------------------+-------
 cn_5001           | 15517
 cn_5002           | 16387
 cn_5003           | 16391
 dn_6001_6002_6003 | 16384
 dn_6004_6005_6006 | 16388
 dn_6007_6008_6009 | 16392
(6 rows)


-- Create a range-distributed table and insert data into the table.
gaussdb=# CREATE TABLE tbl_range(
  id       INT,
  name     VARCHAR(20),
  province VARCHAR(60),                       -- Province
  country   VARCHAR(30) DEFAULT 'China'       -- Country
)DISTRIBUTE BY RANGE(id)(
  SLICE s1 VALUES LESS THAN (100) DATANODE dn_6001_6002_6003,
  SLICE s2 VALUES LESS THAN (200) DATANODE dn_6004_6005_6006,
  SLICE s3 VALUES LESS THAN (MAXVALUE) DATANODE dn_6007_6008_6009
);
gaussdb=# INSERT INTO tbl_range VALUES (52,'Bob','Beijing','China'),
                                       (100,'Ben','Shanghai','China'),
                                       (150,'Scott','Guangzhou','China'),
                                       (300,'Jordan','Beijing','China');

-- Query the table data distributed on the dn_6007_6008_6009 node.
gaussdb=# EXECUTE DIRECT ON(dn_6007_6008_6009) 'SELECT * FROM tbl_range';
 id  |  name  | province | country
-----+--------+----------+---------
 300 | Jordan | Beijing  | China
(1 row)

-- Delete the tbl_range table.
gaussdb=# DROP TABLE tbl_range;
```

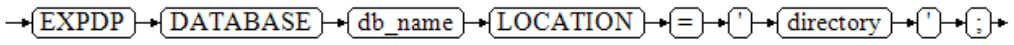
1.13.11.3 EXPDP DATABASE

Description

Exports all physical files in a database.

Syntax

```
EXPDP DATABASE db_name LOCATION = 'directory';
```



Parameters

- **db_name**

  Name of the database to be exported.

- **directory**

  Directory for storing exported files.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

```
-- The EXPDP DATABASE syntax is used for fine-grained backup and restoration and is called by the backup and restoration tool. If a user calls it directly, the error message "ERROR: Only auxdb is allowed to execute expdp cmd." is displayed.
gaussdb=# EXPDP DATABASE test LOCATION = '/data1/expdp/database';
```

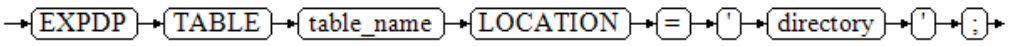
1.13.11.4 EXPDP TABLE

Description

Exports all table-related index, sequence, partition, TOAST, and TOAST index files.

Syntax

```
EXPDP TABLE table_name LOCATION = 'directory';
```



Parameters

- **table_name**

  Name of the table to be exported.
- **directory**

  Directory for storing exported files.

Examples

```
-- The EXPDP TABLE syntax is used for fine-grained backup and restoration and is called by the backup and restoration tool. If a user calls it directly, the error message "ERROR: Only auxdb is allowed to execute expdp cmd." is displayed.
gaussdb=# EXPDP TABLE test_t LOCATION = '/data1/expdp/table0';
```

1.13.11.5 EXPLAIN

Description

Shows the execution plan of an SQL statement, including:

- How the tables referenced by the SQL statement will be scanned, for example, by plain sequential scan or index scan.
- How the tables referenced by the SQL statement will be joined.

Precautions

- The most critical part of the display is the estimated statement execution cost, which is the plan generator's guess at how long it will take to run the statement.
- If the **ANALYZE** option is specified, the SQL statement will be executed and show actual run times and other statistics. This is useful for determining whether the plan generator's estimates are close to reality. The statistics include the total elapsed time expended within each plan node (in milliseconds) and the total number of rows it actually returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The statement is actually executed when the **ANALYZE** option is used. If you want to use EXPLAIN ANALYZE on an INSERT, UPDATE, DELETE, CREATE TABLE AS, or EXECUTE statement without letting the statement affect your data, use this approach:

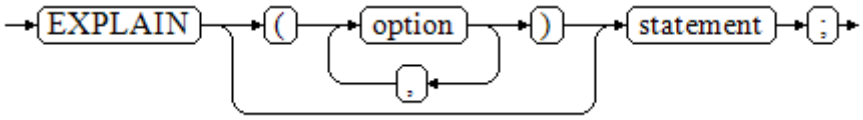
```
START TRANSACTION;
EXPLAIN ANALYZE ...;
ROLLBACK;
```

- When you use EXPLAIN to view an execution plan, the audit type recorded in the audit log depends on the specific SQL statement being analyzed.

Syntax

- Display the execution plan of an SQL statement, which supports multiple options and has no requirements for the order of options.

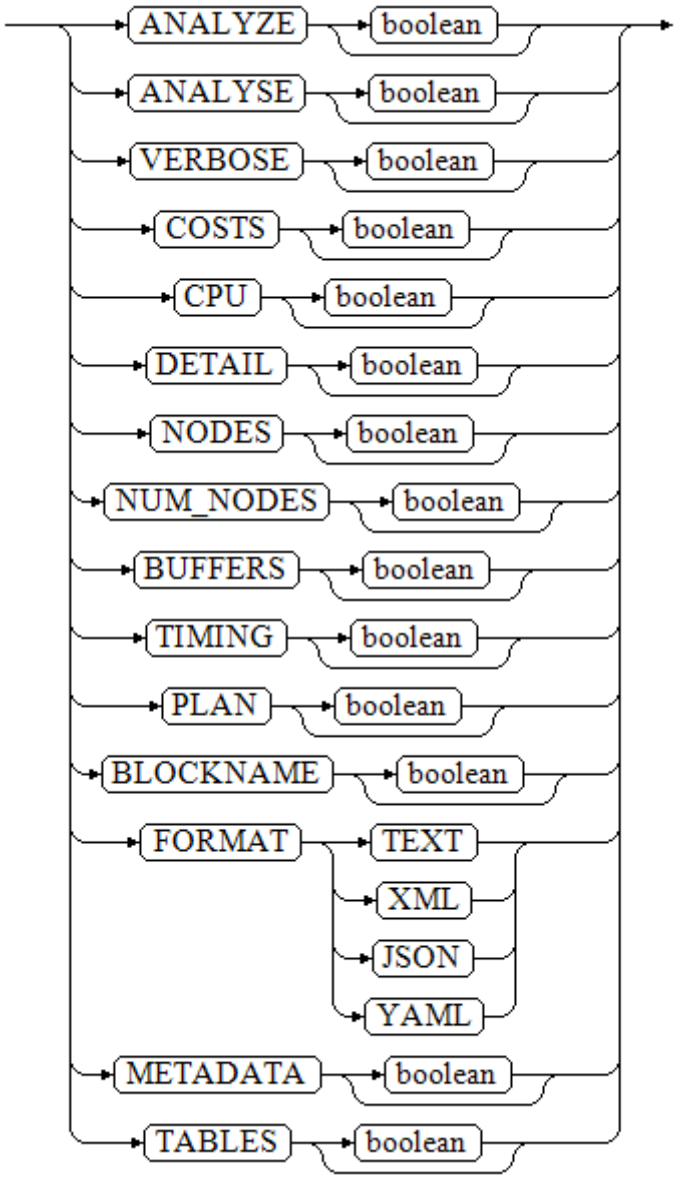
```
EXPLAIN [ ( option [, ...] ) ] statement;
```



option can be as follows:

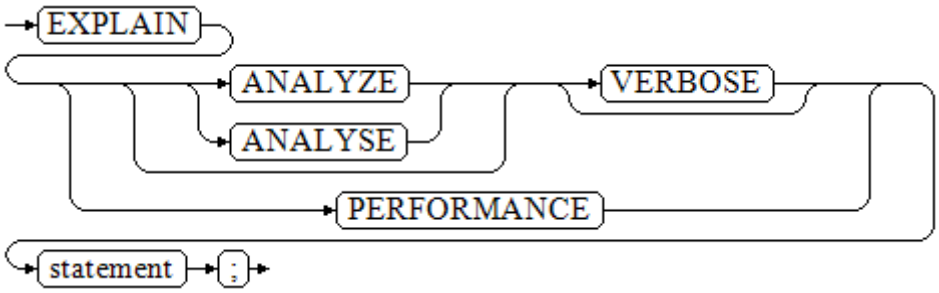
```
{
    ANALYZE [ boolean ] |
    ANALYSE [ boolean ] |
    VERBOSE [ boolean ] |
    COSTS [ boolean ] |
    CPU [ boolean ] |
    DETAIL [ boolean ] |
    NODES [ boolean ] |
    NUM_NODES [ boolean ] |
    BUFFERS [ boolean ] |
    TIMING [ boolean ] |
    PLAN [ boolean ] |
    BLOCKNAME [ boolean ] |
    FORMAT { TEXT | XML | JSON | YAML } |
    METADATA [ boolean ] |
    TABLES [ boolean ]
}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



- Display the execution plan of an SQL statement, where options are in order.

```
EXPLAIN { [ ANALYZE | ANALYSE ] [ VERBOSE ] | PERFORMANCE } statement;
```



Parameters

- **statement**

  Specifies the SQL statement to explain.

- **ANALYZE boolean | ANALYSE boolean**

  Specifies whether to display actual run times and other statistics. When two parameters are used at the same time, the latter one in **option** takes effect.

  Value range:

  – **TRUE** (default): displayed.

  – **FALSE**: not displayed.

- **VERBOSE boolean**

  Specifies whether to display additional information regarding the plan.

  Value range:

  – **TRUE** (default): displayed.

  – **FALSE**: not displayed.

- **COSTS boolean**

  Specifies whether to display the estimated total cost of each plan node, estimated number of rows, estimated width of each row.

  Value range:

  – **TRUE** (default): displayed.

  – **FALSE**: not displayed.

- **CPU boolean**

  Specifies whether to display CPU usage. This parameter must be used together with ANALYZE or ANALYSE.

  Value range:

  – **TRUE** (default): displayed.

  – **FALSE**: not displayed.

- **DETAIL boolean**

  Specifies whether to display DN information. This parameter must be used together with ANALYZE or ANALYSE.

  Value range:

  – **TRUE** (default): displays it.

  – **FALSE**: not printed.

- **NODES boolean**

  Specifies whether to display information about the nodes executed by query.

  Value range:

  – **TRUE** (default): displays it.

  – **FALSE**: not printed.

- **NUM_NODES boolean**

  Specifies whether to display the number of executing nodes.

  Value range:

  – **TRUE** (default): displays it.

  – **FALSE**: not printed.

- **BUFFERS boolean**

  Specifies whether to display buffer usage. This parameter must be used together with ANALYZE or ANALYSE.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **TRUE**: displayed.
- **FALSE** (default): not displayed.

- **TIMING boolean**

  Specifies whether to display the actual startup time and time spent on the output node. This parameter must be used together with ANALYZE or ANALYSE.

  Value range:

  - **TRUE** (default): displayed.
  - **FALSE**: not displayed.

- **PLAN boolean**

  Specifies whether to store the execution plan in **PLAN_TABLE**. If this parameter is set to **on**, the execution plan is stored in **PLAN_TABLE** and is not displayed on the screen. Therefore, this parameter cannot be used together with other parameters when it is set to **on**.

  Value range:

  - **TRUE** (default): The execution plan is stored in **PLAN_TABLE** and not displayed on the screen. If the plan is stored successfully, "EXPLAIN SUCCESS" is returned.
  - **FALSE**: The execution plan is not stored but is printed on the screen.

- **BLOCKNAME boolean**

  Specifies whether to display the query block where each operation of the plan is located. When this option is enabled, each name of the query block where each operation is performed is displayed in the **Query Block** column. This helps users obtain the query block name and use hints to modify the execution plan.

  - **TRUE** (default value): The name of the query block where each operation is located is displayed in the **Query Block** column. This option must be used in the pretty mode. For details, see "Hint-based Optimization > Hints for Specifying the Query Block Where the Hint Is Located" in *SQL Optimization Guide*.
  - **FALSE**: The plan display is not affected.

- **FORMAT**

  Specifies the output format.

  Value range: **TEXT**, **XML**, **JSON**, and **YAML**.

  Default value: **TEXT**.

- **PERFORMANCE**

  Prints all relevant information in execution. Some information is described as follows:

  - **ex c/r**: indicates the average number of CPU cycles used by each row, which is equal to **(ex cyc) / (ex row)**.
  - **ex row**: indicates the number of executed rows.
  - **ex cyc**: indicates the number of used CPU cycles.
  - **inc cyc**: indicates the total number of CPU cycles used by subnodes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - **shared hit**: indicates the shared buffer hits of the operator.
  - **loops**: indicates the number of operator loop execution times.
  - **total_calls**: indicates the total number of generated elements.
  - **remote query poll time stream gather**: indicates the operator used to listen to the network poll time when data on each DN reaches the CN.
  - **deserialize time**: indicates the time required for deserialization.
  - **estimated time**: indicates the estimated time.
  - **Network Poll Time**: indicates the duration for the libcomm receiver to wait for data during distributed stream network communication.
  - **Stream Send time**: indicates the time consumed by libcomm or libpq to send data during distributed stream network communication.
  - **OS Kernel Send time**: indicates the time required for the OS layer to send data during distributed stream network communication. This parameter is displayed only when the value is greater than **0**.
  - **Wait Quota time**: indicates the duration for libcomm to wait for the peer end to send the quota traffic control size during distributed stream network communication. This parameter is displayed only when the value is greater than **0**.
  - **Data Serialize time**: indicates the data serialization time during distributed stream network communication.
  - **Data Copy time**: indicates the data replication time during distributed stream network communication. This parameter is displayed only when the value is greater than **0**.

- **METADATA boolean**

  Specifies whether to display the statement metadata. If **boolean** is not specified, **TRUE** is used by default. This part is displayed only when the statement uses a stream plan. The following describes the output information:

  - Defined objects, including tables, indexes, views, functions, and operators referenced in statements.
  - Statistics, which are classified into table-level and column-level statistics. Their common fields are as follows:

    - **nspname**: schema name, corresponding to **pg_namespace.nspname**.
    - **relname**: table name, corresponding to the **pg_class.relname** of a table or index.
    - **partname**: partition name, corresponding to the **pg_partition.relname** of a partition.
    - **subpartname**: subpartition name, corresponding to the **pg_partition.relname** of a subpartition.

    Other table-level statistics include:

    - **reltuples**: total number of rows in the table, corresponding to **pg_class/pg_partition.reltuples**.
    - **relpages**: total number of pages in the table, corresponding to **pg_class/pg_partition.relpages**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **analyze_time**: timestamp when the table or index was last analyzed, which corresponds to the return value of pg_stat_get_last_analyze_time or gs_stat_get_partition_last_analyze_time.
- **rows_changed**: number of rows changed in the table or index since the last ANALYZE operation, which corresponds to the return value of pg_stat_get_tuples_changed or pg_stat_get_partition_tuples_changed.
- **rows_inserted**: number of rows inserted into the table or index since the last ANALYZE operation. The value is null.

Other column-level statistics (including single-column and multi-column statistics) are as follows:

- **colnames**: column name. For a single column, the value is the same as that of **pg_stats.attname**. For multiple columns, the value is the set of column names represented by the column numbers in **pg_ext_stats.attname**.
- **nullfrac**: null fraction, which corresponds to **pg_stats/ pg_ext_stats.nullfrac**.
- **avgwidth**: average row width, which corresponds to **pg_stats/ pg_ext_stats.avgwidth**.
- **distinct**: number of distinct values, which corresponds to **pg_stats/ pg_ext_stats.n_distinct**.
- **dndistinct**: number of distinct values on a DN, which corresponds to **pg_stats/pg_ext_stats.n_dndistinct**.
- **mcvs_top5**: top 5 most frequent values, which corresponds to **pg_stats/pg_ext_stats.most_common_vals**.
- **mcfs_top5**: frequencies of the top 5 most frequent values, which correspond to **pg_stats/pg_ext_stats.most_common_freqs**.
- **hist_qrtl**: histogram bucket value. For a single column, the value is the same as that of **pg_stats.histogram_bounds**. Only five values on the quartile boundary are used. For multiple columns, the value is empty.
- **correlation**: correlation. For a single column, the value is the same as that of **pg_stats.correlation**. For multiple columns, the value is **0**.

– In **Query Others**, the following information is included:

- **current time**: current timestamp, which is used to compare with **analyze_time**.
- **use_hints: Yes,** *<string>*: If hints are effective for a statement, the hints are printed in a list.

Value range:

– **TRUE** (default value): The statement metadata is displayed. This option must be used in pretty mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - **FALSE**: not displayed.

- **TABLES boolean**

  Specifies whether to display the information of table objects on which the SQL statement depends, including tables and materialized views. If **boolean** is not specified, **TRUE** is used by default. This part is displayed only when the statement uses a stream plan.

  - **TRUE** (default value): The information about the table objects on which the statement depends is displayed. This option must be used in pretty mode.
  - **FALSE**: not displayed.

Example 1

```
-- Create a schema.
gaussdb=# CREATE SCHEMA tpcds;

-- Create the tpcds.customer_address table.
gaussdb=# CREATE TABLE tpcds.customer_address
(
ca_address_sk         INTEGER           NOT NULL,
ca_address_id         CHARACTER(16)     NOT NULL
);

-- Insert multiple records into the table.
gaussdb=# INSERT INTO tpcds.customer_address VALUES (5000, 'AAAAAAAABAAAAAAA'),(10000,
'AAAAAAAACAAAAAAA');

-- Create the tpcds.customer_address_p1 table.
gaussdb=# CREATE TABLE tpcds.customer_address_p1 AS TABLE tpcds.customer_address;

-- Change the value of explain_perf_mode to normal.
gaussdb=# SET explain_perf_mode=normal;

-- Display an execution plan for simple queries in the table.
gaussdb=# EXPLAIN SELECT * FROM tpcds.customer_address_p1;
QUERY PLAN
--------------------------------------------------
Data Node Scan  (cost=0.00..0.00 rows=0 width=0)
Node/s: All datanodes
(2 rows)

-- Use the ANALYZE option to add runtime statistics to the output.
gaussdb=# EXPLAIN ANALYZE SELECT * FROM tpcds.customer_address_p1;
                                  QUERY PLAN
-------------------------------------------------------------------------------------------
 Data Node Scan  (cost=0.00..0.00 rows=0 width=0) (actual time=1.754..3.218 rows=2 loops=1)
   Node/s: All datanodes
 Total runtime: 3.272 ms
(3 rows)

-- Use the ANALYZE and CPU options to output the CPU usage information.
gaussdb=# EXPLAIN (ANALYZE,CPU)SELECT * FROM tpcds.customer_address_p1;
                                    QUERY PLAN
-----------------------------------------------------------------------------------------------------
 Data Node Scan  (cost=0.00..0.00 rows=0 width=0) (actual time=1.996..2.214 rows=2 loops=1)
   Node/s: All datanodes
   (CPU: ex c/r=25694795469106248, ex row=2, ex cyc=51389590938212496, inc cyc=51389590938212496)
 Total runtime: 2.251 ms
(4 rows)

-- Generate an execution plan in JSON format (with explain_perf_mode being normal).
gaussdb=# EXPLAIN(FORMAT JSON) SELECT * FROM tpcds.customer_address_p1;
            QUERY PLAN
--------------------------------------
 [                              +
```

```
  {                            +
    "Plan": {                    +
      "Node Type": "Data Node Scan",+
      "Startup Cost": 0.00,        +
      "Total Cost": 0.00,          +
      "Plan Rows": 0,              +
      "Plan Width": 0,             +
      "Node/s": "All datanodes"    +
    }                            +
  }                              +
]
(1 row)


-- Generate an execution plan in YAML format (with explain_perf_mode being normal).
gaussdb=# EXPLAIN(FORMAT YAML) SELECT * FROM tpcds.customer_address_p1 WHERE
ca_address_sk=10000;
          QUERY PLAN
---------------------------------
 - Plan:                      +
    Node Type: "Data Node Scan"+
    Startup Cost: 0.00        +
    Total Cost: 0.00          +
    Plan Rows: 0              +
    Plan Width: 0             +
    Node/s: "dn_6005_6006"
(1 row)

-- Suppress the execution plan of cost estimation.
gaussdb=# EXPLAIN(COSTS FALSE)SELECT * FROM tpcds.customer_address_p1 WHERE
ca_address_sk=10000;
      QUERY PLAN
-----------------------
 Data Node Scan
   Node/s: dn_6005_6006
(2 rows)

-- Generate an execution plan with aggregate functions for a query.
gaussdb=# EXPLAIN SELECT SUM(ca_address_sk) FROM tpcds.customer_address_p1 WHERE
ca_address_sk<10000;
                                QUERY PLAN
-----------------------------------------------------------------------------------------
 Aggregate  (cost=18.19..14.32 rows=1 width=4)
   ->  Streaming (type: GATHER)  (cost=18.19..14.32 rows=3 width=4)
         Node/s: All datanodes
         ->  Aggregate  (cost=14.19..14.20 rows=3 width=4)
               ->  Seq Scan on customer_address_p1  (cost=0.00..14.18 rows=10 width=4)
                     Filter: (ca_address_sk < 10000)
(6 rows)

-- Drop the tpcds.customer_address_p1 table.
gaussdb=# DROP TABLE tpcds.customer_address_p1;

-- Drop the tpcds.customer_address table.
gaussdb=# DROP TABLE tpcds.customer_address;

-- Drop the schema.
gaussdb=# DROP SCHEMA tpcds CASCADE;
```

Example 2

```
# Display the metadata of the SQL statement.
# Connect to a database using gsql.
[omm@localhost ~]$ gsql -d postgres -p 8000 -r

-- Set explain_perf_mode.
gaussdb=# SET explain_perf_mode = pretty;

-- Disable the Bayesian network for collecting statistics about multiple columns to collect common MCV
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
statistics about multiple columns.
gaussdb=# SET enable_ai_stats = off;

-- Disable automatic parameterization to prevent statements from being skipped during parsing.
gaussdb=# SET cursor_sharing = off;

-- Create a schema and use it by default.
gaussdb=# CREATE SCHEMA em;
gaussdb=# SET search_path = em;

-- Create the em.t1 table.
gaussdb=# CREATE TABLE em.t1 (
    c1 integer
) DISTRIBUTE BY HASH (c1) PARTITION BY RANGE (c1) (
    PARTITION p1 VALUES LESS THAN (1000),
    PARTITION p2 VALUES LESS THAN (MAXVALUE)
);

-- Add an index to em.t1.
gaussdb=# CREATE INDEX em.idx_t1_c1 ON em.t1(c1) LOCAL;

-- Insert multiple records.
gaussdb=# INSERT INTO em.t1 SELECT (s % 2000)::int FROM pg_catalog.generate_series(1, 100000) s;

-- Analyze em.t1.
gaussdb=# ANALYZE em.t1 WITH ALL COMPLETE;

-- Create the em.t2 table.
gaussdb=# CREATE TABLE em.t2 (c1 integer, c2 integer) WITH (storage_type=ASTORE) DISTRIBUTE BY
HASH (c1);

-- Insert multiple records.
gaussdb=# INSERT INTO em.t2 SELECT (s % 6)::int, (s % 3)::int FROM pg_catalog.generate_series(1, 24) s;

-- Analyze em.t2.
gaussdb=# ANALYZE em.t2;

-- Analyze columns c1 and c2 in em.t2.
gaussdb=# ANALYZE em.t2((c1,c2));

-- Create a materialized view em.mv.
gaussdb=# CREATE MATERIALIZED VIEW em.mv AS SELECT c1 FROM t2 WHERE c1 = 2 AND c2 = 2;
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'c1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.

-- Analyze the materialized view.
gaussdb=# ANALYZE em.mv;

-- Use the METADATA option to output the metadata involved in the statement.
gaussdb=# EXPLAIN (METADATA ON) SELECT t.c1, v.c1 FROM em.t1 t, em.mv v WHERE t.c1 < 500 AND
v.c1=2;
 id |                        operation                        | E-rows | E-width | E-costs
----+---------------------------------------------------------+--------+---------+---------
  1 | ->  Streaming (type: GATHER)                            | 100584 |       8 | 4764.48
  2 |    ->  Nested Loop (3,4)                                | 100584 |       8 | 573.48
  3 |       ->  Partitioned Index Only Scan using idx_t1_c1 on t1 t |  25146 |       4 | 150.24
  4 |       ->  Materialize                                   |     12 |       4 | 4.15
  5 |          ->  Streaming(type: BROADCAST)                 |     12 |       4 | 4.13
  6 |             ->  Seq Scan on mv v                        |      4 |       4 | 3.05
(6 rows)

      Predicate Information (identified by plan id)
-------------------------------------------------------------
  3 --Partitioned Index Only Scan using idx_t1_c1 on t1 t
        Index Cond: (c1 < 500)
        Selected Partitions:  1
  6 --Seq Scan on mv v
        Filter: (c1 = 2)
(5 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
                                                
```

```
      Selected Partitions:  1
  6 --Seq Scan on mv v
        Filter: (c1 = 2)
(5 rows)

           Tables Referenced
-------------------------------------------
 ('em', 't1'), ('em', 't2'), ('em', 'mv')
(1 row)

-- Clean up the created objects.
gaussdb=# DROP SCHEMA em CASCADE;
gaussdb=# \q
```

Helpful Links

ANALYZE | ANALYSE

1.13.11.6 EXPLAIN PLAN

Description

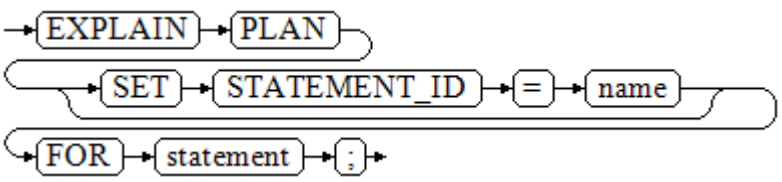
EXPLAIN PLAN saves information about an execution plan into the **PLAN_TABLE** table. Different from the EXPLAIN statement, EXPLAIN PLAN only saves plan information and does not print information on the screen.

Precautions

- EXPLAIN PLAN cannot be executed on a DN.
- Plan information cannot be collected for SQL statements that failed to be executed.
- Data in **PLAN_TABLE** is in a session-level lifecycle. Sessions are isolated from users, and therefore users can only view the data of the current session and current user.
- **PLAN_TABLE** cannot be joined with GDS foreign tables.
- For a query that cannot be pushed down, only such information as **REMOTE_QUERY** and **CTE** can be collected from **PLAN_TABLE** after **EXPLAIN PLAN** is executed. For details, see **Example 2**.
- **PLAN**: saves plan information into **PLAN_TABLE**. If information is stored successfully, "EXPLAIN SUCCESS" is returned.
- **STATEMENT_ID**: tags each query. The tag information will be stored in **PLAN_TABLE**.
- After the EXPLAIN PLAN statement is executed, plan information is automatically stored in **PLAN_TABLE**. INSERT, UPDATE, and ANALYZE cannot be performed on **PLAN_TABLE**. For details about **PLAN_TABLE**, see **PLAN_TABLE**.

Syntax

```
EXPLAIN PLAN [ SET STATEMENT_ID = name ] FOR statement ;
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **name**

  Specifies a query tag.

  Value range: a string.

  > NOTE
  >
  > If **STATEMENT_ID** is not set when the EXPLAIN PLAN statement is executed, **STATEMENT_ID** is left empty by default. In addition, the value of **STATEMENT_ID** cannot exceed 30 bytes. Otherwise, an error will be reported.

- **statement**

  Specifies the SQL statement to explain.

Example 1

```
-- Create tables foo1 and foo2.
gaussdb=# CREATE TABLE foo1(f1 int, f2 text, f3 text[]);
gaussdb=# CREATE TABLE foo2(f1 int, f2 text, f3 text[]);

-- Run EXPLAIN PLAN.
gaussdb=# EXPLAIN PLAN SET STATEMENT_ID = 'TPCH-Q4' FOR SELECT f1, count(*) FROM foo1 WHERE f1
> 1 AND f1 < 3 AND EXISTS (SELECT * FROM foo2) GROUP BY f1;

-- Query PLAN_TABLE.
gaussdb=# SELECT * FROM plan_table;

-- Delete data from the PLAN_TABLE table and delete tables foo1 and foo2.
gaussdb=# DELETE FROM plan_table WHERE STATEMENT_ID = 'TPCH-Q4';
gaussdb=# DROP TABLE foo1;
gaussdb=# DROP TABLE foo2;
```

Example 2

```
-- Disable enable_stream_recursive so that associated WITH RECURSIVE queries cannot be pushed down.
gaussdb=# SET enable_stream_recursive = off;

-- Create the chinamap table.
gaussdb=# CREATE TABLE chinamap
(
  id integer,
  pid integer,
  name text
) DISTRIBUTE BY hash(id);

-- Plan collected by plan_table.
gaussdb=# EXPLAIN PLAN SET statement_id = 'cte can not be push down'
FOR
WITH RECURSIVE rq AS
(
  SELECT id, name FROM chinamap WHERE id = 11
  UNION ALL
  SELECT origin.id, rq.name || ' > ' || origin.name
  FROM rq JOIN chinamap origin ON origin.pid = rq.id
)
SELECT id, name FROM rq ORDER BY 1;

-- Query PLAN_TABLE.
gaussdb=# SELECT * FROM plan_table;

-- Delete the chinamap table.
gaussdb=# DROP TABLE chinamap;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.11.7 EXPLAIN AUTOHINT

Description

EXPLAIN AUTOHINT stores the query execution plan information and execution result in the AUTOHINT memory context while the **EXPLAIN** command is executed, facilitating hint exploration and recommendation of the AUTOHINT feature.

Precautions

- This statement can be automatically called by the kernel only when the AUTOHINT task is running. If a user calls it, no effect will be produced or an AUTOHINT error will be reported for blocking the task.
- The data stored in the AUTOHINT memory context has the session-level lifecycle and is isolated by session and user. The AUTOHINT called in the current session allows you to access and call only the data of the current session and user.

Syntax

```
EXPLAIN AUTOHINT cache_id exec_type argument
FOR statement ;
```

Parameters

- **statement**

  SQL statement to be stored for the plan and execution result of the current AUTOHINT task. (Only the SELECT statement is supported.)

- **cache_id**

  ID of the current AUTOHINT task. The value is of the int type.

- **exec_type**

  Type of the result stored by the current statement.

  Value range:

  – **PLAN**: stores plan information.

  – **EXECUTE**: stores execution result information.

- **argument**

  Parameter required for executing the current statement. The value is in JSON format.

  If **exec_type** is set to **PLAN**, the value is the hint applied to the current statement.

  If **exec_type** is set to **EXECUTE**, the value is the hash value of the execution plan corresponding to the current statement.

Example 1

```
-- AUTOHINT executes the corresponding SQL statement to store the plan information.
gaussdb=# EXPLAIN AUTOHINT 2 PLAN
        '{"hints":[{"type": 0, "name": "enable_seqscan", "value": "true"},
         {"type": 0, "name": "enable_mergejoin", "value": "false"}]}' FOR
            SELECT * FROM t1, t2 WHERE t1.id = t2.id and t1.a > 100 and t2.b < 200;
-- The user directly executes the statement, which does not take effect.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 QUERY PLAN
------------
(0 rows)
-- The user directly executes the statement, which is blocked with an error reported.
gaussdb=# ERROR:  Can not get exploration cache for cache id: 2.
```

Example 2

```
-- AUTOHINT executes the corresponding SQL statement to store the execution information.
gaussdb=# EXPLAIN AUTOHINT 2 EXECUTE '{"topology": 740798663, "cards": [18936500, 10592, 9308,
9308]}' FOR
            SELECT * FROM t1, t2 WHERE t1.id = t2.id and t1.a > 100 and t2.b < 200;
-- The user directly executes the statement, which does not take effect.
 QUERY PLAN
------------
(0 rows)
-- The user directly executes the statement, which is blocked with an error reported.
gaussdb=# ERROR:  Can not get exploration cache for cache id: 2.
```

1.13.12 F

1.13.12.1 FETCH

Description

FETCH retrieves rows using a previously created cursor.

A cursor has an associated position, which is used by FETCH. The cursor position can be before the first row of the query result, on any particular row of the result, or after the last row of the result.

- When created, a cursor is positioned before the first row.
- After fetching some rows, the cursor is positioned on the row most recently retrieved.
- If FETCH runs off the end of the available rows then the cursor is left positioned after the last row, or before the first row if fetching backward.
- FETCH ALL or FETCH BACKWARD ALL will always leave the cursor positioned after the last row or before the first row.

Precautions

- If the cursor is declared with **NO SCROLL**, backward fetches like FETCH BACKWARD are not allowed.
- The forms **NEXT**, **PRIOR**, **FIRST**, **LAST**, **ABSOLUTE**, and **RELATIVE** fetch a single row after moving the cursor appropriately. If there is no such row, an empty result is returned, and the cursor is left positioned before the first row (backward fetch) or after the last row (forward fetch) as appropriate.
- The forms using **FORWARD** and **BACKWARD** retrieve the indicated number of rows moving in the forward or backward direction, leaving the cursor positioned on the last-returned row or after (backward fetch)/before (forward fetch) all rows if the **count** exceeds the number of rows available.
- **RELATIVE 0**, **FORWARD 0**, and **BACKWARD 0** all request fetching the current row without moving the cursor, that is, re-fetching the most recently fetched row. This action will succeed unless the cursor is positioned before the first

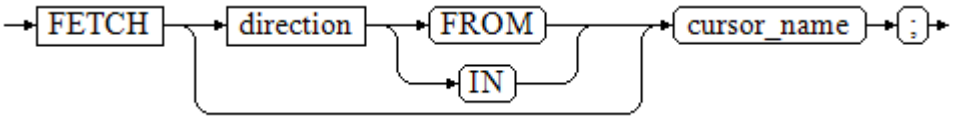
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

row or after the last row. If the cursor is positioned before the first row or after the last row, no row is returned.

- If the cursor of FETCH involves a non-system catalog, backward fetches like **BACKWARD**, **PRIOR**, and **FIRST** are not allowed.

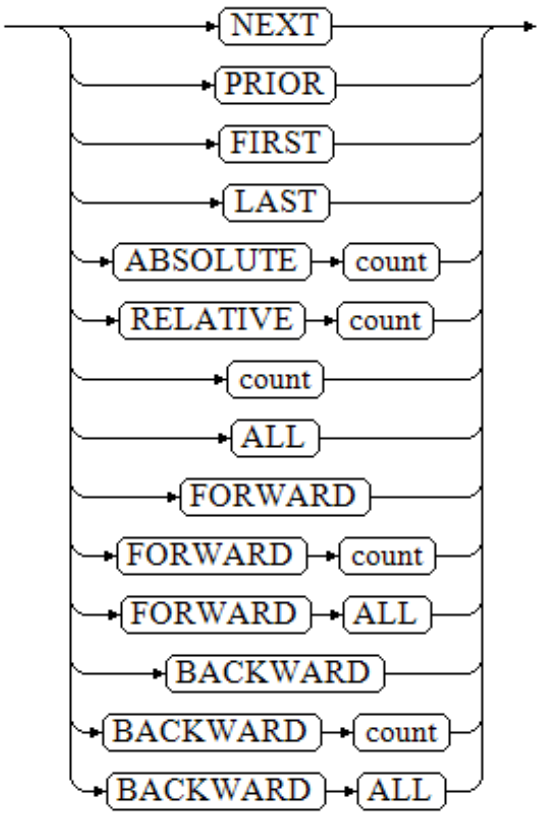
Syntax

```
FETCH [ direction { FROM | IN } ] cursor_name;
```



The **direction** clause specifies optional parameters.

```
{ NEXT
  | PRIOR
  | FIRST
  | LAST
  | ABSOLUTE count
  | RELATIVE count
  | count
  | ALL
  | FORWARD
  | FORWARD count
  | FORWARD ALL
  | BACKWARD
  | BACKWARD count
  | BACKWARD ALL }
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **direction**

  Defines the fetch direction.

  Value range:

  - NEXT (default value)

    Fetches the next row.
  - PRIOR

    Fetches the prior row.
  - FIRST

    Fetches the first row of the query (same as **ABSOLUTE 1**).
  - LAST

    Fetches the last row of the query (same as **ABSOLUTE –1**).
  - ABSOLUTE count

    Fetches the *count*th row of the query.

    ABSOLUTE fetches are not any faster than navigating to the desired row with a relative move: the underlying implementation must traverse all the intermediate rows anyway.

    Value range: a possibly-signed integer

    - If *count* is positive, the *count*th row of the query will be fetched. If *count* is less than the current cursor position, rewind is required, which is currently not supported.
    - If *count* is negative or 0, backward scan is required, which is currently not supported.
  - RELATIVE count

    Fetches the *count*th succeeding row or the *count*th prior row.

    Value range: signed integer

    - If *count* is a positive integer, the *count*th succeeding row will be fetched.
    - If *count* is a negative integer or 0, backward scan is required, which is currently not supported.
    - If the current row contains no data, **RELATIVE 0** returns null.
  - count

    Fetches the next *count* rows (same as **FORWARD** *count*).
  - ALL

    Fetches all remaining rows (same as **FORWARD ALL**).
  - FORWARD

    Fetches the next row (same as **NEXT**).
  - FORWARD count

    Fetches the next or prior *count* rows (same as **RELATIVE** *count*).
  - FORWARD ALL

    Fetches all remaining rows.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - BACKWARD

    Fetches the prior row (same as **PRIOR**).
  - BACKWARD count

    Fetches the prior *count* rows (scanning backwards).

    Value range: signed integer

    - If *count* is a positive integer, the *count*th prior row will be fetched.
    - If *count* is a negative integer, the *abs(count)*th succeeding row will be fetched.
    - **BACKWARD 0** re-fetches the current row, if any.
  - BACKWARD ALL

    Fetches all prior rows (scanning backwards).
- **{ FROM | IN } cursor_name**

  Specifies the cursor name using the keyword **FROM** or **IN**.

  Value range: an existing cursor name

Examples

```
-- Create the test table and insert 20 data records into the table.
gaussdb=# CREATE TABLE test(c1 int, c2 int);
gaussdb=# INSERT INTO test VALUES (generate_series(1,20),generate_series(1,20));

-- Start a transaction and create a cursor named cursor1.
gaussdb=# START TRANSACTION;
gaussdb=# CURSOR cursor1 FOR SELECT * FROM test ORDER BY 1;

-- The cursor retrieves three rows of data from the associated position.
gaussdb=# FETCH FORWARD 3 FROM cursor1;
 c1 | c2
----+----
  1 |  1
  2 |  2
  3 |  3
(3 rows)

-- Close the cursor and commit the transaction.
gaussdb=# CLOSE cursor1;
gaussdb=# END;

-- Drop the table.
gaussdb=# DROP TABLE test;
```

Helpful Links

CLOSE, **MOVE**, and **CURSOR**

1.13.13 G

1.13.13.1 GENERATED UPDATE SYSTEM OBJECT

Description

Generates an upgrade rollback script for system objects. OM will call this script during upgrade rollback. This is an internal syntax; its use is therefore discouraged.

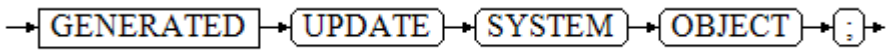
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

- This statement is called exclusively by OM during upgrades. Specifically, it can be executed only when the GUC parameter **upgrade_mode** is not **0** and **application_name** is set to **OM**.
- Only the initial user has permission to execute this statement.

Syntax

```
GENERATED UPDATE SYSTEM OBJECT;
```



1.13.13.2 GRANT

Description

Grants permissions to roles and users.

GRANT is used in the following scenarios:

- **Granting system permissions to roles or users**

  System permissions are also called user attributes, including SYSADMIN, CREATEDB, CREATEROLE, AUDITADMIN, MONADMIN, OPRADMIN, POLADMIN, INHERIT, REPLICATION, and LOGIN.

  They can be specified only by the CREATE ROLE or ALTER ROLE statement. The SYSADMIN permissions can be granted and revoked using GRANT ALL PRIVILEGE and REVOKE ALL PRIVILEGE, respectively. System permissions cannot be inherited by roles or users, nor can they be granted to **PUBLIC**.

- **Granting database object permissions to roles or users**

  Grant permissions on a database object (table, view, column, database, function, schema, or tablespace) to a role or user.

  GRANT gives specific permissions on a database object to one or more roles. These permissions are added to those already granted, if any.

  PUBLIC indicates that the permissions are to be granted to all roles, including those that might be created later. **PUBLIC** can be thought of as an implicitly defined group that always includes all roles. Any particular role will have the sum of permissions granted directly to it, permissions granted to any role it is presently a member of, and permissions granted to **PUBLIC**.

  If **WITH GRANT OPTION** is specified, the recipient of the permission can in turn grant it to others. Without a grant option, the recipient cannot do that. This option cannot be granted to PUBLIC, which is a unique GaussDB attribute.

  GaussDB grants the permissions on certain types of objects to PUBLIC. By default, permissions on tables, columns, sequences, foreign data sources, foreign servers, schemas, and tablespaces are not granted to PUBLIC, but the following permissions are granted to PUBLIC: CONNECT and CREATE TEMP TABLE permissions on databases, EXECUTE permission on functions, and USAGE permission on languages and data types. An object owner can revoke the default permissions granted to **PUBLIC** and grant permissions to other users as needed. For security purposes, you are advised to create an object

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

and set its permissions in the same transaction so that other users do not have time windows to use the object. These default permissions can be modified by running the **ALTER DEFAULT PRIVILEGES** command.

By default, an object owner has all permissions on the object. For security purposes, the owner can discard some permissions. However, the ALTER, DROP, COMMENT, INDEX, VACUUM, and re-grantable permissions of the object are inherent permissions implicitly owned by the owner.

- **Granting the permissions of one role or user to others**

  The permissions of one role or user can be granted to others. In this case, every role or user can be regarded as a set of one or more database permissions.

  If **WITH ADMIN OPTION** is specified, the recipients can in turn grant the permissions to or revoke them from others. If the grantors' permissions are changed or revoked later, the grantees' permissions will also change.

  When separation of duties is disabled, a system administrator can grant or revoke the permissions of any non-permanent user or O&M administrator. Security administrators can grant or revoke permissions of any non-system administrator, built-in role, permanent user or O&M administrator.

- **Granting ANY permissions to roles or users**

  The ANY permissions can be granted to specific roles or users. See the "Syntax" section for the scope of ANY permissions. If **WITH ADMIN OPTION** is specified, the recipients can in turn grant the ANY permissions to others or revoke them from others. The ANY permissions can be inherited by a role but cannot be granted to **PUBLIC**. The initial user and system administrators (when separation of duties is disabled) can grant the ANY permissions to or revoke them from any roles or users.

  Currently, the following ANY permissions are supported: CREATE ANY TABLE, ALTER ANY TABLE, DROP ANY TABLE, SELECT ANY TABLE, INSERT ANY TABLE, UPDATE ANY TABLE, DELETE ANY TABLE, TRUNCATE ANY TABLE, CREATE ANY SEQUENCE, CREATE ANY INDEX, CREATE ANY FUNCTION, EXECUTE ANY FUNCTION, CREATE ANY PACKAGE, EXECUTE ANY PACKAGE, CREATE ANY TYPE, ALTER ANY TYPE, DROP ANY TYPE, ALTER ANY SEQUENCE, DROP ANY SEQUENCE, SELECT ANY SEQUENCE, ALTER ANY INDEX, DROP ANY INDEX, CREATE ANY SYNONYM, DROP ANY SYNONYM, CREATE ANY TRIGGER, ALTER ANY TRIGGER, and DROP ANY TRIGGER. For details about the ANY permission scope, see **Table 1-402**.

Precautions

- The ANY permissions cannot be granted to **PUBLIC**, nor can they be revoked from **PUBLIC**.
- The ANY permissions are database-specific and apply only to objects within the granting database. For example, SELECT ANY TABLE allows users to view data from all user tables in the current database, but does not grant access to user tables in other databases.
- The ANY permissions operate independently of existing permissions.
- If a user is granted the CREATE ANY TABLE permission, the owner of a table created in a schema with the same name as the user is the creator of the schema. When the user performs other operations on the table, the user needs to be granted the corresponding operation permission. Similarly, if a

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

user is granted the CREATE ANY FUNCTION, CREATE ANY PACKAGE, CREATE ANY TYPE, CREATE ANY SEQUENCE, or CREATE ANY INDEX permission, the owner of an object created in a schema with the same name is the owner of the schema. If a user is granted the CREATE ANY TRIGGER or CREATE ANY SYNONYM permission, the owner of an object created in a schema with the same name is the creator.

- Exercise caution when granting the CREATE ANY FUNCTION or CREATE ANY PACKAGE permission to users to prevent other users from using DEFINER functions or PACKAGE for privilege escalation.
- When GRANT is used to grant a user the permission to use a table, if the permission is not properly used, ALTER may be used to add expressions to the default values and constraints of the table, or indexes may be created to add expressions to INDEX. In this case, the permission may be exploited.
- When GRANT is used to grant the TRIGGER permission, if the permission is not properly used, the WHEN condition may be used to create expressions. When the trigger is triggered, the permission may be exploited.
- When granting permissions to users, pay special attention to the definer permission on functions/packages. The definer permission is executed as the owner of the functions/packages. If the permission is not properly granted (including GRANT ROLE TO ROLE), the permission may be exploited.
- Do not grant object permissions to too many users. You can use roles or PUBLIC based on service requirements.

Syntax

- Grant the table or view access permission to a user or role.

```
GRANT { { SELECT | INSERT | UPDATE | DELETE | TRUNCATE | REFERENCES | TRIGGER | ALTER | DROP
| COMMENT | INDEX | VACUUM } [, ...]
    | ALL [ PRIVILEGES ] }
  ON { [ TABLE ] table_name [, ...]
    | ALL TABLES IN SCHEMA schema_name [, ...] }
  TO { [ GROUP ] role_name | PUBLIC } [, ...]
  [ WITH GRANT OPTION ];
```

- Grant the column access permission to a user or role.

```
GRANT { {{ SELECT | INSERT | UPDATE | REFERENCES | COMMENT } ( column_name [, ...] )} [, ...]
    | ALL [ PRIVILEGES ] ( column_name [, ...] ) }
  ON [ TABLE ] table_name [, ...]
  TO { [ GROUP ] role_name | PUBLIC } [, ...]
  [ WITH GRANT OPTION ];
```

  NOTE

  If you have permission to access a table, you automatically have permission to access all columns within that table by default. To grant access only to a specific column in a table, you must revoke the table-level access permission first.

- Grant the sequence access permission to a specified user or role.

```
GRANT { { SELECT | UPDATE | USAGE | ALTER | DROP | COMMENT } [, ...]
    | ALL [ PRIVILEGES ] }
  ON { [ SEQUENCE ] sequence_name [, ...]
    | ALL SEQUENCES IN SCHEMA schema_name [, ...] }
  TO { [ GROUP ] role_name | PUBLIC } [, ...]
  [ WITH GRANT OPTION ];
```

- Grant the database access permission to a user or role.

```
GRANT { { CREATE | CONNECT | TEMPORARY | TEMP | ALTER | DROP | COMMENT } [, ...]
    | ALL [ PRIVILEGES ] }
  ON DATABASE database_name [, ...]
  TO { [ GROUP ] role_name | PUBLIC } [, ...]
  [ WITH GRANT OPTION ];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Grant the CMK access permission to a specified user or role.

```
GRANT { { USAGE | DROP } [, ...] | ALL [ PRIVILEGES ] }
    ON CLIENT_MASTER_KEY client_master_key
    TO { [ GROUP ] role_name | PUBLIC } [, ...]
    [ WITH GRANT OPTION ];
```

- Grant the column encryption key (CEK) access permission to a specified user or role.

```
GRANT { { USAGE | DROP } [, ...] | ALL [ PRIVILEGES ] }
    ON COLUMN_ENCRYPTION_KEY column_encryption_key
    TO { [ GROUP ] role_name | PUBLIC } [, ...]
    [ WITH GRANT OPTION ];
```

- Grant the foreign data source access permission to a user or role.

```
GRANT { USAGE | ALL [ PRIVILEGES ] }
    ON FOREIGN DATA WRAPPER fdw_name [, ...]
    TO { [ GROUP ] role_name | PUBLIC } [, ...]
    [ WITH GRANT OPTION ];
```

- Grant the foreign server access permission to a user or role.

```
GRANT { { USAGE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
    ON FOREIGN SERVER server_name [, ...]
    TO { [ GROUP ] role_name | PUBLIC } [, ...]
    [ WITH GRANT OPTION ];
```

- Grant the function access permission to a user or role.

```
GRANT { { EXECUTE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
    ON { FUNCTION {function_name ( [ {[ argmode ] [ arg_name ] arg_type} [, ...] ] )} [, ...]
      | ALL FUNCTIONS IN SCHEMA schema_name [, ...] }
    TO { [ GROUP ] role_name | PUBLIC } [, ...]
    [ WITH GRANT OPTION ];
```

- Grant the procedural procedure access permission to a user or role.

```
GRANT { { EXECUTE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
    ON PROCEDURE {proc_name ( [ {[ argmode ] [ arg_name ] arg_type} [, ...] ] )} [, ...]
    TO { [ GROUP ] role_name | PUBLIC } [, ...]
    [ WITH GRANT OPTION ];
```

- Grant the procedural language access permission to a user or role.

```
GRANT { USAGE | ALL [ PRIVILEGES ] }
    ON LANGUAGE lang_name [, ...]
    TO { [ GROUP ] role_name | PUBLIC } [, ...]
    [ WITH GRANT OPTION ];
```

NOTE

In the current version, all users can create C functions, whereas only users with the SYSADMIN permission can create Java and internal functions.

- When a user with the SYSADMIN permission grants the permission for creating C functions to others, the user must specify a recipient and cannot use GRANT USAGE ON LANGUAGE c TO PUBLIC.
- When a user with the SYSADMIN permission grants the permission for creating C functions to others, the user cannot specify WITH GRANT OPTION.

- Grant the sub-cluster access permission to a user or role.

```
GRANT { { CREATE | USAGE | COMPUTE | ALTER | DROP } [, ...] | ALL [ PRIVILEGES ] }
    ON NODE GROUP group_name [, ...]
    TO { [ GROUP ] role_name | PUBLIC } [, ...]
    [ WITH GRANT OPTION ];
```

NOTE

When the CREATE permission of a sub-cluster is granted to a specified user or role, the USAGE and COMPUTE permissions are granted to the specified user or role by default.

- Grant the schema access permission to a user or role.

```
GRANT { { CREATE | USAGE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
    ON SCHEMA schema_name [, ...]
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  TO { [ GROUP ] role_name | PUBLIC } [, ...]
  [ WITH GRANT OPTION ];
```

NOTE

When granting permissions on tables or views within a schema to other users, you also need to grant the USAGE permission on the schema containing those objects. Without the USAGE permission, grantees can only see the object names but cannot actually access them.

- Grant the tablespace access permission to a user or role.

```
GRANT { { CREATE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
  ON TABLESPACE tablespace_name [, ...]
  TO { [ GROUP ] role_name | PUBLIC } [, ...]
  [ WITH GRANT OPTION ];
```

- Grant the type access permission to a user or role.

```
GRANT { { USAGE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
  ON TYPE type_name [, ...]
  TO { [ GROUP ] role_name | PUBLIC } [, ...]
  [ WITH GRANT OPTION ];
```

NOTE

In the current version, the type access permission cannot be granted.

- Grant the DIRECTORY permission to a role.

```
GRANT { { READ | WRITE | ALTER | DROP } [, ...] | ALL [PRIVILEGES] }
  ON DIRECTORY directory_name [, ...]
  TO { [GROUP] role_name | PUBLIC } [, ...]
  [WITH GRANT OPTION];
```

- Grant the PACKAGE permission to a role.

```
GRANT { { EXECUTE | ALTER | DROP | COMMENT } [, ...] | ALL [PRIVILEGES] }
  ON PACKAGE package_name [, ...]
  TO { [GROUP] role_name | PUBLIC } [, ...]
  [WITH GRANT OPTION];
```

- Grant a role's permissions to other users or roles.

```
GRANT role_name [, ...]
  TO role_name [, ...]
  [ WITH ADMIN OPTION ];
```

NOTE

After the execution of **GRANT role1 to role2**, role2 does not inherit system permissions from role1. However, as role2 acquires all object permissions from role1, there is a potential risk of exploiting role1's system permissions. Exercise caution when granting a role with system permissions to other users or roles.

- Grant the SYSADMIN permission to a role.

```
GRANT ALL { PRIVILEGES | PRIVILEGE }
  TO role_name;
```

- Grant the ANY permissions to a user or role.

```
GRANT { CREATE ANY TABLE | ALTER ANY TABLE | DROP ANY TABLE | SELECT ANY TABLE |
INSERT ANY TABLE | UPDATE ANY TABLE | DELETE ANY TABLE | TRUNCATE ANY TABLE | CREATE ANY
SEQUENCE |
CREATE ANY INDEX | CREATE ANY FUNCTION | EXECUTE ANY FUNCTION | CREATE ANY PACKAGE |
EXECUTE ANY PACKAGE | CREATE ANY TYPE | ALTER ANY TYPE | DROP ANY TYPE |
ALTER ANY SEQUENCE | DROP ANY SEQUENCE | SELECT ANY SEQUENCE | ALTER ANY INDEX |
DROP ANY INDEX | CREATE ANY SYNONYM | DROP ANY SYNONYM | CREATE ANY TRIGGER | ALTER
ANY TRIGGER |
  DROP ANY TRIGGER } [, ...]
  TO [ GROUP ] role_name [, ...]
  [ WITH ADMIN OPTION ];
```

- Grant the database link object permission to a specified user.

```
GRANT { CREATE | ALTER | DROP } [PUBLIC] DATABASE LINK TO role_name;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- **PUBLIC**: creates a public database link visible to all users. If this clause is omitted, the database link is private and used only as a compatible API. The data that can be accessed on the remote database depends on the identity used by the database link during connection.
- When the permission to create a database link is granted to a user, the user can remotely access a database by using the IP address of the remote database. Exercise caution when granting this permission to users.
- In addition to the statement for directly granting the database link permission, you can also obtain the database link permission by inheriting permission and granting permission to an administrator.
- For details about database links, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.

- Grant the permission to create PUBLIC synonyms to a specified user.

```
GRANT { CREATE | DROP } PUBLIC SYNONYM TO role_name [ WITH ADMIN OPTION ];
```

  Built-in roles (gs_role_public_synonym_create and gs_role_public_synonym_drop) can also be used to grant users the permission to create and delete PUBLIC synonyms.

  – Grant the permission to create PUBLIC synonyms to a user.

```
GRANT gs_role_public_synonym_create TO role_name;
```

  – Grant the permission to delete PUBLIC synonyms to a user.

```
GRANT gs_role_public_synonym_drop TO role_name;
```

Parameters

In GRANT, permissions are classified as follows:

- **SELECT**

  Allows SELECT from any table, view, or sequence. The SELECT permission on the corresponding column is also required for UPDATE or DELETE.

- **INSERT**

  Allows INSERT of a new row into a table.

- **UPDATE**

  Allows you to execute the UPDATE statement on any column in the specified table. The UPDATE statement also requires the SELECT permission to query which rows need to be updated. In addition to the SELECT permission, SELECT ... FOR UPDATE and SELECT ... FOR SHARE require the UPDATE permission.

- **DELETE**

  Allows DELETE of a row from a table. DELETE also requires the SELECT permission to query which rows need to be deleted.

- TRUNCATE

  Allows TRUNCATE on a table.

- **REFERENCES**

  Allows to create a foreign key constraint referencing a table. This permission is required on both referencing and referenced tables. This function is not supported in distributed mode.

- **TRIGGER**

  Allows the creation of a trigger on the specified table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **CREATE**
  - For databases, allows new schemas to be created within the database.
  - Allows users to create objects within a schema. To rename an existing object, they must own the object and have the CREATE permission on the schema containing that object.
  - Allows users to create tables within a tablespace. When creating a database or schema, they can specify the tablespace as the default tablespace.
  - For sub-clusters, allows tables to be created within the sub-cluster.
- **CONNECT**

  Allows the grantee to connect to the database.
- **TEMPORARY**

  Allows users to create temporary tables in the database.
- **READ**

  Allows users to read a specified directory.
- **WRITE**

  Allows users to write data to a specified directory.
- **EXECUTE**

  Allows calling a function, including use of any operators that are implemented on top of the function.
- **USAGE**
  - For procedural languages, allows use of the language for the creation of functions in that language.
  - Allows users to access objects within a schema. Without the USAGE permission, users can only see the object names.
  - Allows the use of the nextval function for sequences.
  - For sub-clusters, allows users who can access objects contained in the schema to access tables in the sub-cluster.
  - For a key object, USAGE is the permission to use the key.
- **COMPUTE**

  For computing sub-clusters, allows users to perform elastic computing in the sub-cluster that they have the compute permission on.
- **ALTER**

  Allows users to modify the attributes of an object, excluding the object owner and schema.
- **DROP**

  Allows users to drop an object.
- **COMMENT**

  Allows users to define or modify the comment of an object.
- **INDEX**

  Allows users to create indexes on specified tables, manage indexes on the tables, and perform REINDEX and CLUSTER operations on the tables.
- **VACUUM**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Allows users to analyze and vacuum a table.

- **ALL PRIVILEGES**

  Grants all available permissions to a user or role at a time. Only system administrators can execute GRANT ALL PRIVILEGES.

GRANT parameters are as follows:

- **role_name**

  Specifies an existing username.
- **table_name**

  Specifies an existing table name.
- **column_name**

  Specifies an existing column name.
- **schema_name**

  Specifies an existing schema name.
- **database_name**

  Specifies the database name.
- **function_name**

  Specifies the function name.
- **sequence_name**

  Specifies an existing sequence name.
- **fdw_name**

  Specifies the foreign data wrapper name.
- **lang_name**

  Specifies the language name.
- **type_name**

  Specifies an existing type name.
- **group_name**

  Specifies the sub-cluster name.
- **argmode**

  Specifies the parameter mode.

  Value range: character strings complying with **Identifier Naming Conventions**.
- **arg_name**

  Specifies the parameter name.

  Value range: character strings complying with **Identifier Naming Conventions**.
- **arg_type**

  Specifies the parameter type.

  Value range: character strings complying with **Identifier Naming Conventions**.
- **tablespace_name**

  Specifies the tablespace name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **client_master_key**

  Name of the CMK.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **column_encryption_key**

  Name of the column encryption key.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **directory_name**

  Specifies the directory name.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **WITH GRANT OPTION**

  If **WITH GRANT OPTION** is specified, the recipient of the permission can in turn grant it to others. Without a grant option, the recipient cannot do that. Grant options cannot be granted to **PUBLIC**.

When a non-owner of an object attempts to grant permissions on the object:

- The statement will fail outright if the user has no permissions whatsoever on the object.
- As long as some permission is available, the statement will proceed, but it will grant only those permissions for which the user has grant options.
- The GRANT ALL PRIVILEGES forms will issue a warning message if no grant options are held, while the other forms will issue a warning if grant options for any of the permissions specifically named in the statement are not held.

NOTE

When separation of duties is disabled, database administrators can access all objects, regardless of object permission settings. It is unwise to operate as a system administrator except when necessary.

- **WITH ADMIN OPTION**

  When **WITH ADMIN OPTION** is specified for granting a role, the recipients can in turn grant the role to or revoke it from others.

  When **WITH ADMIN OPTION** is specified for granting ANY permissions, the recipients can in turn grant the permissions to or revoke them from others.

Table 1-402 ANY permissions

Permission	Description
CREATE ANY TABLE	This permission allows users to create tables or views in the schemas of user **PUBLIC** and other common users. The users must be granted the permission to create sequences to create a table that contains serial columns.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Permission	Description
ALTER ANY TABLE	This permission allows users to alter tables or views in the schemas of user **PUBLIC** and other common users. To add PRIMARY KEY or UNIQUE constraints to the table by modifying the table's unique index, the users must also be granted the index permission on the table.
DROP ANY TABLE	This permission allows users to drop tables or views in the schemas of user **PUBLIC** and other common users.
SELECT ANY TABLE	Users' SELECT permission on tables or views in the public and user schemas, which is still subject to row-level security.
UPDATE ANY TABLE	Users' UPDATE permission on tables or views in the public and user schemas, which is still subject to row-level security.
INSERT ANY TABLE	This permission allows users to insert data into tables or views in the schemas of user **PUBLIC** and other common users.
DELETE ANY TABLE	Users' DELETE permission on tables or views in the public and user schemas, which is still subject to row-level security.
TRUNCATE ANY TABLE	This permission allows users to truncate tables or views in the schemas of user **PUBLIC** and other common users.
CREATE ANY FUNCTION	Users can create functions or stored procedures in the user schemas.
EXECUTE ANY FUNCTION	Users' EXECUTE permission on functions or stored procedures in the public and user schemas.
CREATE ANY PACKAGE	Users' permission to create packages in the public and user schemas.
EXECUTE ANY PACKAGE	Users' EXECUTE permission on packages in the public and user schemas.
CREATE ANY TYPE	Users can create types in the public and user schemas.
CREATE ANY SEQUENCE	This permission allows users to create sequences in the schemas of user **PUBLIC** and other common users.
CREATE ANY INDEX	This permission allows users to create indexes in the schemas of user **PUBLIC** and other common users. To create a partitioned table index in a tablespace, the users must also be granted the CREATE permission on that tablespace.
ALTER ANY TYPE	Users have the ALTER permission on types in public and user schemas, excluding modifying the owner of a type or modifying the schema of a type.
DROP ANY TYPE	Users' DROP permission on types in the public and user schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Permission	Description
ALTER ANY SEQUENCE	This permission allows users to alter sequences (excluding sequence owners) in the schemas of user **PUBLIC** and other common users.
DROP ANY SEQUENCE	This permission allows users to drop sequences in the schemas of user **PUBLIC** and other common users.
SELECT ANY SEQUENCE	This permission allows users to select, use, and update sequences in the schemas of user **PUBLIC** and other common users.
ALTER ANY INDEX	This permission allows users to alter indexes in the schemas of user **PUBLIC** and other common users. To rename an index, the users must also have permission to create objects in the schema containing that index. If tablespace operations are involved, the corresponding tablespace permissions are required as well. To set an index as **UNUSABLE**, they still require the DROP ANY INDEX permission.
DROP ANY INDEX	This permission allows users to drop indexes in the schemas of user **PUBLIC** and other common users.
CREATE ANY TRIGGER	Users can create triggers in the public and user schemas.
ALTER ANY TRIGGER	Users' ALTER permission on triggers in the public and user schemas.
DROP ANY TRIGGER	Users' DROP permission on triggers in the public and user schemas.
CREATE ANY SYNONYM	Users can create synonyms in user schema.
DROP ANY SYNONYM	Users' DROP permission on synonyms in the public and user schemas.

NOTE

If a user is granted any ANY permission, the user has the USAGE permission on the public and user schemas but does not have the USAGE permission on the system schemas except public listed in **Table 8-1**.

Examples

Example 1: Grant system permissions to a user or role.

Create user **joe** and grant the SYSADMIN permission to it.

```
gaussdb=# CREATE USER joe PASSWORD 'xxxxxxxxxx';
gaussdb=# GRANT ALL PRIVILEGES TO joe;
```

On success, **joe** gains the SYSADMIN permission.

Example 2: Grant object permissions to a user or role.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1. Revoke the SYSADMIN permission from the **joe** user. Grant the usage permission of the **tpcds** schema and all permissions on the **tpcds.reason** table to **joe**.

```
gaussdb=# CREATE SCHEMA tpcds;
gaussdb=# CREATE TABLE tpcds.reason (
r_reason_sk         INTEGER      NOT NULL,
r_reason_id         CHAR(16)     NOT NULL,
r_reason_desc       VARCHAR(20)
);
gaussdb=# REVOKE ALL PRIVILEGES FROM joe;
gaussdb=# GRANT USAGE ON SCHEMA tpcds TO joe;
gaussdb=# GRANT ALL PRIVILEGES ON tpcds.reason TO joe;
```

   Then **joe** has all permissions on the **tpcds.reason** table, including create, retrieve, update, and delete.

2. Grant the retrieve permission of **r_reason_sk**, **r_reason_id**, and **r_reason_desc** columns and the update permission of the **r_reason_desc** column in the **tpcds.reason** table to **joe**.

```
gaussdb=# GRANT select (r_reason_sk,r_reason_id,r_reason_desc),update (r_reason_desc) ON
tpcds.reason TO joe;
```

   Then **joe** has the retrieve permission on the **r_reason_sk**, **r_reason_id**, and **r_reason_desc** columns in the **tpcds.reason** table. To enable **joe** to grant these permissions to other users, execute the following statement:

```
gaussdb=# GRANT select (r_reason_sk, r_reason_id) ON tpcds.reason TO joe WITH GRANT OPTION;
```

   Grant the connection and schema creation permissions of the **testdb** database to **joe**, and allow **joe** to grant these permissions to other users.

```
gaussdb=# CREATE DATABASE testdb;
gaussdb=# GRANT create,connect on database testdb TO joe WITH GRANT OPTION;
```

   Create the **tpcds_manager** role, grant the access and object creation permissions of the tpcds schema to **tpcds_manager**, but do not allow **tpcds_manager** to grant these permissions to others.

```
gaussdb=# CREATE ROLE tpcds_manager PASSWORD 'xxxxxxxxxx';
gaussdb=# GRANT USAGE,CREATE ON SCHEMA tpcds TO tpcds_manager;
```

   Grant all permissions on the **tpcds_tbspc** tablespace to **joe**, but do not allow **joe** to grant these permissions to others.

```
gaussdb=# CREATE TABLESPACE tpcds_tbspc RELATIVE LOCATION 'tablespace/tablespace_1';
gaussdb=# GRANT ALL ON TABLESPACE tpcds_tbspc TO joe;
```

   Grant the PUBLIC synonym permission on creating the **tpcds.reason** table to **joe** using the built-in role (**gs_role_public_synonym_create**).

```
gaussdb=# CREATE OR REPLACE SYNONYM t1 FOR tpcds.reason;
gaussdb=# GRANT gs_role_public_synonym_create TO joe;
```

3. Create the fun1 function in the tpcds schema, and grants the ALTER permission of the fun1 function to **joe**.

```
gaussdb=# CREATE or replace FUNCTION tpcds.fun1() RETURN boolean AS
BEGIN
SELECT current_user;
RETURN true;
END;
/
gaussdb=# GRANT ALTER ON FUNCTION tpcds.fun1() TO joe;
```

Example 3: Grant a user's or role's permissions to other users or roles.

1. Create the **manager** role, grant **joe**'s permissions to **manager**, and allow **manager** to grant these permissions to others.

```
gaussdb=# CREATE ROLE manager PASSWORD 'xxxxxxxxxx';
gaussdb=# GRANT joe TO manager WITH ADMIN OPTION;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2. Create user **senior_manager** and grant **manager**'s permissions to it.

```
gaussdb=# CREATE ROLE senior_manager PASSWORD 'xxxxxxxxxx';
gaussdb=# GRANT manager TO senior_manager;
```

3. Revoke permissions and drop the user.

```
gaussdb=# REVOKE joe FROM manager;
gaussdb=# REVOKE manager FROM senior_manager;
gaussdb=# DROP USER manager;
gaussdb=# DROP DATABASE testdb;
```

Example 4: Grant the CMK or CEK permission to other users or roles.

1. Connect to an encrypted database.

```
-- Use the -C parameter to enable the encrypted database function.
gsql -p 57101 postgres -r -C
gaussdb=#  CREATE CLIENT MASTER KEY MyCMK1 WITH ( KEY_STORE = xxx , KEY_PATH = xxx ,
ALGORITHM = AES_256_CBC);
CREATE CLIENT MASTER KEY
gaussdb=# CREATE COLUMN ENCRYPTION KEY MyCEK1 WITH VALUES (CLIENT_MASTER_KEY =
MyCMK1, ALGORITHM = AEAD_AES_256_CBC_HMAC_SHA256);
CREATE COLUMN ENCRYPTION KEY
```

2. Create a role **newuser** and grant the key permission to **newuser**.

```
gaussdb=# CREATE USER newuser PASSWORD 'xxxxxxxxx';
CREATE ROLE
gaussdb=# GRANT ALL ON SCHEMA public TO newuser;
GRANT
gaussdb=# GRANT USAGE ON COLUMN_ENCRYPTION_KEY MyCEK1 to newuser;
GRANT
gaussdb=# GRANT USAGE ON CLIENT_MASTER_KEY MyCMK1 to newuser;
GRANT
```

3. Set the user to connect to a database and use a CEK to create an encrypted table.

```
gaussdb=# SET ROLE newuser PASSWORD 'xxxxxxxxx';
gaussdb=>  CREATE TABLE acltest1 (x int, x2 varchar(50) ENCRYPTED WITH
(COLUMN_ENCRYPTION_KEY = MyCEK1, ENCRYPTION_TYPE = DETERMINISTIC));
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'x' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=> SELECT has_cek_privilege('newuser', 'MyCEK1', 'USAGE');
 has_cek_privilege
-------------------
 t
(1 row)
```

4. Revoke permissions and drop the user.

```
gaussdb=> RESET ROLE;
gaussdb=# REVOKE USAGE ON COLUMN_ENCRYPTION_KEY MyCEK1 FROM newuser;
gaussdb=# REVOKE USAGE ON CLIENT_MASTER_KEY MyCMK1 FROM newuser;
gaussdb=# DROP TABLE newuser.acltest1;
gaussdb=# DROP COLUMN ENCRYPTION KEY MyCEK1;
gaussdb=# DROP CLIENT MASTER KEY MyCMK1;
gaussdb=# DROP SCHEMA IF EXISTS newuser CASCADE;
gaussdb=# REVOKE ALL ON SCHEMA public FROM newuser;
gaussdb=# DROP ROLE IF EXISTS newuser;
```

Example 5: Revoke permissions and delete roles and users.

```
gaussdb=# REVOKE gs_role_public_synonym_create FROM joe;
gaussdb=# REVOKE ALTER ON FUNCTION tpcds.fun1() FROM joe;
gaussdb=# REVOKE ALL PRIVILEGES ON tpcds.reason FROM joe;
gaussdb=# REVOKE ALL PRIVILEGES ON SCHEMA tpcds FROM joe;
gaussdb=# REVOKE ALL ON TABLESPACE tpcds_tbspc FROM joe;
gaussdb=# DROP TABLESPACE tpcds_tbspc;
gaussdb=# REVOKE USAGE,CREATE ON SCHEMA tpcds FROM tpcds_manager;
gaussdb=# DROP ROLE tpcds_manager;
gaussdb=# DROP ROLE senior_manager;
gaussdb=# DROP USER joe CASCADE;
gaussdb=# DROP TABLE tpcds.reason;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# DROP SYNONYM t1;
gaussdb=# DROP FUNCTION tpcds.fun1();
gaussdb=# DROP SCHEMA tpcds CASCADE;
```

Helpful Links

REVOKE and **ALTER DEFAULT PRIVILEGES**

1.13.14 I

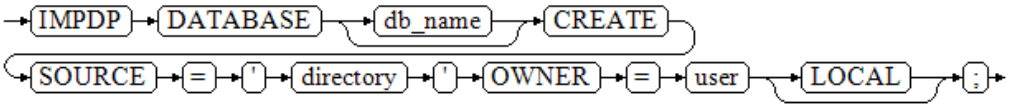
1.13.14.1 IMPDP DATABASE CREATE

Description

Specifies the preparation phase for importing a database.

Syntax

```
IMPDP DATABASE [db_name] CREATE SOURCE = 'directory' OWNER = user [LOCAL];
```



Parameters

- **db_name**

  Name of the new database after the import. If it is not specified, the original database name is retained after the import.
- **directory**

  Data source directory of the imported database.
- **user**

  Owner of the imported database.
- **LOCAL**

  If it is specified, data is imported to the original cluster. Otherwise, data is imported to a new cluster.

Examples

```
-- The IMPDP DATABASE CREATE syntax is used for fine-grained backup and restoration and is called by the backup and restoration tool. If you directly call the syntax, an error message may be displayed, indicating that the directory does not exist. Therefore, you are advised not to directly call the syntax.
gaussdb=# IMPDP DATABASE test CREATE SOURCE = '/data1/impdp/database' OWNER = admin;
```

1.13.14.2 IMPDP RECOVER

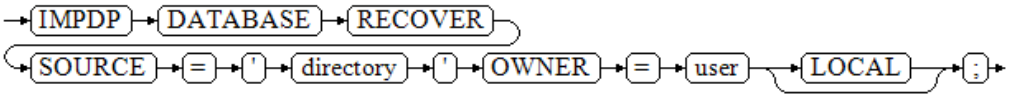
Description

Specifies the execution phase of importing a database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax

```
IMPDP DATABASE RECOVER SOURCE = 'directory' OWNER = user [LOCAL];
```



Parameters

- **directory**

  Data source directory of the imported database.
- **user**

  Owner of the imported database.
- **LOCAL**

  If it is specified, data is imported to the original cluster. Otherwise, data is imported to a new cluster.

Examples

```
-- The IMPDP DATABASE RECOVER syntax is used for fine-grained backup and restoration and is called by the backup and restoration tool. If you directly call the syntax, an error message may be displayed, indicating that the directory does not exist. Therefore, you are advised not to directly call the syntax.
gaussdb=# IMPDP DATABASE RECOVER SOURCE = '/data1/impdp/database' OWNER=admin;
```

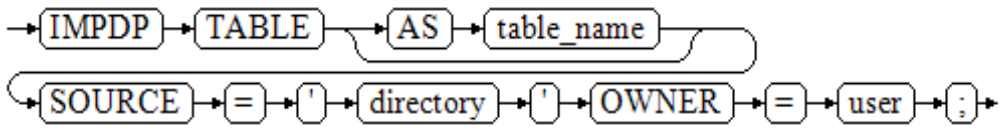
1.13.14.3 IMPDP TABLE

Description

Specifies the execution phase of importing a table.

Syntax

```
IMPDP TABLE [AS table_name] SOURCE = 'directory' OWNER = user;
```



Parameters

- **table_name**

  Name of the new table after the import. If it is not specified, the original table name is retained after the import.
- **directory**

  Data source directory of the imported table.
- **user**

  Owner of the imported table.

Examples

```
-- The IMPDP TABLE syntax is used for fine-grained backup and restoration and is called by the backup and restoration tool. If you directly call the syntax, an error message may be displayed, indicating that the
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
directory does not exist. Therefore, you are advised not to directly call the syntax.
gaussdb=# IMPDP TABLE SOURCE = '/data1/impdp/table0' OWNER=admin;
```

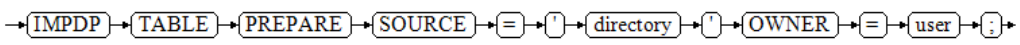
1.13.14.4 IMPDP TABLE PREPARE

Description

Specifies the preparation phase for importing a table.

Syntax

```
IMPDP TABLE PREPARE SOURCE = 'directory' OWNER = user;
```



Parameters

- **directory**

  Data source directory of the imported table.
- **user**

  Owner of the imported table.

Examples

```
-- The IMPDP TABLE PREPARE syntax is used for fine-grained backup and restoration and is called by the backup and restoration tool. If you directly call the syntax, an error message may be displayed, indicating that the directory does not exist. Therefore, you are advised not to directly call the syntax.
gaussdb=# IMPDP TABLE PREPARE SOURCE = '/data1/impdp/table0' OWNER=admin;
```

1.13.14.5 INSERT

Description

Inserts one or more rows of data into a table.

Precautions

- The owner of a table, users with the INSERT permission on the table, or users with the INSERT ANY TABLE permission can insert data into the table. System administrators have the permission to insert data into the table by default when separation of duties is disabled.
- Use of the RETURNING clause requires the SELECT permission on all columns mentioned in RETURNING.
- If ON DUPLICATE KEY UPDATE is used, you must have the INSERT and UPDATE permissions on the table and the SELECT permission on the columns of the UPDATE clause.
- If you use the query clause to insert rows from a query, you need to have the SELECT permission on any table or column used in the query.
- If you use the query clause to insert data from the dynamic data anonymization column, the inserted result is the anonymized value and cannot be restored.
- When you connect to a TD-compatible database and **td_compatible_truncation** is **on**, long strings will be automatically truncated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

If later INSERT statements (not involving foreign tables) insert long strings into columns of CHAR- and VARCHAR-typed columns in the target table, the system will truncate the long strings to ensure no strings exceed the maximum length defined in the target table.

NOTE

If multi-byte character data (such as Chinese characters) is inserted into a database with the character set byte encoding (such as SQL_ASCII and LATIN1), and the character data crosses the truncation position, the string is truncated based on its bytes instead of characters. Unexpected result will occur in tail after the truncation. If you want correct truncation result, you are advised to adopt encoding set such as UTF8, which has no character data crossing the truncation position.

Syntax

```
[ WITH [ RECURSIVE ] with_query [, ...] ]
INSERT [/*+ plan_hint */] [ IGNORE ] INTO table_name [ { [alias_name] [ ( column_name [, ...] ) ] } |
{ [partition_clause] [ AS alias ] [ ( column_name [, ...] ) ] } ]
   { DEFAULT VALUES
   | { VALUES | VALUE } {( { expression | DEFAULT } [, ...] ) }[, ...]
   | query }
   [ { ON DUPLICATE KEY UPDATE { NOTHING | { column_name = { expression | DEFAULT } } [, ...]
[ WHERE condition ] } }
   | { [ ON CONFLICT [ conflict_target ] conflict_action ] } ]
   [ RETURNING {* | {output_expression [ [ AS ] output_name ] }[, ...]} ];
```

Insert subqueries and views.

```
[ WITH [ RECURSIVE ] with_query [, ...] ]
INSERT [/*+ plan_hint */] INTO {subquery | view_name} [ AS alias ] [ ( column_name [, ...] ) ]
   { DEFAULT VALUES
   | { VALUES | VALUE } {( { expression | DEFAULT } [, ...] ) }[, ...]
   | query }
   [ RETURNING {* | {output_expression [ [ AS ] output_name ] }[, ...]} ];
```

- Detailed information of **subquery**:

  ```
  [ WITH [ RECURSIVE ] with_query [, ...] ]
  SELECT [/*+ plan_hint */] [ ALL ]
  { * | {expression [ [ AS ] output_name ]} [, ...] }
  [ into_option ]
  [ FROM from_item [, ...] ]
  [ WHERE condition ]
  [ [ START WITH condition ] CONNECT BY [NOCYCLE] condition [ ORDER SIBLINGS BY expression ] ]
  [ ORDER BY {expression [ [ ASC | DESC | USING operator ] | nlssort_expression_clause ] [ NULLS
  { FIRST | LAST } ]} [, ...] ]
  [ FETCH { FIRST | NEXT } [ count ] { ROW | ROWS } ONLY ]
  [ into_option ];
  ```

- The specified subquery source **from_item** is as follows:

  ```
  {[ ONLY ] {table_name | view_name} [ * ] [ partition_clause ] [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
  |( select ) [ AS ] alias [ ( column_alias [, ...] ) ]
  |with_query_name [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
  |from_item [ NATURAL ] join_type from_item [ ON join_condition | USING ( join_column [, ...] ) ]}
  ```

- Detailed information of **with_query**:

  ```
  with_query_name [ ( column_name [, ...] ) ] AS [ [ NOT ] MATERIALIZED ]
  ( {SELECT | VALUES | INSERT | UPDATE | DELETE} )
  ```

- Detailed information of partition_clause:

  ```
  PARTITION { ( partition_name ) | FOR ( partition_value [, ...] ) }
  ```

- Detailed information of **conflict_target**:

  ```
  ( { index_column_name | ( index_expression ) } [, ...] ) [ WHERE index_predicate ]
  ```

- Detailed information of **conflict_action**:

  ```
  DO NOTHING |
  DO UPDATE SET { column_name = { expression | DEFAULT } | ( column_name [, ...] ) = ( { expression |
  DEFAULT } [, ...] ) } [, ...] [ WHERE condition ]
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **WITH [RECURSIVE] with_query [, ...]**

  Specifies one or more subqueries that can be referenced by name in the main query, which is equivalent to a temporary table.

  If **RECURSIVE** is specified, a SELECT subquery can be referenced by name.

  with_query:

  – **with_query_name** specifies the name of the result set generated by a subquery. Such names can be used to access the result sets of subqueries in a query.

  – **column_name** specifies the column name displayed in the subquery result set.

  - Each subquery can be a SELECT, VALUES, INSERT, UPDATE, or DELETE statement.

  NOTICE

  If a subquery is a DML statement with a RETURNING statement, the number of command output records of the INSERT statement is determined by the subquery. Assume that table **T1** exists. Run the following statement:

  ```
  WITH CTE AS (INSERT INTO T1 VALUES(1,2) RETURNING *) INSERT INTO T1 SELECT * FROM CTE;
  ```

  The number of command output records of the preceding statement is determined by the following part instead of the entire statement. That is, the command output is "INSERT 0 1" instead of "INSERT 0 2".

  ```
  INSERT INTO T1 VALUES(1,2) RETURNING *
  ```

  – You can use MATERIALIZED or NOT MATERIALIZED to modify the CTE.

  – If **MATERIALIZED** is specified, the WITH query will be materialized, and a copy of the subquery result set is generated. The copy is directly queried at the reference point. Therefore, the WITH subquery cannot be jointly optimized with the SELECT statement trunk (for example, predicate pushdown and equivalence class transfer). In this scenario, you can use **NOT MATERIALIZED** for modification. If the WITH query can be executed as a subquery inline, the preceding optimization can be performed.

  – If the user does not explicitly declare the materialized attribute, comply with the following rules: If the CTE is referenced only once in the trunk statement to which it belongs and semantically supports inline execution, it will be rewritten as subquery inline execution. Otherwise, the materialized execution will be performed in CTE Scan mode.

  NOTE

  INSERT ON DUPLICATE KEY UPDATE does not support the WITH or WITH RECURSIVE clauses.

- **plan_hint** clause

  Follows the INSERT keyword in the /*+ */ format. It is used to optimize the plan of a INSERT statement block using hints. For details, see "Hint-based Optimization" in *SQL Optimization Guide*. In each statement, only the first **/*+** *plan_hint* ***/** comment block takes effect as a hint. Multiple hints can be written.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **IGNORE**

  When the INSERT statement uses the IGNORE keyword, some ERROR-level errors can be degraded to WARNING-level errors, and invalid values can be adjusted to the closest values based on error scenarios. GaussDB supports the following error codes for error downgrade:

  - Damage to the NOT NULL constraint
  - UNIQUE KEY conflict
  - No partition found for the inserted value
  - Unmatch between the inserted data and the specified partition
  - Multiple rows returned for a subquery
  - Oversized data
  - Time function overflow
  - Division by 0
  - Incorrect value

  NOTE

  1. INSERT IGNORE is supported only when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** set to **'5.7'**, and **b_format_dev_version** set to **'s1'**.
  2. INSERT IGNORE does not support encrypted tables and foreign tables.
  3. INSERT IGNORE does not support PRIMARY KEY constraints or UNIQUE constraints that take effect with a delay.
  4. INSERT IGNORE allows **NULL** values to damage NOT NULL constraints in terms of some data types, whose OIDs are TIMESTAMPOID, TIMESTAMPTZOID, TIMEOID, TIMETZOID, RELTIMEOID, INTERVALOID, TINTERVALOOID, SMALLDATETIMEOID, DATEOID, NAMEOID, POINTOID, PATHOID, POLYGONOID, CIRCLEOID, LSEGOID, BOXOID, JSONOID, JSONBOID, XMLOID, XMLTYPEOID, VARBITOID, NUMERICOID, CIDROID, INETOID, MACADDROID, NUMRANGEOID, INT8RANGEOID, INT4RANGEOID, TSRANGEOID, TSTZRANGEOID, DATERANGEOID, ABSTIMEOID, BOOLOID, INT8OID, INT4OID, INT2OID, INT1OID, FLOAT4OID, FLOAT8OID, CASHOID, UINT1OID, UINT2OID, UINT4OID, UINT8OID, BPCHAROID, VARCHAROID, NVARCHAR2OID, CHAROID, BYTEAOID, RAWOID, BLOBOID, CLOBOID, TEXTOID, and YEAROID. **Table 1-403** lists the zero values of each data type.
  5. When the default zero value is used for calculation, exercise caution when using IGNORE to ignore the NOT NULL constraint.
  6. INSERT IGNORE does not support tables with global secondary indexes.
  7. An error generated on a CN is degraded to the warning information while an error generated on a DN is not degraded to the warning information.

Table 1-403 Default zero values for the data type

OID of Each Data Type	Default Zero Value
INT8OID, INT4OID, INT2OID, INT1OID, UINT1OID, UINT2OID, UINT4OID, UINT8OID, FLOAT4OID, FLOAT8OID, NUMERICOID	0 or 0.00 (The number of 0s after the decimal point is specified by a parameter.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

OID of Each Data Type	Default Zero Value
BPCHAROID, VARCHAROID, CHAROID, BYTEAOID, RAWOID, BLOBOID, NVARCHAR2OID, CLOBOID, TEXTOID, VARBITOID, NAMEOID	Empty string
NUMRANGEOID, INT8RANGEOID, INT4RANGEOID, TSRANGEOID, TSTZRANGEOID, DATERANGEOID	empty
TIMEOID	time '00:00:00'
TIMETZOID	timetz '00:00:00'
INTERVALOID	interval '00:00:00'
TINTERVALOOID	tinterval(abstime '1970-01-01 00:00:00', abstime '1970-01-01 00:00:00')
SMALLDATETIMEOID	smalldatetime '1970-01-01 00:00:00'
ABSTIMEOID	abstime '1970-01-01 00:00:00'
RELTIMEOID	reltime '00:00:00'
TIMESTAMPOID, TIMESTAMPTZOID	1970-01-01 00:00:00
DATEOID	1970-01-01
YEAROID	0000
POINTOID	(0,0)
PATHOID, POLYGONOID	((0,0))
CIRCLEOID	<(0,0),0>
LSEGOID	[(0,0),(0,0)]
BOXOID	(0,0),(0,0)
JSONOID, JSONBOID, XMLOID	'null'
XMLTYPEOID	'<null/>'
CIDROID	0.0.0.0/32
INETOID	0.0.0.0
MACADDROID	00:00:00:00:00:00
BOOLOID	f
CASHOID	$0.00

- **table_name**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the name of the target table where data will be inserted.

Value range: existing table names.

NOTE

You can use database links to perform operations on remote tables. For details, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.

- **subquery**

  The inserted object can be a subquery. When a subquery is inserted, the subquery is regarded as a temporary view. The **CHECK OPTION** option can be added to the end of the subquery.

  If a subquery contains only one table, data is inserted into the table. If a subquery contains multiple tables or has nested relationships, check whether a key-preserved table exists to determine whether data can be inserted. For details about key-preserved tables and **WITH CHECK OPTION**, see **CREATE VIEW**.

- **view_name**

  Specifies the target view to be inserted.

  NOTE

  The restrictions on inserting views and subqueries are as follows:

  - The INSERT operation can be performed only on columns that directly reference user columns in the base table.
  - A subquery or view must contain at least one updatable column. For details about updatable columns, see **CREATE VIEW**.
  - The view or subquery must not contain the DISTINCT, GROUP BY, HAVING, LIMIT, or OFFSET clause at the top layer.
  - Views and subqueries that contain set operations (UNION, INTERSECT, EXCEPT, and MINUS) at the top layer are not supported.
  - Views and subqueries whose target lists contain aggregate functions, window functions, or return set functions (such as array_agg, json_agg, and generate_series) are not supported.
  - Views with BEFORE or AFTER triggers but without INSTEAD OF triggers or INSTEAD rules are not supported.
  - ON DUPLICATE KEY UPDATE is not supported.
  - Table types supported in views and subqueries include ordinary tables, temporary tables, global temporary tables, partitioned tables, subpartitioned tables, Ustore tables, and Astore tables.
  - For a multi-table join view or subquery, only one base table can be inserted at a time.
  - When INSERT joins a view or subquery, explicitly specified columns to be inserted or implicitly specified columns (columns specified during view or subquery creation) cannot reference columns of non-key-preserved tables; if the WITH CHECK OPTION clause is used, INSERT operations cannot be performed on join columns in join views or subqueries. For details about key-preserved tables, see **CREATE VIEW**.
  - The INSERT operation cannot be performed on the system view.

- **alias_name**

  Specifies the table alias when the INSERT statement is used without AS alias.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

1. When the INSERT statement is used without AS alias, the table alias cannot be a keyword (such as SELECT and VALUE) or an expression. The alias must comply with the identifier naming rule.
2. When the INSERT statement is used without AS alias, data cannot be inserted into the specified partition.

- **partition_clause**

  Inserts data to a specified partition.

  For details about the keywords, see **SELECT**.

  If the value of the VALUE clause is inconsistent with that of the specified partition, an exception is displayed.

- **column_name**

  Specifies the name of a column in a table.

  - The column name can be qualified with a subcolumn name or array index, if needed.
  - Each column not present in the column list will be filled with a default value, either its declared default value or **NULL** if there is none. Inserting data into only some columns of a composite type leaves the other columns **NULL**.
  - The target column names **column_name** can be listed in any order. If no list of column names is given at all, the default is all the columns of the table in their declared order.
  - The target columns are the first *N* column names, if there are only *N* columns supplied by the VALUE clause or query.
  - The values supplied by the VALUE clause or query are joined with the columns left-to-right.
  - Data can be inserted in the format of **table alias.column name**.

  Value range: existing column names.

- **expression**

  Specifies an expression or a value to assign to the corresponding column.

  - In the INSERT ON DUPLICATE KEY UPDATE statement, expression can be **VALUES(***column_name***)** or **EXCLUDED.***column_name*, indicating that the value of **column_name** corresponding to the conflict row is referenced. **VALUES(***column_name***)** can be nested in a compound expression, for example, **VALUES(***column_name***)+1**, **VALUES(***column_name***) +VALUES(***column_name***)**, and **function_name(VALUES(***column_name***))**.

    NOTE

    - **VALUES(***column_name***)** can be used only in the ON DUPLICATE KEY UPDATE clause.
    - **VALUES(***column_name***)** cannot be used in the IN or NOT IN expressions.
  - If single quotation marks (') are inserted into a column, they must be escaped by using an additional pair of single quotation marks.
  - If the expression for any column is not of the correct data type, automatic type conversion will be attempted. If the attempt fails, data insertion fails, and the system returns an error message.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **DEFAULT**

  Specifies the default value of a column. The value is **NULL** if there is no default value.

- **query**

  Specifies a query statement (SELECT statement) that uses the query result as the inserted data.

- **RETURNING**

  Returns the inserted rows. The syntax of the RETURNING list is identical to that of the output list of SELECT.

- **output_expression**

  Specifies an expression used to calculate the output result of the INSERT statement after each row is inserted.

  Value range: The expression can use any column in the table. You can use the asterisk (*) to return all columns of the inserted row.

- **output_name**

  Specifies a name to use for a returned column.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **ON DUPLICATE KEY UPDATE**

  For a table with a UNIQUE constraint (UNIQUE INDEX or PRIMARY KEY), if the inserted data violates the UNIQUE constraint, the UPDATE clause is executed to update the conflicting rows. If **NOTHING** is specified for the UPDATE clause, no operation will be performed. You can use **EXCLUDED.** or **VALUES()** to select the column corresponding to the source data.

  For a table without a unique constraint, only insert is performed.

  - The INSERT ... ON DUPLICATE KEY UPDATE statement supports STATEMENT triggers but does not support ROW triggers. Triggers are determined by the executed statement:
    - Executing **UPDATE NOTHING** will trigger the BEFORE INSERT and AFTER INSERT triggers.
    - Executing **UPDATE** will trigger the BEFORE INSERT, AFTER INSERT, BEFORE UPDATE, and AFTER UPDATE triggers.
  - Deferrable UNIQUE constraints or primary keys are not supported.
  - If a table has multiple unique constraints and the inserted data violates multiple unique constraints, only the first row that has a conflict is updated. (The check sequence is closely related to index maintenance. Generally, the conflict check is performed on the index that is created first.)
  - Primary keys and unique index columns cannot be updated.
  - Foreign tables are not supported.
  - The WHERE clause and expression of UPDATE cannot contain sublinks.

- **ON CONFLICT [conflict_target] conflict_action**

  Specifies whether to continue an INSERT operation when a unique or exclusive constraint conflict occurs during the operation. If no conflict occurs, the INSERT operation is performed. When a conflict occurs, if **ON CONFLICT**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

is followed by **DO NOTHING**, the current INSERT operation is abandoned; if **ON CONFLICT** is followed by **DO UPDATE SET**, the UPDATE operation is performed.

- **conflict_target**
  - **index_column_name**: Name of the index column.
  - **index_expression**: Similar to **index_column_name**, it is an expression index based on one or more columns of the table and can be used for fast access to a variation of the basic data.
  - **index_predicate**: condition expression that specifies the range of a partially unique index.
- **conflict_action**
  - **expression**: expression or value to be assigned to the corresponding column.
  - **DEFAULT**: The corresponding column will be filled with the default value.

NOTE

The INSERT... ON CONFLICT syntax has the following restrictions:

- Only PG compatibility is supported.
- The [COLLATE collation] [opclass] syntax is not supported.
- The ON CONSTRAINT constraint_name syntax is not supported.
- Users must have the UPDATE permission before using DO UPDATE SET.
- Nesting in the conflict_target, UPDATE SET, or UPDATE SET clause is not supported.
- Column-format tables, foreign tables, STREAM objects, and CONTVIEW objects are not supported.
- Distribution keys cannot be updated.
- In distributed mode, if the execution plan is **stream**, **light proxy**, or **fqs**, multiple data records with the same unique key cannot be inserted into the same statement at a time. If the execution plan is **PGXC**, the preceding operation is allowed.
- GSIs are not supported.
- Database links are not supported.
- Subqueries cannot be inserted.
- Only STATEMENT triggers are supported. ROW triggers are not supported.

Examples

- **Insert a data record.**

  Example:

```
-- Create a table.
gaussdb=# CREATE TABLE test_t1(col1 INT,col2 VARCHAR);

-- Insert data.
gaussdb=# INSERT INTO test_t1 (col1, col2) VALUES (1,'AB');

-- Add values to some columns of the table.
gaussdb=# INSERT INTO test_t1 (col1) VALUES (2);

-- There is no parenthesis on the left of the VALUES keyword. Values must be added to all fields in the parenthesis on the right according to the table definition sequence.
gaussdb=# INSERT INTO test_t1 VALUES (3,'AC');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Query the table.
gaussdb=# SELECT * FROM test_t1;
 col1 | col2
------+------
    1 | AB
    2 |
    3 | AC
(3 rows)
-- Drop.
gaussdb=# DROP TABLE test_t1;
```

- **Insert multiple data records.**

  Example:

```
-- Create a table.
gaussdb=# CREATE TABLE test_t2(col1 INT,col2 VARCHAR);
gaussdb=# CREATE TABLE test_t3(col1 INT,col2 VARCHAR);

-- Insert multiple data records.
gaussdb=# INSERT INTO test_t2 (col1, col2) VALUES (10,'AA'),(20,'BB'),(30,'CC');

-- Query.
gaussdb=# SELECT * FROM test_t2;
 col1 | col2
------+------
   10 | AA
   20 | BB
   30 | CC
(3 rows)

-- Insert data in test_t2 into test_t3.
gaussdb=# INSERT INTO test_t3 SELECT * FROM test_t2;

-- Query.
gaussdb=# SELECT * FROM test_t3;
 col1 | col2
------+------
   10 | AA
   20 | BB
   30 | CC
(3 rows)

-- Drop.
gaussdb=# DROP TABLE test_t2;
DROP TABLE test_t3;
```

- **ON DUPLICATE KEY UPDATE**

  Example:

```
-- Create a table.
gaussdb=# CREATE TABLE test_t4 (id INT PRIMARY KEY, info VARCHAR(10));
gaussdb=# INSERT INTO test_t4 VALUES (1, 'AA'), (2,'BB'), (3, 'CC');

-- Use the ON DUPLICATE KEY UPDATE keyword.
gaussdb=# INSERT INTO test_t4 VALUES (3, 'DD'), (4, 'EE') ON DUPLICATE KEY UPDATE info =
VALUES(info);

-- Query.
gaussdb=# SELECT * FROM test_t4;
 id | info
----+------
  1 | AA
  2 | BB
  4 | EE
  3 | DD

-- Drop.
gaussdb=# DROP TABLE test_t4;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **ON CONFLICT**

  Example:

```
-- Create a PG-compatible database and a data table.
gaussdb=# CREATE DATABASE pg_db dbcompatibility = 'PG';
gaussdb=# \c pg_db
pg_db=# CREATE TABLE insert_onconflict_tb1(id INT, a INT PRIMARY KEY, b INT, c VARCHAR2(20));
pg_db=# INSERT INTO insert_onconflict_tb1 VALUES (1, 1, 2, 'aaa'), (2, 2, 0, 'zzz');

-- Query table information.
pg_db=# SELECT * FROM insert_onconflict_tb1;
 id | a | b |  c
----+---+---+-----
  1 | 1 | 2 | aaa
  2 | 2 | 0 | zzz
(2 rows)

-- Use the ON CONFLICT keyword.
pg_db=# INSERT INTO insert_onconflict_tb1 VALUES (1, 1, 5, 'ddd'), (2, 2, 1, 'yyy') ON CONFLICT (a)
DO UPDATE SET b=excluded.b, c=excluded.c;
pg_db=# INSERT INTO insert_onconflict_tb1 VALUES (1, 1, 4, 'ccc'), (2, 2, 3, 'zzz') ON CONFLICT (a)
DO NOTHING;

-- Query table information.
pg_db=# SELECT * FROM insert_onconflict_tb1;
 id | a | b |  c
----+---+---+-----
  1 | 1 | 5 | ddd
  2 | 2 | 1 | yyy
(2 rows)

-- Drop the data table and the PG-compatible database.
pg_db=# DROP TABLE insert_onconflict_tb1;
pg_db=# \c postgres
gaussdb=# DROP DATABASE pg_db;
```

- **INSERT IGNORE**

  Example 1: Damage to the NOT NULL constraint

```
-- Create a table.
gaussdb=# CREATE TABLE test_t5(f1 INT NOT NULL);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'f1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE

-- Use the IGNORE keyword.
gaussdb=# INSERT IGNORE INTO test_t5 VALUES(NULL);
WARNING:  The null value in column "f1" violates the not-null constraint
DETAIL:  Failing row contains (null).
INSERT 0 1

-- Query the table.
gaussdb=# SELECT * FROM test_t5;
 f1
----
  0
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE test_t5;
DROP TABLE
```

  Example 2: UNIQUE KEY conflict

```
-- Create a table.
gaussdb=# CREATE TABLE test_t6(f1 INT PRIMARY KEY);
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "test_t6_pkey" for table "test_t6"
CREATE TABLE

-- Insert data.
gaussdb=# INSERT INTO test_t6 VALUES(1);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
INSERT 0 1

-- Use the IGNORE keyword.
gaussdb=# INSERT IGNORE INTO test_t6 VALUES(1);
INSERT 0 0

-- Query the table.
gaussdb=# SELECT * FROM test_t6;
 f1
----
  1
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE test_t6;
DROP TABLE
```

Example 3: No partition found for the inserted value

```
-- Create a table.
gaussdb=# CREATE TABLE test_t7(f1 INT, f2 INT) PARTITION BY LIST(f1) (PARTITION p0 VALUES(1, 4,
7), PARTITION p1 VALUES (2, 5, 8));
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'f1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE

-- Use the IGNORE keyword.
gaussdb=# INSERT IGNORE INTO test_t7 VALUES(3, 5);
INSERT 0 0

-- Query the table.
gaussdb=# SELECT * FROM test_t7;
 f1 | f2
----+----
(0 rows)

-- Drop the table.
gaussdb=# DROP TABLE test_t7;
DROP TABLE
```

Example 4: Unmatch between the inserted data and the specified partition

```
-- Create a table.
gaussdb=# CREATE TABLE test_t8(f1 INT NOT NULL, f2 TEXT, f3 INT) PARTITION BY RANGE(f1)
(PARTITION p0 VALUES LESS THAN(5), PARTITION p1 VALUES LESS THAN(10), PARTITION p2
VALUES LESS THAN(15), PARTITION p3 VALUES LESS THAN(MAXVALUE));
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'f1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE

-- Use the IGNORE keyword.
gaussdb=# INSERT IGNORE INTO test_t8 PARTITION(p2) VALUES(20, 'Jan', 1);
INSERT 0 0

-- Query the table.
gaussdb=# SELECT * FROM test_t8;
 f1 | f2 | f3
----+----+----
(0 rows)

-- Drop the table.
gaussdb=# DROP TABLE test_t8;
DROP TABLE
```

Example 5: Multiple rows returned for a subquery

```
-- Create a table.
gaussdb=# CREATE TABLE test_t9(f1 INT, f2 INT);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'f1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Insert data.
gaussdb=# INSERT INTO test_t9 VALUES(1, 1), (2, 2), (3, 3);
INSERT 0 3

-- Use the IGNORE keyword.
gaussdb=# INSERT IGNORE INTO test_t9 VALUES((SELECT f1 FROM test_t9), 0);
WARNING:  more than one row returned by a subquery used as an expression
CONTEXT:  referenced column: f1
INSERT 0 1

-- Query the table.
gaussdb=# SELECT * FROM test_t9 WHERE f2 = 0;
 f1 | f2
----+----
    |  0
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE test_t9;
DROP TABLE
```

Example 6: Oversized data

```
-- Create a table.
gaussdb=# CREATE TABLE test_t10(f1 VARCHAR(5));
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'f1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE

-- Use the IGNORE keyword.
gaussdb=# INSERT IGNORE INTO test_t10 VALUES('aaaaaaaaa');
WARNING:  value too long for type character varying(5)
CONTEXT:  referenced column: f1
INSERT 0 1

-- Query the table.
gaussdb=# SELECT * FROM test_t10;
  f1
-------
 aaaaa
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE test_t10;
DROP TABLE
```

Example 7: Time function overflow

```
-- Create a table.
gaussdb=# CREATE TABLE test_t11(f1 DATETIME);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'f1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE

-- Use the IGNORE keyword.
gaussdb=# INSERT IGNORE INTO test_t11 VALUES(date_sub('2000-01-01', INTERVAL 2001 YEAR));
WARNING:  Datetime function: datetime field overflow
CONTEXT:  referenced column: f1
INSERT 0 1

-- Query the table.
gaussdb=# SELECT * FROM test_t11;
 f1
----

(1 row)

-- Drop the table.
gaussdb=# DROP TABLE test_t11;
DROP TABLE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example 8: Division by 0

```
-- Create a table.
gaussdb=# CREATE TABLE test_t12(f1 INT);
CREATE TABLE

-- Use the IGNORE keyword.
gaussdb=# INSERT IGNORE INTO test_t12 VALUES(1/0);
WARNING:  division by zero
CONTEXT:  referenced column: f1
INSERT 0 1

-- Query the table.
gaussdb=# SELECT * FROM test_t12;
 f1
----

(1 row)

-- Drop the table.
gaussdb=# DROP TABLE test_t12;
DROP TABLE
```

Example 9: Incorrect value

```
-- Create a table.
gaussdb=# CREATE TABLE test_t13(f0 INT, f1 FLOAT);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'f0' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE

-- Use the IGNORE keyword.
gaussdb=# INSERT IGNORE INTO test_t13 VALUES(1, '1.11aaa');
WARNING:  invalid input syntax for type real: "1.11aaa"
LINE 1: INSERT IGNORE INTO test_t13 VALUES(1, '1.11aaa');
                                              ^
CONTEXT:  referenced column: f1
INSERT 0 1

-- Query the table.
gaussdb=# SELECT * FROM  test_t13;
 f0 |  f1
----+------
  1 | 1.11
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE test_t13;
DROP TABLE
```

Example 10: Insert a data record using a table alias.

```
-- Create a table.
gaussdb=# create table tb1 (va int , vb int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'va' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE

 -- Use a table alias.
gaussdb=# insert into tb1 as tt(tt.va, tt.vb) values (1,2);
INSERT 0 1
gaussdb=# insert into tb1 tt(tt.va, tt.vb) values (3,4);
INSERT 0 1

-- Query the table.
gaussdb=# select * from tb1;
 va | vb
----+----
  1 |  2
  3 |  4
(2 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop the table.
gaussdb=# DROP TABLE tb1;
DROP TABLE
```

- **WITH [RECURSIVE] with_query [, ...]**

  Example:

```
-- Grade table.
gaussdb=# CREATE TABLE grade (
    sid INT,
    course VARCHAR(20),
    score FLOAT
);
-- Student table.
gaussdb=# CREATE TABLE student(
    sid INT PRIMARY KEY,
    class INT,
    name  VARCHAR(50),
    gender INT CHECK (gender = 0 or gender = 1)
);

-- Insert data.
gaussdb=# WITH student_sid AS ( INSERT INTO student ( sid, CLASS, NAME, gender ) VALUES ( 1, 1,
'scott', 1 ) RETURNING sid )
    INSERT INTO grade ( sid, course, score )
    VALUE ( ( SELECT sid FROM student_sid ), 'Mathematics', '96' ),
      ( ( SELECT sid FROM student_sid ), 'Chinese', '82' ),
      ( ( SELECT sid FROM student_sid ), 'English', '86' );

-- Query data.
gaussdb=# SELECT * FROM student;
 sid | class | name  | gender
-----+-------+-------+-----
   1 |     1 | scott |   1
(1 row)

gaussdb=# SELECT * FROM grade;
 sid | course  | score
-----+---------+-------
   1 | Mathematics    |    96
   1 | Chinese    |    82
   1 | English    |    86
(3 rows)

-- Drop.
gaussdb=# DROP TABLE student;
gaussdb=# DROP TABLE grade;
```

- **Insert data into a view or subquery.**

  Example 1: Insert a subquery.

```
-- Create a schema.
gaussdb=# CREATE SCHEMA ins_subqry;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = 'ins_subqry';
SET

-- Create tables.
gaussdb=# CREATE TABLE t1 (x1 int, y1 int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'x1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# CREATE TABLE t2 (x2 int PRIMARY KEY, y2 int);
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "t2_pkey" for table "t2"
CREATE TABLE

-- Insert data into t1 through a subquery.
gaussdb=# INSERT INTO (SELECT * FROM t1) VALUES (1, 1);
INSERT 0 1
gaussdb=# INSERT INTO (SELECT * FROM t1 WHERE y1 < 3) VALUES (5, 5);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
INSERT 0 1

-- Insert a subquery with CHECK OPTION specified.
gaussdb=# INSERT INTO (SELECT * FROM t1 WHERE y1 < 3 WITH CHECK OPTION) VALUES (5, 5);
ERROR:  New row violates WITH CHECK OPTION for view "__unnamed_subquery__"
DETAIL:  Failing row contains (5, 5).

-- Insert a subquery with READONLY specified.
gaussdb=# INSERT INTO (SELECT * FROM t1 WITH READ ONLY) VALUES (5, 5);
ERROR:  DML operation cannot be performed on a read-only view.

-- Insert a multi-table join subquery.
gaussdb=# INSERT INTO (SELECT * FROM t1, t2 WHERE x1 = x2) (x1, y1) VALUES (2, 2);
INSERT 0 1

-- Drop a schema.
gaussdb=# RESET CURRENT_SCHEMA;
RESET
gaussdb=# DROP SCHEMA ins_subqry CASCADE;
NOTICE:  drop cascades to 2 other objects
DETAIL:  drop cascades to table ins_subqry.t1
drop cascades to table ins_subqry.t2
DROP SCHEMA
```

Example 2: Insert a view.

```
-- Create a schema.
gaussdb=# CREATE SCHEMA ins_view;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = 'ins_view';
SET

-- Create tables.
gaussdb=# CREATE TABLE t1 (x1 int, y1 int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'x1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# CREATE TABLE t2 (x2 int PRIMARY KEY, y2 int);
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "t2_pkey" for table "t2"
CREATE TABLE

-- Create a single table view.
gaussdb=# CREATE VIEW v_ins1 AS SELECT * FROM t1;
CREATE VIEW
gaussdb=# CREATE VIEW v_ins2 AS SELECT * FROM t1 WHERE y1 < 3;
CREATE VIEW
gaussdb=# CREATE VIEW v_ins2_wco AS SELECT * FROM t1 WHERE y1 < 3 WITH CHECK OPTION;
CREATE VIEW
gaussdb=# CREATE VIEW v_ins_read AS SELECT * FROM t1 WITH READ ONLY;
CREATE VIEW

-- Insert data into t1 through a view.
gaussdb=# INSERT INTO v_ins1 VALUES (1, 1);
INSERT 0 1
gaussdb=# INSERT INTO v_ins2 VALUES (5, 5);
INSERT 0 1
gaussdb=# INSERT INTO v_ins2_wco VALUES (5, 5);
ERROR:  new row violates WITH CHECK OPTION for view "v_ins2_wco"
DETAIL:  Failing row contains (5, 5).

gaussdb=# INSERT INTO v_ins_read VALUES (5, 5);
ERROR:  DML operation cannot be performed on a read-only view.

-- Create a multi-table view.
gaussdb=# CREATE VIEW vv_ins AS SELECT * FROM t1, t2 WHERE x1 = x2;
CREATE VIEW

-- Insert data into t1 through a view.
gaussdb=# INSERT INTO vv_ins (x1, y1) VALUES (2, 2);
INSERT 0 1
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop a schema.
gaussdb=# RESET CURRENT_SCHEMA;
RESET
gaussdb=# DROP SCHEMA ins_view CASCADE;
NOTICE:  drop cascades to 7 other objects
DETAIL:  drop cascades to table ins_view.t1
drop cascades to table ins_view.t2
drop cascades to view ins_view.v_ins1
drop cascades to view ins_view.v_ins2
drop cascades to view ins_view.v_ins2_wco
drop cascades to view ins_view.v_ins_read
drop cascades to view ins_view.vv_ins
DROP SCHEMA
```

Suggestions

- VALUES

  When you run the INSERT statement to insert data in batches, you are advised to combine multiple records into one statement to improve data loading performance.

  Example:

  ```
  INSERT INTO sections VALUES (30, 'Administration', 31, 1900),(40, 'Development', 35, 2000), (50, 'Development' , 60 , 2001);
  ```

  If values of an INSERT statement are distributed on a DN, GaussDB can push the statement down to the corresponding DN for execution. Currently, VALUES only supports constants, simple expressions, and pushdown functions (**provolatile** in pg_proc is set to **'i'**). If a column in the table has a default value, the value must be a constant or a simple expression. Neither a single-value statement nor a multi-value statement can be pushed down to a single DN.

- Batch insertion index performance optimization

  If any of the following scenarios meet all the restriction conditions, you are advised to enable the GUC parameter **batch_insert_index_types** and set it to **"rcr_ubtree"** to improve the performance of inserting indexes in batches:

  Scenario 1: Use JDBC to connect to the database and use the Java batch data insertion API executeBatch to insert indexes in batches.

  Scenario 2: Run the **insert into t1 select * from t2** statement, where tables **t1** and **t2** have the same schema, and **t1** contains indexes. This statement can be used to insert indexes in batches to the **t1** table. In this scenario, the stream plan must be enabled (the **enable_fast_query_shipping** parameter is set to **off**). Indexes are inserted in batches only when table data is inserted in batches. (You can run the **explain verbose** command to check whether table data is inserted in batches. **Streaming** indicates that data is inserted in batches.) If these prerequisites are met, enable the batch index insertion function to insert indexes in batches.

  Scenario 3: Run the **COPY FROM** statement (for example, **COPY t1 FROM 'file.csv' delimiter ',' csv header**), where **t1** and **file.csv** have the same schema, and **t1** contains indexes. This statement can be used to insert indexes in batches to the **t1** table.

  Restriction 1: The inserted index is a UB-tree RCR index.

  Restriction 2: The inserted data does not increase. (If the inserted data increases, the cache is used, that is, the last page is used. In this case, the performance of batch index insertion is not improved.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Restriction 3: **enable_stream_operator** is set to **on**.

Restriction 4: Batch index insertion is not performed in an SMP environment.

NOTE

This feature affects performance in the following scenarios:

1. If the data to be inserted in batches increases (violating restriction 2), after batch insertion is disabled, the performance is good. In this scenario, enabling the batch insertion function will slightly deteriorate the performance.

2. During continuous batch insertion, the index gradually increases. As a result, the batch insertion performance deteriorates.

3. If the cache page hit ratio is too low or the running memory is too small due to scattered data inserted in batches, enabling the batch insertion function will greatly deteriorate the performance.

If the performance deteriorates greatly, you are advised to set the GUC parameter **batch_insert_index_types** to **""** to disable the function. Then, the performance deterioration problem can be resolved.

If technical support needs to locate the cause, you can set the GUC parameter **logging_module** to **'on(BIINDEX)'** to enable the log printing function for batch index insertion. Perform batch insertion for several minutes until the **BIINDEX** character string is displayed in the log.

1.13.14.6 INSERT ALL

Description

Inserts one or multiple data records into multiple tables.

Precautions

- You must have the INSERT permission on a table to insert data into it. If a user is granted the INSERT ANY TABLE permission, the user has the USAGE permission on all schemas except system schemas and the INSERT permission on tables in these schemas.
- If you use the query clause to insert rows from a query, you need to have the SELECT permission on any table or column used in the query.
- Data cannot be written directly to a generated column. In the INSERT statement, values cannot be specified for generated columns, but the keyword DEFAULT can be specified.
- This syntax can be executed only in the ORA-compatible mode.

Syntax

```
INSERT [/*+ plan_hint */] [ ALL | FIRST ]
   { [ WHEN condition THEN ]
   { INTO table_name
   [ { [alias_name] [ ( column_name [, ...] ) ] } | { [partition_clause] [ AS alias ]
   [ ( column_name [, ...] ) ] } ]
   [ VALUES {( { expression | DEFAULT } [, ...] ) } ] }[, ...] }[, ...]
   { subquery };
```

Detailed information of **subquery**:

```
[ WITH [ RECURSIVE ] with_query [, ...] ]
SELECT [/*+ plan_hint */] [ ALL ]
{ * | {expression [ [ AS ] output_name ]} [, ...] }
[ into_option ]
[ FROM from_item [, ...] ]
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
[ WHERE condition ]
[ ORDER BY {expression [ [ ASC | DESC | USING operator ] | nlssort_expression_clause ] [ NULLS { FIRST | LAST } ]} [, ...] ]
[ FETCH { FIRST | NEXT } [ count ] { ROW | ROWS } ONLY ]
[ into_option ];
```

- Detailed information of **with_query**:

  ```
  with_query_name [ ( column_name [, ...] ) ] AS [ [ NOT ] MATERIALIZED ]
  ( {select | values | insert | update | delete} )
  ```

- The specified subquery source **from_item** is as follows:

  ```
  {[ ONLY ] {table_name | view_name} [ * ] [ partition_clause ] [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
  |( select ) [ AS ] alias [ ( column_alias [, ...] ) ]
  |with_query_name [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
  |from_item [ NATURAL ] join_type from_item [ ON join_condition | USING ( join_column [, ...] ) ]}
  ```

  Detailed information of partition_clause:

  ```
  PARTITION { ( partition_name ) | FOR ( partition_value [, ...] ) } |
  SUBPARTITION { ( subpartition_name ) | FOR ( subpartition_value [, ...] ) }
  ```

Parameters

- plan_hint clause

  Follows the INSERT keyword in the **/*+ */** format. Only the syntax format is adapted, and specific functions are not supported.

- **ALL/FIRST**

  The keyword cannot be omitted and can only be **ALL**, if a conditional clause (WHEN condition THEN) is not used. The keyword can be omitted and is **ALL** by default if a conditional clause is used. In this case, the meanings of **ALL** and **FIRST** are as follows:

  - **ALL**: Run through all branches of matching conditions and insert data into the corresponding table as long as the conditions are met.
  - **FIRST**: After finding the first branch that meets the conditions, insert the data into the table corresponding to the branch, and the check for remaining branches stops.

- **WHEN condition THEN**

  Specifies the condition judgment when inserting data into a table. The insertion is performed only when the condition is met. For conditions, refer to the columns in the subquery.

  Subqueries are not supported in the condition.

- **table_name**

  Specifies the name of the target table where data will be inserted.

  Value range: an existing table name.

- **alias_name**

  Specifies the table alias when the INSERT statement is used without AS alias.

  NOTE

  1. When the INTO clause is used without AS alias, the table alias cannot be a keyword (such as SELECT and VALUE) or an expression. The alias must comply with **Identifier Naming Conventions**.
  2. When the INTO clause is used without AS alias, the table alias cannot be in the INTO table_name alias_name(alias_name.col1, ...,alias_name.coln) VALUES(xxx); format.
  3. When the INTO clause is used without AS alias, data cannot be inserted into the specified partition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **partition_clause**

  Inserts data to a specified partition.

  ```
  PARTITION { ( partition_name ) | FOR ( partition_value [, ...] ) } |
  SUBPARTITION { ( subpartition_name ) | FOR ( subpartition_value [, ...] ) }
  ```

  For details about the keywords, see **SELECT**.

  If the value of the VALUE clause is inconsistent with that of the specified partition, an exception is displayed.

- **column_name**

  Specifies the name of a column in a table.

  - The column name can be qualified with a subcolumn name or array index, if needed.
  - Each column not present in the column list will be filled with a default value, either its declared default value or **NULL** if there is none. Inserting data into only some columns of a composite type leaves the other columns **NULL**.
  - The target column names **column_name** can be listed in any order. If no list of column names is given at all, the default is all the columns of the table in their declared order.
  - The target columns are the first *N* column names, if there are only *N* columns provided by the VALUE clause and QUERY.
  - The values provided by the VALUE clause and QUERY are joined with the corresponding columns from left to right in the table.

  Value range: name of an existing column.

- **VALUES**

  It cannot be followed by multiple rows.

  Subqueries are not supported.

- **expression**

  Specifies an expression or a value to assign to the corresponding column.

  - If single-quotation marks **(')** are inserted into a column, the single-quotation marks need to be used for escape.
  - If the expression for any column is not of the correct data type, automatic type conversion will be attempted. If the attempt fails, data insertion fails, and the system returns an error message.
  - Aggregate functions are not supported.
  - Subqueries are not supported.

- **DEFAULT**

  Specifies the default value of a column. The value is **NULL** if there is no default value.

- **subquery**

  Specifies a query statement (SELECT statement) that uses query results as the inserted data and cannot be omitted. If there are no subqueries, use SELECT * FROM DUAL.

  If a subquery contains a table alias, the alias cannot be referenced in the CONDITION and INTO clauses.

  For details, see **SELECT**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

- **Inserting data to multiple tables without conditions**

  Example:

```
-- Create an ORA-compatible database.
gaussdb=# CREATE DATABASE insertall_test_db DBCOMPATIBILITY 'ORA';

-- Switch to the ORA-compatible database.
gaussdb=# \c insertall_test_db;

-- Create tables.
insertall_test_db=# CREATE TABLE insert_all_test0(c1 INT, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test1(c1 INT, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test2(c1 INT, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test3(c1 INT, c2 VARCHAR2(20));

-- Construct data.
insertall_test_db=# INSERT INTO insert_all_test0 VALUES(1,'a'),(2,'b'),(3,'c'),(4,'d'),(5,'e');

-- Insert data to the tables.
insertall_test_db=# INSERT ALL
INTO insert_all_test1
INTO insert_all_test1 VALUES(1,'a')
INTO insert_all_test1(c1,c2)
INTO insert_all_test1(c1) VALUES(1)
INTO insert_all_test1(c2) VALUES('aaa')
INTO insert_all_test2(c2) VALUES(c2)
INTO insert_all_test2(c2) VALUES(c2||'*')
INTO insert_all_test3(c1,c2) VALUES(1,'a')
INTO insert_all_test3(c1,c2) VALUES(1,c2)
INTO insert_all_test3(c1,c2) VALUES(c1,c2)
INTO insert_all_test3(c2,c1) VALUES('***',666)
SELECT * FROM insert_all_test0;

-- Query data.
insertall_test_db=# SELECT * FROM insert_all_test1 ORDER BY c1,c2;
 c1 | c2
----+-----
  1 | a
  1 | a
  1 | a
  1 | a
  1 | a
  1 | a
  1 | a
  1 |
  1 |
  1 |
  1 |
  1 |
  2 | b
  2 | b
  3 | c
  3 | c
  4 | d
  4 | d
  5 | e
  5 | e
    | aaa
    | aaa
    | aaa
    | aaa
    | aaa
(25 rows)

insertall_test_db=# SELECT * FROM insert_all_test2 ORDER BY c1,c2;
 c1 | c2
----+----
    | a
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
   | a*
   | b
   | b*
   | c
   | c*
   | d
   | d*
   | e
   | e*
(10 rows)

insertall_test_db=# SELECT * FROM insert_all_test3 ORDER BY c1,c2;
 c1  | c2
-----+-----
   1 | a
   1 | a
   1 | a
   1 | a
   1 | a
   1 | a
   1 | a
   1 | b
   1 | c
   1 | d
   1 | e
   2 | b
   3 | c
   4 | d
   5 | e
 666 | ***
 666 | ***
 666 | ***
 666 | ***
 666 | ***
(20 rows)

-- Drop the tables.
insertall_test_db=# DROP TABLE insert_all_test0;
insertall_test_db=# DROP TABLE insert_all_test1;
insertall_test_db=# DROP TABLE insert_all_test2;
insertall_test_db=# DROP TABLE insert_all_test3;

insertall_test_db=# \c postgres;
gaussdb=# DROP DATABASE insertall_test_db;
```

- **Inserting data to multiple tables with conditions**

  Example 1: INSERT FIRST

```
-- Create an ORA-compatible database.
gaussdb=# CREATE DATABASE insertall_test_db DBCOMPATIBILITY 'ORA';

-- Switch to the ORA-compatible database.
gaussdb=# \c insertall_test_db;

-- Create tables.
insertall_test_db=# CREATE TABLE insert_all_test0(c1 int, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test1(c1 int, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test2(c1 int, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test3(c1 int, c2 VARCHAR2(20));

-- Construct data.
insertall_test_db=# INSERT INTO insert_all_test0 VALUES(1,'a'),(2,'b'),(3,'c'),(4,'d'),(5,'e');
-- Insert data to the tables.
insertall_test_db=# INSERT FIRST
WHEN c1 < 4 THEN
INTO insert_all_test1
WHEN c1 < 3 THEN
INTO insert_all_test2
ELSE
INTO insert_all_test3
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
SELECT * FROM insert_all_test0;

-- Query data.
insertall_test_db=# SELECT * FROM insert_all_test1 ORDER BY c1,c2;
 c1 | c2
----+----
  1 | a
  2 | b
  3 | c
(3 rows)

insertall_test_db=# SELECT * FROM insert_all_test2 ORDER BY c1,c2;
 c1 | c2
----+----
(0 rows)

insertall_test_db=# SELECT * FROM insert_all_test3 ORDER BY c1,c2;
 c1 | c2
----+----
  4 | d
  5 | e
(2 rows)

-- Drop the tables.
insertall_test_db=# DROP TABLE insert_all_test0;
insertall_test_db=# DROP TABLE insert_all_test1;
insertall_test_db=# DROP TABLE insert_all_test2;
insertall_test_db=# DROP TABLE insert_all_test3;

insertall_test_db=# \c postgres;
gaussdb=# DROP DATABASE insertall_test_db;
```

Example 2: INSERT ALL

```
-- Create an ORA-compatible database.
gaussdb=# CREATE DATABASE insertall_test_db DBCOMPATIBILITY 'ORA';

-- Switch to the ORA-compatible database.
gaussdb=# \c insertall_test_db;

-- Create tables.
insertall_test_db=# CREATE TABLE insert_all_test0(c1 int, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test1(c1 int, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test2(c1 int, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test3(c1 int, c2 VARCHAR2(20));

-- Construct data.
insertall_test_db=# INSERT INTO insert_all_test0 VALUES(1,'a'),(2,'b'),(3,'c'),(4,'d'),(5,'e');

-- Insert data to the tables with ALL.
insertall_test_db=# INSERT ALL
WHEN c1 < 4 THEN
INTO insert_all_test1
WHEN c1 < 3 THEN
INTO insert_all_test2
ELSE
INTO insert_all_test3
SELECT * FROM insert_all_test0;

-- Query data.
insertall_test_db=# SELECT * FROM insert_all_test1 ORDER BY c1,c2;
 c1 | c2
----+----
  1 | a
  2 | b
  3 | c
(3 rows)

insertall_test_db=# SELECT * FROM insert_all_test2 ORDER BY c1,c2;
 c1 | c2
----+----
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 1 | a
 2 | b
(2 rows)

insertall_test_db=# SELECT * FROM insert_all_test3 ORDER BY c1,c2;
 c1 | c2
----+----
 4 | d
 5 | e
(2 rows)

-- Drop data.
insertall_test_db=# TRUNCATE TABLE insert_all_test1;
insertall_test_db=# TRUNCATE TABLE insert_all_test2;
insertall_test_db=# TRUNCATE TABLE insert_all_test3;

-- Insert data to the tables without ALL.
insertall_test_db=# insert
WHEN c1 < 4 THEN
INTO insert_all_test1
WHEN c1 < 3 THEN
INTO insert_all_test2
ELSE
INTO insert_all_test3
SELECT * FROM insert_all_test0;

-- Query data.
insertall_test_db=# SELECT * FROM insert_all_test1 ORDER BY c1,c2;
 c1 | c2
----+----
 1 | a
 2 | b
 3 | c
(3 rows)

insertall_test_db=# SELECT * FROM insert_all_test2 ORDER BY c1,c2;
 c1 | c2
----+----
 1 | a
 2 | b
(2 rows)

insertall_test_db=# SELECT * FROM insert_all_test3 ORDER BY c1,c2;
 c1 | c2
----+----
 4 | d
 5 | e
(2 rows)

-- Drop the tables.
insertall_test_db=# DROP TABLE insert_all_test0;
insertall_test_db=# DROP TABLE insert_all_test1;
insertall_test_db=# DROP TABLE insert_all_test2;
insertall_test_db=# DROP TABLE insert_all_test3;

insertall_test_db=# \c postgres;
gaussdb=# DROP DATABASE insertall_test_db;
```

- **Support INSERT ALL in PL/SQL.**

  Example:

```
-- Create an ORA-compatible database.
gaussdb=# CREATE DATABASE insertall_test_db DBCOMPATIBILITY 'ORA';

-- Switch to the ORA-compatible database.
gaussdb=# \c insertall_test_db;

-- Create tables.
insertall_test_db=# CREATE TABLE insert_all_test0(c1 int, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test1(c1 int, c2 VARCHAR2(20));
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
insertall_test_db=# CREATE TABLE insert_all_test2(c1 int, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test3(c1 int, c2 VARCHAR2(20));

-- Construct data.
insertall_test_db=# INSERT INTO insert_all_test0 VALUES(1,'a'),(2,'b'),(3,'c'),(4,'d'),(5,'e');

-- Insert data to the tables using PL/SQL.
insertall_test_db=# DECLARE
var1 INT := 666;
var2 VARCHAR2(20) := '***';
BEGIN
INSERT ALL
INTO insert_all_test1 VALUES(var1,var2)
INTO insert_all_test1(c1) VALUES(var1)
INTO insert_all_test2(c2) VALUES(var2||'???')
INTO insert_all_test3(c1,c2) VALUES(c1,c2)
SELECT * FROM insert_all_test0;
END;
/

-- Query data.
insertall_test_db=# SELECT * FROM insert_all_test1 ORDER BY c1,c2;
 c1  | c2
-----+-----
 666 | ***
 666 | ***
 666 | ***
 666 | ***
 666 | ***
 666 |
 666 |
 666 |
 666 |
 666 |
(10 rows)

insertall_test_db=# SELECT * FROM insert_all_test2 ORDER BY c1,c2;
 c1 |   c2
----+--------
    | ***???
    | ***???
    | ***???
    | ***???
    | ***???
(5 rows)

insertall_test_db=# SELECT * FROM insert_all_test3 order by c1,c2;
 c1 | c2
----+----
  1 | a
  2 | b
  3 | c
  4 | d
  5 | e
(5 rows)

-- Drop the tables.
insertall_test_db=# DROP TABLE insert_all_test0;
insertall_test_db=# DROP TABLE insert_all_test1;
insertall_test_db=# DROP TABLE insert_all_test2;
insertall_test_db=# DROP TABLE insert_all_test3;

insertall_test_db=# \c postgres;
gaussdb=# DROP DATABASE insertall_test_db;
```

- **Exceptions**

  Example:

  ```
  -- Create an ORA-compatible database.
  gaussdb=# CREATE DATABASE insertall_test_db DBCOMPATIBILITY 'ORA';
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Switch to the ORA-compatible database.
gaussdb=# \c insertall_test_db;

-- Create tables.
insertall_test_db=# CREATE TABLE insert_all_test0(c1 int, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test1(c1 int, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test2(c1 int, c2 VARCHAR2(20));
insertall_test_db=# CREATE TABLE insert_all_test3(c1 int, c2 VARCHAR2(20));

-- Construct data.
insertall_test_db=# INSERT INTO insert_all_test0 VALUES(1,'a'),(2,'b'),(3,'c'),(4,'d'),(5,'e');

-- Aggregate functions are not supported in VALUES.
insertall_test_db=# INSERT ALL INTO insert_all_test1 VALUES(max(c1), 'a') SELECT * FROM
insert_all_test0;
ERROR:  An aggregate function cannot be used in VALUES.
LINE 1: INSERT ALL INTO insert_all_test1 VALUES(max(c1), 'a') SELECT

-- Multiple rows cannot be added after VALUES.
insertall_test_db=# INSERT ALL INTO insert_all_test1 VALUES(1,'a'),(2,'b') SELECT * FROM
insert_all_test0;
ERROR:  syntax error at or near ","
LINE 1: INSERT ALL INTO insert_all_test1 VALUES(1,'a'),(2,'b') SELEC...

-- The column in the WHEN condition does not exist in the subquery.
insertall_test_db=# INSERT ALL WHEN c1>1 THEN INTO insert_all_test1 VALUES(1,'a') SELECT c2
FROM insert_all_test0;
ERROR:  Column "c1" does not exist.
LINE 1: INSERT ALL WHEN c1>1 THEN INTO insert_all_test1 VALUES(1,'a'...
                    ^
HINT:  There is a column named "c1" in table "insert_all_test1", but it cannot be referenced from this
part of the query.

-- Aliases cannot be set for subquery tables.
insertall_test_db=# INSERT ALL INTO insert_all_test1 VALUES(t1.c1, 'a') SELECT * FROM
insert_all_test0 t1;
ERROR:  Missing FROM-clause entry for table "t1"
LINE 2: INSERT ALL INTO insert_all_test1 VALUES(t1.c1, 'a')

-- Data cannot be inserted into a view.
insertall_test_db=# CREATE VIEW v1 as SELECT * FROM insert_all_test1;
insertall_test_db=# INSERT ALL INTO v1 SELECT * FROM insert_all_test0;
ERROR:  Not allowed to insert into view.

-- Subqueries do not exist.
insertall_test_db=# INSERT ALL INTO insert_all_test1 VALUES(1, 'a');
ERROR:  syntax error at or near ";"
LINE 2: INSERT ALL INTO insert_all_test1 VALUES(1, 'a');

-- Drop the tables.
insertall_test_db=# DROP VIEW v1;
insertall_test_db=# DROP TABLE insert_all_test0;
insertall_test_db=# DROP TABLE insert_all_test1 CASCADE;
insertall_test_db=# DROP TABLE insert_all_test2;
insertall_test_db=# DROP TABLE insert_all_test3;

insertall_test_db=# \c postgres;
gaussdb=# DROP DATABASE insertall_test_db;
```

1.13.15 L

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.15.1 LOAD DATA

Description

Imports data from a file to a specified table in the database.

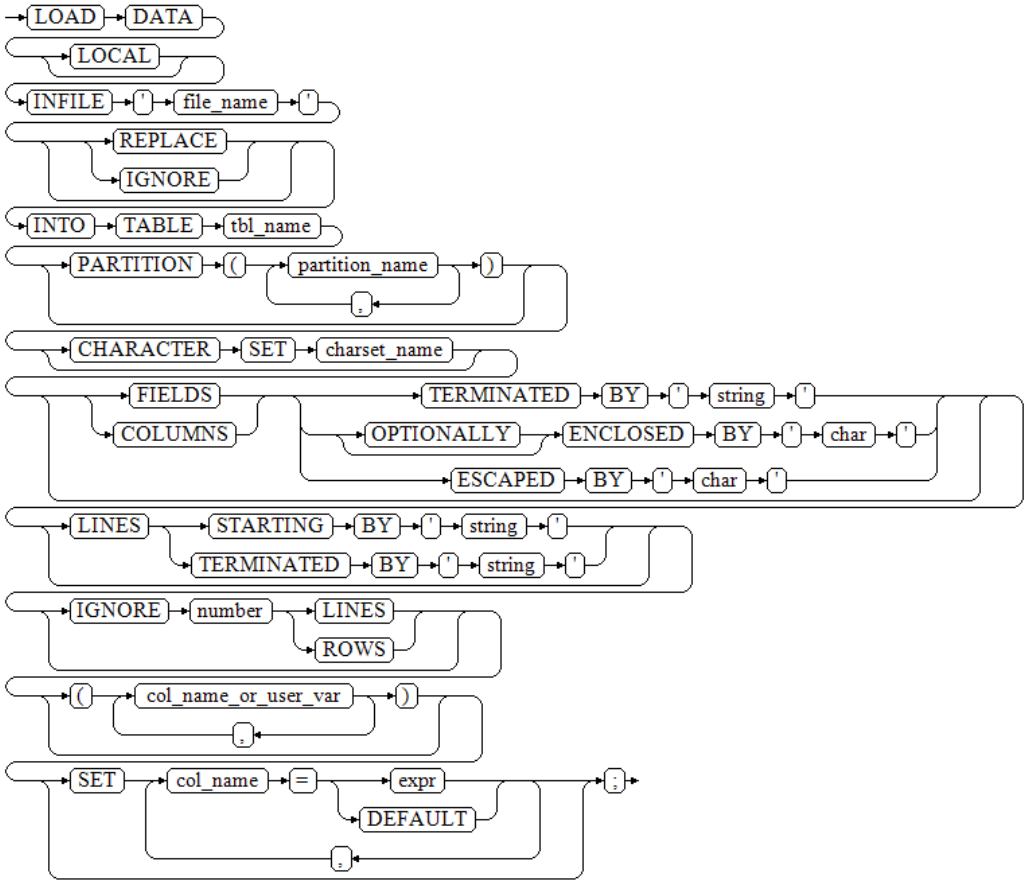
Precautions

- This syntax can be used only in the MySQL-compatible mode (**sql_compatibility = 'MYSQL'**) where **b_format_version** is **'5.7'** and **b_format_dev_version** is **'s2'**.
- In other compatibility modes or when the compatibility parameters (**b_format_version='5.7'** and **b_format_dev_version='s2'**) are not set, several statements will be returned. These statements are mainly used internally by gs_loader.
- **Native Syntax** is internally called only by gs_loader. You are advised to use it based on **Recommended Syntax** instead of using it directly.
- The GUC parameters related to the LOAD DATA syntax are the same as those of the COPY FROM syntax. For details, see **Precautions**.
- The LOAD DATA syntax requires the INSERT and DELETE permissions on tables.
- If the data written to a table by running **LOAD DATA** cannot be converted to the data type of the table, the import fails.
- LOAD DATA applies only to tables but not views.

Recommended Syntax

```
LOAD DATA [LOCAL] INFILE 'file_name'
    [REPLACE | IGNORE]
    INTO TABLE tbl_name
    [PARTITION (partition_name [, ...])]
    [CHARACTER SET charset_name]
    [{FIELDS | COLUMNS} {TERMINATED BY 'string' | [OPTIONALLY] ENCLOSED BY 'char' | ESCAPED BY
'char'}]
    [LINES {STARTING BY 'string' | TERMINATED BY 'string'}]
    [IGNORE number {LINES | ROWS}]
    [(col_name_or_user_var [, ...])]
    [SET {col_name={expr | DEFAULT}} [, ...]];
```

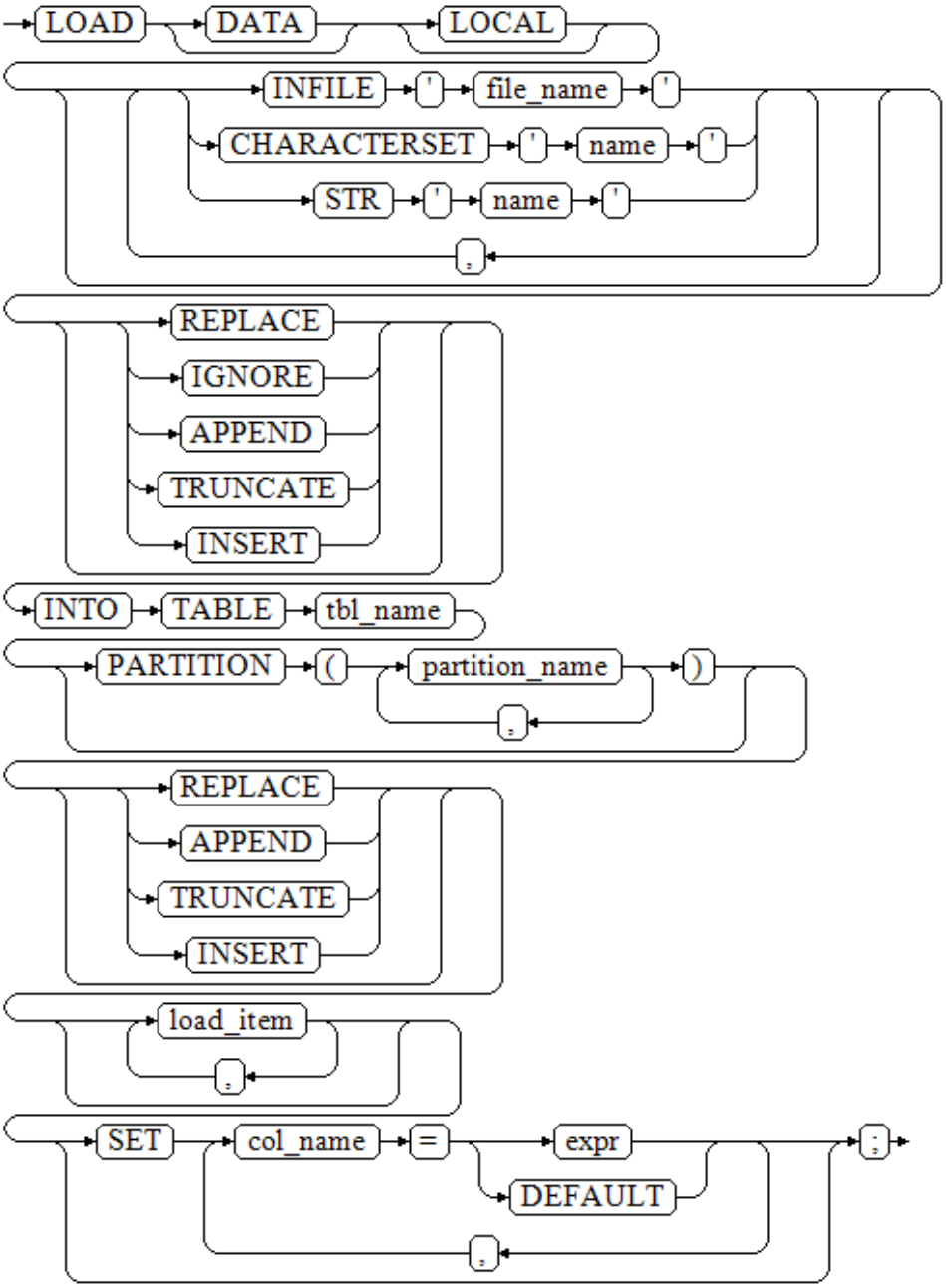
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Native Syntax

```
LOAD [DATA] [LOCAL]
   [{INFILE 'file_name' | CHARACTERSET 'name' | STR 'name' } [,...]]
   [REPLACE | IGNORE | APPEND | TRUNCATE | INSERT ]
   INTO TABLE tbl_name
   [PARTITION (partition_name [, ...])]
 [REPLACE | APPEND | TRUNCATE | INSERT ]
   [load_item [,...]]
   [SET {col_name={expr | DEFAULT}} [,...]];
```

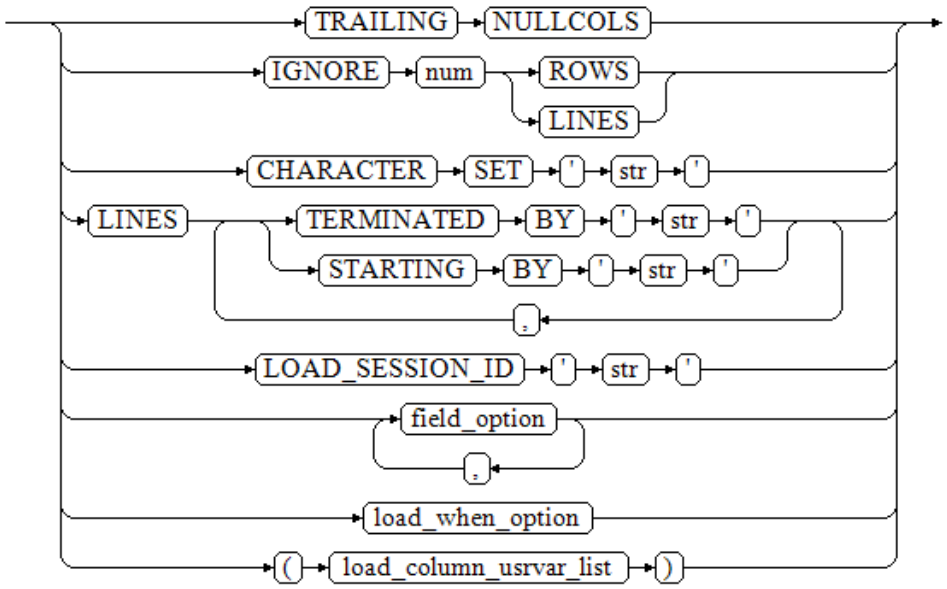
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



- **load_item** specifies the options for importing a file. The syntax is as follows:

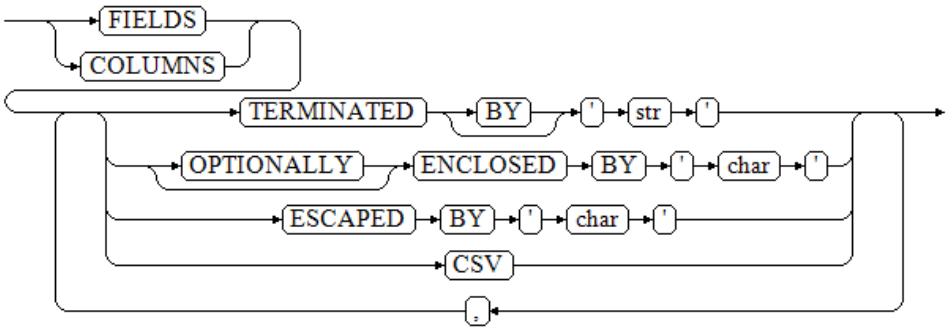
```
{
  TRAILING NULLCOLS | IGNORE num {ROWS | LINES} | CHARACTER SET 'str'|
   LINES {{TERMINATED BY 'str' | STARTING BY 'str'} [, ...]} |
   LOAD_SESSION_ID 'str' |
   {field_option [,...]} |
   load_when_option |
   ( load_column_usrvar_list )
}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



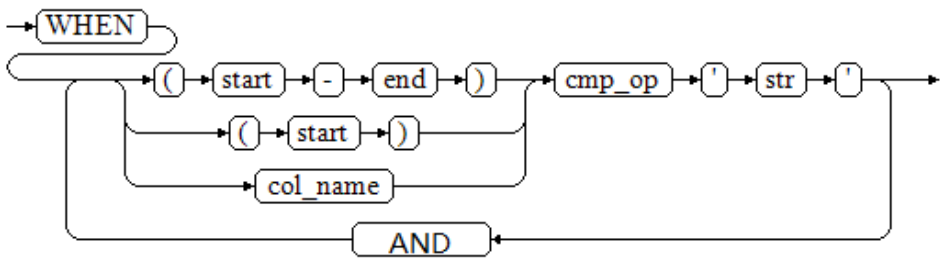
- **filed_option** is an option related to the format of the imported file. The syntax format is as follows:

```
{
 {FIELDS | COLUMNS}
 {{TERMINATED [BY] 'str' | [OPTIONALLY] ENCLOSED BY 'char' | ESCAPED BY 'char' | CSV } [, ...]}
}
```



- **load_when_option** indicates the specific filtering option used for importing files. The syntax format is as follows:

```
{
 WHEN
 {{{( start - end ) | (start) | col_name } cmp_op 'str' }[AND ...]}
}
```

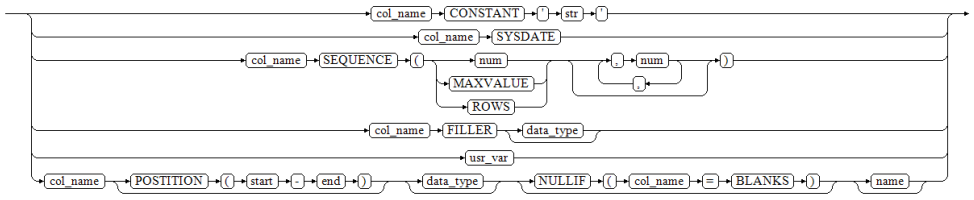


- **load_column_usrvar_list** indicates the syntax options for assigning values to columns, generating sequence values, and filtering data during data import. The syntax format is as follows:

```
{
 col_name CONSTANT 'str' | col_name SYSDATE |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 col_name SEQUENCE ( {num | MAXVALUE | ROWS} [ {, num} [, ...]] ) |
 col_name FILLER [data_type] | usr_var |
 col_name [ POSTITION ( start - end ) ] [ data_type ] [ NULLIF ( col_name = BLANKS )] [ name ]
}
```



NOTE

If **disable_unsupported_copy_load_options** is enabled, an error will be reported when options in **Native Syntax** are used.

Parameters

- **LOCAL**

  Specifies the location of the file to be imported.

  If **LOCAL** is not specified and **file_name** is a relative path, data is imported to the data directory by default.

  If the **LOCAL** parameter is specified, **file_name** must be set to an absolute path. If **file_name** is set to a relative path, data is imported to the location of the database binary file by default, that is, *$GAUSSHOME*/**bin/**.

- **REPLACE | IGNORE**

  Specifies an option for handling any data conflicts or errors during file import.

  - Specifying the **REPLACE** option

    If the imported data conflicts with existing data in the target table, the conflicting rows will be replaced. If the imported file contains fewer columns than the target table, the database will issue an alarm in loose mode or raise an error in strict mode.

  - Specifying the **IGNORE** option

    If the imported data conflicts with existing data in the target table, the conflicting rows will be skipped. If the imported file contains fewer columns than the target table, unspecified columns in the table will be filled with default values, and an alarm will be issued.

  - Specifying neither **REPLACE** nor **IGNORE**

    - If the LOCAL option is not specified, the database will raise an error upon data conflicts with existing rows in the target table. If the imported file contains fewer columns than the target table, the database will issue an alarm in loose mode or raise an error in strict mode.
    - When the **LOCAL** option is specified, the behavior mirrors that of the **IGNORE** option.

- **PARTITION**

  If the table to be imported is a partitioned table, this parameter specifies a partition. If the data is inconsistent with the specified partition range, an error is reported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **CHARACTER SET**

  Specifies the encoding format of a data file. The default value is the current client-side encoding format.

- **FIELDS | COLUMNS**
  - **TERMINATED BY**

    Specifies the delimiter between columns. The default value is **"\t"**.

    NOTE

    The specified newline character cannot be the same as the delimiter.

  - **[OPTIONALLY] ENCLOSED BY**

    Specifies the quotation mark character. The default value is **''**.

    The **OPTIONALLY** parameter is optional and has no practical effect.

    The quotation mark can only be a single character and cannot be a character string.

  - **ESCAPED BY**

    Specifies the escape character. The default value is **"\"**.

    NOTE

    - The escape character must be a single character, not a string.
    - When a non-empty escape character is specified, the database checks whether this character, together with the subsequent sequence, matches the defined escape sequence. If a match is found, the database converts them into the designated characters. Otherwise, the escape character is ignored, and data import proceeds. The supported escape sequences are controlled by the presence of the **new_escape_string** option in the GUC parameter **load_data_behavior_compat_options**.

- **LINES**
  - **STARTING BY**

    Specifies the style of the starting column in the data file to be imported.

  - **TERMINATED BY**

    Specifies the newline character style of the imported data file.

    NOTE

    The **new_escape_string** option in the GUC parameter **load_data_behavior_compat_options** controls whether to escape the specified row terminator.

- **IGNORE**

  Specifies that the first *number* rows of the data file are skipped during data import.

- **col_name_or_user_var**

  Specifies an optional list of columns to be copied.

  Value range: any columns. All columns will be copied if no column list is specified.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The parameter for specifying columns cannot be used to specify a column repeatedly.
- When columns are specified using the LOAD DATA syntax, **col_name_or_user_var** can only be specified as existing columns but not user variables.

- **SET**

  Specifies the column value, which can be an expression or **DEFAULT**.

  NOTE

  - The expression does not support column names.
  - If no implicit conversion exists between the expression result type and the type of the assigned column, an error is reported.

Examples

```
-- Prepare the environment.
gaussdb=# CREATE DATABASE test_load WITH DBCOMPATIBILITY = 'MYSQL';
gaussdb=# \c test_load
test_load=# SET b_format_version='5.7';
test_load=# SET b_format_dev_version='s2';

-- Prepare the files to be imported.
test_load=# CREATE TABLE load_data_tmp1(a int, b int);
test_load=# INSERT INTO load_data_tmp1 VALUES(1,1),(2,2),(3,3);
test_load=# \copy load_data_tmp1 to '/home/omm/load1.csv';
test_load=# CREATE TABLE load_data_tmp2(a int, b int);
test_load=# INSERT INTO load_data_tmp2 VALUES(1,2);
test_load=# \copy load_data_tmp2 to '/home/omm/load2.csv';
test_load=# CREATE TABLE load_data_tmp3(a int, b int);
test_load=# INSERT INTO load_data_tmp3 VALUES(4,4),(5,5);
test_load=# \copy load_data_tmp3 to '/home/omm/load3.csv';
test_load=# CREATE TABLE load_data_tmp4(a char(50));
test_load=# INSERT INTO load_data_tmp4 VALUES('"load test quote"'), ('\'load test single_quote\'');
test_load=# \copy load_data_tmp4 to '/home/omm/load4.csv';

-- Create a table.
test_load=# CREATE TABLE load_data_tbl1(load_col1 INT UNIQUE, load_col2 INT, load_col3 CHAR(10));

-- Insert a data record into the table.
test_load=# INSERT INTO load_data_tbl1 VALUES(0,0,'load0');

-- Copy data from the /home/omm/load1.csv file to the load_data_tbl table, specify a column name, and
set the value of the load_col3 column to load.
test_load=# LOAD DATA INFILE '/home/omm/load1.csv' INTO TABLE load_data_tbl1(load_col1, load_col2)
SET load_col3 = 'load';
-- The values imported to the load_col3 column later are all load.
test_load=# SELECT * FROM load_data_tbl1 ORDER BY load_col1;
load_col1 | load_col2 | load_col3
-----------+-----------+-----------
         0 |         0 | load0
         1 |         1 | load
         2 |         2 | load
         3 |         3 | load
(4 rows)

-- Copy data from the /home/omm/load2.csv file to the load_data_tbl table and specify IGNORE to
ignore conflicts.
test_load=# LOAD DATA INFILE '/home/omm/load2.csv' IGNORE INTO TABLE load_data_tbl1;
-- Data in the load_data_tbl1 table remains unchanged, and conflicting data is skipped.
test_load=# SELECT * FROM load_data_tbl1 ORDER BY load_col1;
load_col1 | load_col2 | load_col3
-----------+-----------+-----------
         0 |         0 | load0
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
       1 |         1 | load
       2 |         2 | load
       3 |         3 | load
(4 rows)

-- Create a partitioned table.
test_load=# CREATE TABLE load_data_tbl2
(
    load_col_col1 INT,
    load_col_col2 INT
) PARTITION BY RANGE (load_col_col2)
(
    PARTITION load_p1 VALUES LESS THAN(3),
    PARTITION load_p2 VALUES LESS THAN(9),
    PARTITION load_p3 VALUES LESS THAN(MAXVALUE)
);

-- Copy data from the /home/omm/load3.csv file to the load_data_tbl2 table and specify a partition.
test_load=# LOAD DATA INFILE '/home/omm/load3.csv' INTO TABLE load_data_tbl2 PARTITION (load_p2);
-- Import data to the specified partition in the load_data_tbl2 table.
test_load=# SELECT * FROM load_data_tbl2;
 load_col_col1 | load_col_col2
---------------+---------------
           4 |             4
           5 |             5
(2 rows)

-- Create a table.
test_load=# CREATE TABLE load_data_tbl3(load_col_col1 CHAR(30));

-- Copy data from the /home/omm/load4.csv file to the load_data_tbl3 table and specify FIELDS
ENCLOSED BY.
test_load=# LOAD DATA INFILE '/home/omm/load4.csv' INTO TABLE load_data_tbl3 FIELDS ENCLOSED BY
'"';
-- The double quotation marks ("") of "load test quote" are removed, and the single quotation marks of
'load test single_quote' are retained.
test_load=# SELECT * from load_data_tbl3;
        load_col_col1
--------------------------------
 load test quote
 'load test single_quote'
(2 rows)
-- Drop tables.
test_load=# DROP TABLE load_data_tmp1;
test_load=# DROP TABLE load_data_tmp2;
test_load=# DROP TABLE load_data_tmp3;
test_load=# DROP TABLE load_data_tmp4;
test_load=# DROP TABLE load_data_tbl1;
test_load=# DROP TABLE load_data_tbl2;
test_load=# DROP TABLE load_data_tbl3;
```

1.13.15.2 LOCK

Description

LOCK TABLE obtains a table-level lock.

GaussDB always tries to select the lock mode with minimum restrictions when automatically requesting a lock for a statement referenced by a table. Use LOCK if users need a more strict lock mode. For example, suppose an application runs a transaction at the **READ COMMITTED** isolation level and needs to ensure that data in a table remains stable in the duration of the transaction. To achieve this, you could obtain **SHARE** lock mode over the table before the query. This prevents data from being concurrently modified, ensuring that subsequent queries can read committed and persistent data. It is because the **SHARE** lock mode conflicts with

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

the **ROW EXCLUSIVE** lock acquired by writers, and your **LOCK TABLE name IN SHARE MODE** statement will wait until any concurrent holders of **ROW EXCLUSIVE** mode locks commit or roll back. Therefore, once you obtain the lock, there are no uncommitted writes outstanding; furthermore none can begin until you release the lock.

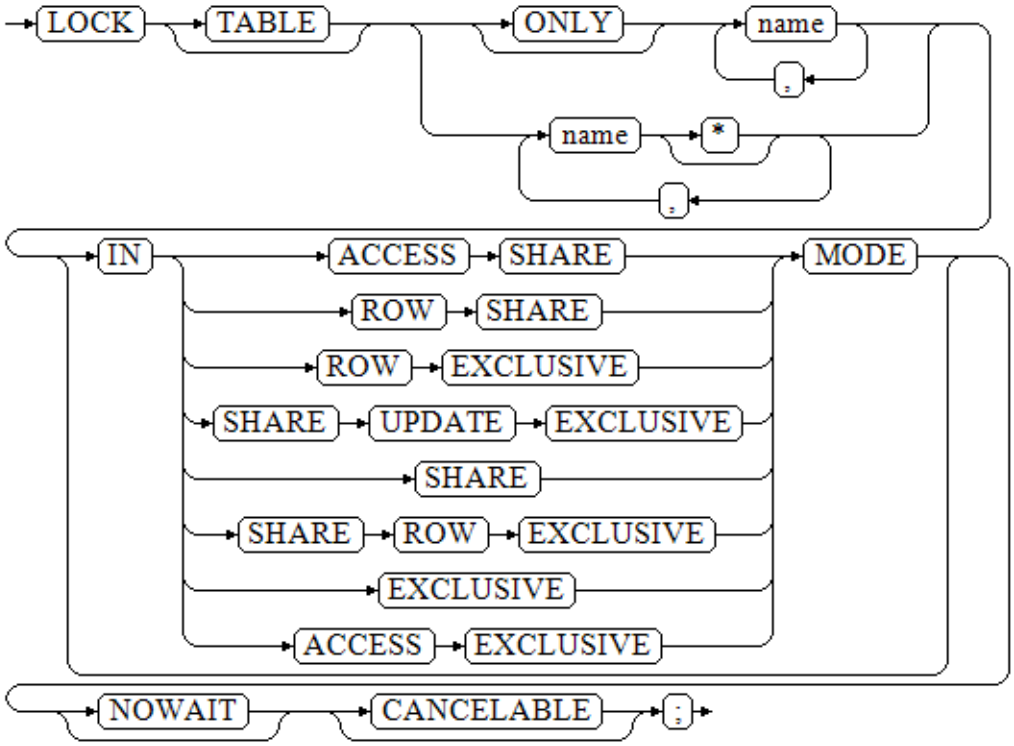
Precautions

- LOCK TABLE is useless outside a transaction block: the lock would remain held only to the completion of the statement. If LOCK TABLE is out of any transaction block, an error is reported.
- If no lock mode is specified, then **ACCESS EXCLUSIVE**, the most restrictive mode, is used.
- LOCK TABLE ... **IN ACCESS SHARE MODE** requires the SELECT permission on the target table. All other forms of LOCK require table-level UPDATE and/or the DELETE permission.
- There is no UNLOCK TABLE statement. Locks are always released at transaction end.
- LOCK TABLE only deals with table-level locks, and so the mode names involving **ROW** are all misnomers. These mode names should generally be read as indicating the intention of the user to acquire row-level locks within the locked table. Also, **ROW EXCLUSIVE** mode is a shareable table lock. Note that all lock modes have the same semantics as long as LOCK TABLE is involved. The only difference lies in whether locks conflict with each other. For details about the rules, see **Table 1-404**.
- If the xc_maintenance_mode parameter is not enabled, an error is reported when an ACCESS EXCLUSIVE lock is applied for a system catalog.
- Only the redistribution tool can use the automatic CANCEL service API.

Syntax

```
LOCK [ TABLE ] {[ ONLY ] name [, ...]| {name [ * ]} [, ...]}
    [ IN {ACCESS SHARE | ROW SHARE | ROW EXCLUSIVE | SHARE UPDATE EXCLUSIVE | SHARE | SHARE
ROW EXCLUSIVE | EXCLUSIVE | ACCESS EXCLUSIVE} MODE ]
    [ NOWAIT ][CANCELABLE];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

Table 1-404 Lock mode conflicts

Requested Lock Mode/ Current Lock Mode	ACCESS SHARE	ROW SHARE	ROW EXCLUSIVE	SHARE UPDATE EXCLUSIVE	SHARE	SHARE ROW EXCLUSIVE	EXCLUSIVE	ACCESS EXCLUSIVE
ACCESS SHARE	-	-	-	-	-	-	-	X
ROW SHARE	-	-	-	-	-	-	X	X
ROW EXCLUSIVE	-	-	-	-	X	X	X	X
SHARE UPDATE EXCLUSIVE	-	-	-	X	X	X	X	X

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Requested Lock Mode/ Current Lock Mode	ACCESS SHARE	ROW SHARE	ROW EXCLUSIVE	SHARE UPDATE EXCLUSIVE	SHARE	SHARE ROW EXCLUSIVE	EXCLUSIVE	ACCESS EXCLUSIVE
SHARE	-	-	X	X	-	X	X	X
SHARE ROW EXCLUSIVE	-	-	X	X	X	X	X	X
EXCLUSIVE	-	X	X	X	X	X	X	X
ACCESS EXCLUSIVE	X	X	X	X	X	X	X	X

LOCK parameters are as follows:

- **name**

  Specifies the name (optionally schema-qualified) of an existing table to lock.

  Tables are locked one-by-one in the order specified in the LOCK TABLE statement.

  Value range: existing table names.

  NOTE

  You can use database links to perform operations on remote tables. For details, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.

- **ONLY**

  If **ONLY** is specified, only that table is locked. If it is not specified, the table and all its sub-tables are locked.

- **ACCESS SHARE**

  Allows only read operations on a table. In general, any SQL statements that only read a table and do not modify it will acquire this lock mode. The SELECT statement acquires an **ACCESS SHARE** lock on referenced tables.

- **ROW SHARE**

  Allows concurrent read of a table but does not allow any other operations on the table.

  SELECT FOR UPDATE and **SELECT FOR SHARE** automatically acquire the **ROW SHARE** lock on the target table and add the **ACCESS SHARE** lock to other referenced tables except **FOR SHARE** and **FOR UPDATE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

For a partitioned table, **SELECT FOR SHARE** obtains the **ROW EXCLUSIVE** lock of the partition object on the DN for concurrency control.

- **ROW EXCLUSIVE**

  Allows concurrent read of a table but does not allow modification of data in the table. **ROW EXCLUSIVE** is the same as **ROW SHARE**. **UPDATE**, **DELETE**, and **INSERT** automatically acquire this lock on the target table and add the **ACCESS SHARE** lock to other referenced tables. Generally, all statements that modify table data acquire the **ROW EXCLUSIVE** lock for tables.

- **SHARE UPDATE EXCLUSIVE**

  Protects a table against concurrent schema changes and **VACUUM** runs.

  The VACUUM (without FULL), ANALYZE, and CREATE INDEX CONCURRENTLY statements automatically request this lock.

- **SHARE**

  Allows concurrent queries of a table but does not allow modification of the table.

  The CREATE INDEX (without CONCURRENTLY) statement automatically requests this lock.

- **SHARE ROW EXCLUSIVE**

  Protects a table against concurrent data changes, and is self-exclusive so that only one session can hold it at a time.

  No SQL statements automatically acquire this lock mode.

- **EXCLUSIVE**

  Allows concurrent queries of the target table but does not allow any other operations.

  This mode allows only concurrent ACCESS SHARE locks; that is, only reads from the table can proceed concurrently with a transaction holding this lock mode.

  No SQL statements automatically acquire this lock mode on user tables. However, it will be acquired on some system catalogs in case of some operations.

- **ACCESS EXCLUSIVE**

  Guarantees that the holder is the only transaction accessing the table in any way.

  Acquired by the **ALTER TABLE**, **DROP TABLE**, **TRUNCATE**, **REINDEX**, **CLUSTER**, and **VACUUM FULL** commands.

  This is also the default lock mode for LOCK TABLE statements that do not specify a mode explicitly.

- **NOWAIT**

  Specifies that LOCK TABLE does not wait for any conflicting locks to be released. If the lock cannot be obtained immediately, the command exits and an error message is displayed.

  If a table-level lock is obtained without specifying **NOWAIT** and other mutex locks exist, the system waits for other locks to be released.

- **CANCELABLE**

  Allows the waiting thread to send CANCEL signals to the holding threads and waiting threads.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Only the redistribution tool can use this parameter. An error message is displayed when the parameter is used by users.

Examples

- SHARE ROW EXCLUSIVE

```
-- Create a schema.
gaussdb=# CREATE SCHEMA tpcds;

-- Create the tpcds.reason table.
gaussdb=# CREATE TABLE tpcds.reason (
r_reason_sk         INTEGER      NOT NULL,
r_reason_id         CHAR(16)     NOT NULL,
r_reason_desc       INTEGER
);

-- Insert multiple records into the table.
gaussdb=# INSERT INTO tpcds.reason VALUES (1, 'AAAAAAAABAAAAAAA', '18'),(5,
'AAAAAAAACAAAAAAA', '362'),(7, 'AAAAAAAADAAAAAAA', '585');

-- Create a table named reason_t1.
gaussdb=# CREATE TABLE tpcds.reason_t1 AS TABLE tpcds.reason;

-- Start a transaction.
gaussdb=# START TRANSACTION;

-- Use the SHARE ROW EXCLUSIVE MODE lock mode to lock a table.
gaussdb=# LOCK TABLE tpcds.reason_t1 IN SHARE ROW EXCLUSIVE MODE;

-- If you perform the DELETE operation on another device, the operation is blocked.
gaussdb=# DELETE FROM tpcds.reason_t1 WHERE r_reason_sk = 7;

-- End the transaction and release the lock.
gaussdb=# COMMIT;
```

- ROW EXCLUSIVE

```
-- Start a transaction.
gaussdb=# START TRANSACTION;

-- After the UPDATE statement is executed, request a ROW EXCLUSIVE lock on the referenced table.
gaussdb=# UPDATE tpcds.reason_t1 SET r_reason_desc=180 WHERE r_reason_sk=1;

-- When the ALTER TABLE statement is executed on another device, the operation is blocked.
gaussdb=# ALTER TABLE tpcds.reason_t1 ADD r_reason_asc int;

-- End the transaction and release the lock.
gaussdb=# COMMIT;
```

- ACCESS EXCLUSIVE

```
-- Start a transaction.
gaussdb=# START TRANSACTION;

-- After the TRUNCATE statement is executed, request an ACCESS SHARE lock on the referenced table.
gaussdb=# TRUNCATE tpcds.reason_t1;

-- When the SELECT statement is executed on another device, the operation is blocked.
gaussdb=# SELECT * FROM tpcds.reason_t1;

-- End the transaction and release the lock.
gaussdb=# COMMIT;

-- Drop the tpcds.reason_t1 table.
gaussdb=# DROP TABLE tpcds.reason_t1;

-- Drop the tpcds.reason table.
gaussdb=# DROP TABLE tpcds.reason;

-- Drop the schema.
gaussdb=# DROP SCHEMA tpcds CASCADE;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.15.3 LOCK BUCKETS

Description

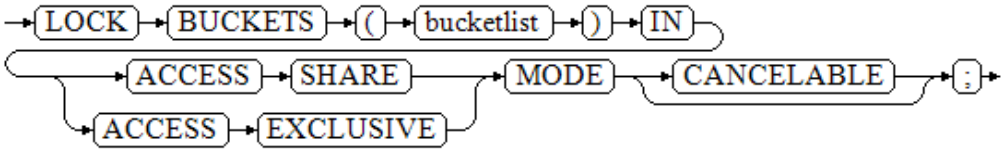
Bucket-based lock.

Precautions

- It can be called only during hash bucket scaling out.
- LOCK BUCKETS is useless outside a transaction block: the lock would remain held only to the completion of the statement. If LOCK BUCKETS is out of any transaction block, an error is reported.

Syntax

```
LOCK BUCKETS (bucketlist) IN {ACCESS SHARE|ACCESS EXCLUSIVE} MODE [CANCELABLE];
```



Parameters

Parameters of LOCK BUCKETS are as follows:

- **bucketlist**

  Specifies the list of buckets to be locked.

  Value range: For each bucket, the value range is [0,1023].
- **ACCESS SHARE**

  Allows only read operations on a bucket.
- **ACCESS EXCLUSIVE**

  Guarantees that the holder is the only transaction accessing the bucket in any way.
- **CANCELABLE**

  Allows the waiting thread to send CANCEL signals to the holding threads and waiting threads.

Examples

```
gaussdb=# START TRANSACTION;

gaussdb=# LOCK BUCKETS (0,1,2,3) IN ACCESS EXCLUSIVE MODE;

gaussdb=# COMMIT;
```

1.13.16 M

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.16.1 MARK BUCKETS

Description

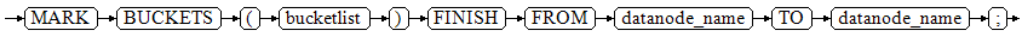
Used by the scale-out tool to notify the kernel of the buckets that have been migrated.

Precautions

It can be called by administrators only during hash bucket scale-out.

Syntax

```
MARK BUCKETS (bucketlist) FINISH FROM datanode_name TO datanode_name;
```



Parameters

- **bucketlist**

  Specifies the list of buckets that have been migrated.

  Value range: For each bucket, the value range is [0,1023].

- **datanode_name**

  Specifies the name of the node involved in bucket migration.

  Value range: character strings complying with **Identifier Naming Conventions**.

Examples

```
gaussdb=# MARK BUCKETS (0,1,2,3) FINISH FROM datanode1 TO datanode3;
```

1.13.16.2 MERGE INTO

Description

MERGE INTO conditionally matches data in a target table with that in a source table. If data matches, UPDATE is executed on the target table; if data does not match, INSERT is executed. You can use this syntax to run UPDATE and INSERT at a time for convenience.

Precautions

- You have the INSERT and UPDATE permissions for the target table and the SELECT permission for the source table.
- MERGE INTO cannot be executed during redistribution.
- If the source table of the MERGE INTO operation contains data columns that are dynamically anonymized, the result of inserting data to or updating data in the target table is the anonymized value and cannot be restored.
- The STREAM plan of MERGE INTO does not allow concurrent DELETE and UPDATE operations, but other plans do.
- The MERGE INTO statement does not support triggers.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax

```
MERGE [/*+ plan_hint */] INTO table_name [ [ AS ] alias ]
USING { { table_name | view_name } | subquery } [ [ AS ] alias ]
ON ( condition )
[
  WHEN MATCHED THEN
  UPDATE SET { column_name = { expression  | DEFAULT } |
         ( column_name [, ...] ) = ( { expression  | DEFAULT } [, ...] ) } [, ...]
  [ WHERE condition ]
]
[
  WHEN NOT MATCHED THEN
  INSERT { DEFAULT VALUES |
  [ ( column_name [, ...] ) ] VALUES ( { expression  | DEFAULT } [, ...] ) [, ...] [ WHERE condition ] }
];
```

Parameters

- plan_hint clause

  Follows the MERGE keyword in the /*+ */ format. It is used to optimize the plan of a MERGE statement block using hints. For details, see "Hint-based Optimization" in *SQL Optimization Guide*. In each statement, only the first **/*+** *plan_hint* ***/** comment block takes effect as a hint. Multiple hints can be written.

- INTO clause

  Specifies the target table that is being updated or has data being inserted. If the target table is a replication table, the default value of a column (such as auto-increment column) in the target table cannot be the volatile function. If **enable_stream_operator** is set to **off**, the target table must contain a primary key or UNIQUE and NOT NULL constraints.

  - **table_name**

    Specifies the name of the target table.

  - **alias**

    Specifies the alias of the target table.

    Value range: character strings complying with **Identifier Naming Conventions**.

- USING clause

  Specifies the source table, which can be a table, view, or subquery. If the target table is a replication table, the USING clause cannot contain non-replication tables.

  - **table_name | view_name**

    Name of the source table or view.

  - **subquery**

    A subquery that provides the rows to be merged into the target table.

- ON clause

  condition

  Specifies the condition used to match data between the source and target tables. Columns in the condition cannot be updated.

- WHEN MATCHED clause

  Performs UPDATE if data in the source table matches that in the target table based on the condition. For details about the syntax, see **UPDATE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Distribution keys cannot be updated. System catalogs and system columns cannot be updated.

- WHEN NOT MATCHED clause

  Performs INSERT if data in the source table does not match that in the target table based on the condition. For details about the syntax, see **INSERT**.

  An INSERT clause cannot contain multiple VALUES.

  The order of WHEN MATCHED and WHEN NOT MATCHED clauses can be reversed. One of them can be used by default, but they cannot be both used at one time. Two WHEN MATCHED or WHEN NOT MATCHED clauses cannot be specified at the same time.

- **DEFAULT**

  Specifies the default value of a column.

  The value is **NULL** if there is no default value.

- WHERE condition

  Specifies the conditions for the UPDATE and INSERT clauses. The two clauses will be executed only when the conditions are met. The default value can be used. System columns cannot be referenced in WHERE condition. You are advised not to use numeric types such as int as conditions because such types can be implicitly converted to bool values (non-zero values are implicitly converted to **true** and **0** is implicitly converted to **false**), which may cause unexpected results.

Examples

```
-- Create the target table products and source table newproducts, and insert data to them.
gaussdb=# CREATE TABLE products
(
product_id INTEGER,
product_name VARCHAR2(60),
category VARCHAR2(60)
);

gaussdb=# INSERT INTO products VALUES (1501, 'vivitar 35mm', 'electrncs');
gaussdb=# INSERT INTO products VALUES (1502, 'olympus is50', 'electrncs');
gaussdb=# INSERT INTO products VALUES (1600, 'play gym', 'toys');
gaussdb=# INSERT INTO products VALUES (1601, 'lamaze', 'toys');
gaussdb=# INSERT INTO products VALUES (1666, 'harry potter', 'dvd');

gaussdb=# CREATE TABLE newproducts
(
product_id INTEGER,
product_name VARCHAR2(60),
category VARCHAR2(60)
);

gaussdb=# INSERT INTO newproducts VALUES (1502, 'olympus camera', 'electrncs');
gaussdb=# INSERT INTO newproducts VALUES (1601, 'lamaze', 'toys');
gaussdb=# INSERT INTO newproducts VALUES (1666, 'harry potter', 'toys');
gaussdb=# INSERT INTO newproducts VALUES (1700, 'wait interface', 'books');

-- Run MERGE INTO.
gaussdb=# MERGE INTO products p
USING newproducts np
ON (p.product_id = np.product_id)
WHEN MATCHED THEN
  UPDATE SET p.product_name = np.product_name, p.category = np.category WHERE p.product_name !=
'play gym'
WHEN NOT MATCHED THEN
  INSERT VALUES (np.product_id, np.product_name, np.category) WHERE np.category = 'books';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
MERGE 4

-- Query updates.
gaussdb=# SELECT * FROM products ORDER BY product_id;
 product_id |  product_name  | category
------------+----------------+----------
       1501 | vivitar 35mm   | electrncs
       1502 | olympus camera | electrncs
       1600 | play gym       | toys
       1601 | lamaze         | toys
       1666 | harry potter   | toys
       1700 | wait interface | books
(6 rows)

-- Drop tables.
gaussdb=# DROP TABLE products;
gaussdb=# DROP TABLE newproducts;
```

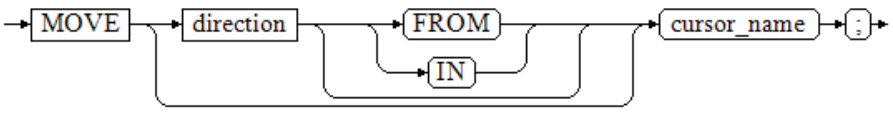
1.13.16.3 MOVE

Description

MOVE repositions a cursor without retrieving any data. MOVE works exactly like the **FETCH** statement, except MOVE only repositions the cursor and does not return rows.

Syntax

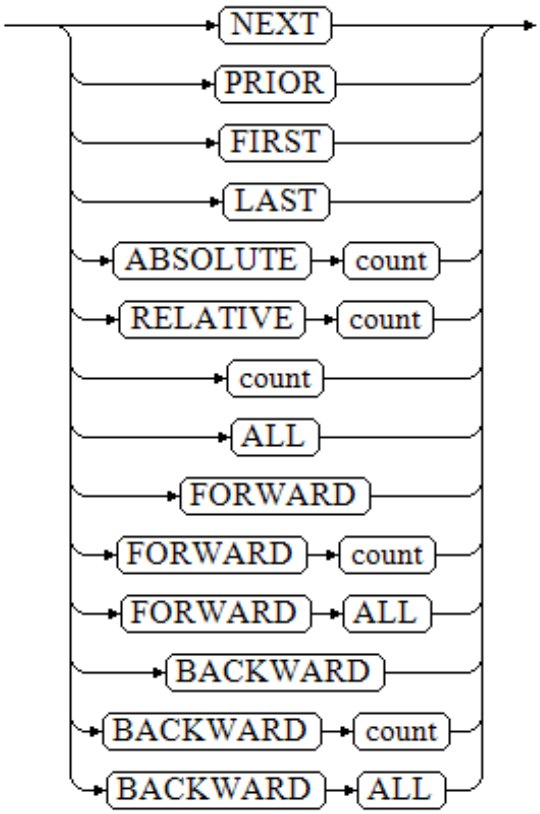
```
MOVE [ direction [ FROM | IN ] ] cursor_name;
```



The direction clause specifies optional parameters.

```
{ NEXT
  | PRIOR
  | FIRST
  | LAST
  | ABSOLUTE count
  | RELATIVE count
  | count
  | ALL
  | FORWARD
  | FORWARD count
  | FORWARD ALL
  | BACKWARD
  | BACKWARD count
  | BACKWARD ALL }
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

The parameters of MOVE and FETCH are the same. For details, see **Parameters** in **FETCH**.

NOTE

On successful completion, a MOVE statement returns a tag of the form **MOVE count**. The **count** is the number of rows that a FETCH statement with the same parameters would have returned (possibly zero).

Examples

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE tbl_test(c1 int);
gaussdb=# INSERT INTO tbl_test VALUES (generate_series(1,20));

-- Set up cursor1.
gaussdb=# BEGIN;
gaussdb=# CURSOR cursor1 FOR SELECT * FROM tbl_test ORDER BY 1;

-- Run the MOVE command to move the cursor backwards by five rows. No result is returned.
gaussdb=# MOVE FORWARD 5 FROM cursor1;
MOVE 5

-- Run the FETCH command to retrieve two rows of data.
gaussdb=# FETCH FORWARD 2 FROM cursor1;
 c1
----
  6
  7
(2 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Close the cursor and end the transaction.
gaussdb=# CLOSE cursor1;
gaussdb=# END;

-- Delete.
gaussdb=# DROP TABLE tbl_test;
```

Helpful Links

CLOSE and **FETCH**

1.13.17 P

1.13.17.1 PREDICT BY

Description

Uses a trained model to perform prediction tasks.

Precautions

This syntax is not supported currently.

1.13.17.2 PREPARE

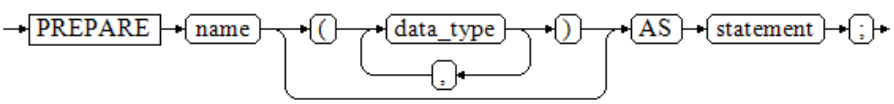
Description

Creates a prepared statement.

A prepared statement is a performance optimizing object on the server. When the PREPARE statement is executed, the specified query is parsed, analyzed, and rewritten. When EXECUTE is executed, the prepared statement is planned and executed. This avoids repetitive parsing and analysis. After the PREPARE statement is created, it exists throughout the database session. Once it is created (even if in a transaction block), it will not be deleted when a transaction is rolled back. Unless automatically deleted upon session termination, the PREPARE statement can only be deleted by explicitly calling **DEALLOCATE**.

Syntax

```
PREPARE name [ ( data_type [, ...] ) ] AS statement;
```



Parameters

- **name**

  Prepared statement name. It must be unique in the session.

- **data_type**

  Specifies the type of an argument.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **statement**

  Specifies a SELECT, INSERT, UPDATE, DELETE, MERGE INTO, or VALUES statement.

  NOTE

  During the parsing of the PREPARE statement, all parameters are parsed to the unknown data type by default. If the parsing involves the selection of operators or functions, follow the rules for handling parameters of the unknown type in **Operators** and **Functions**.

Examples

See **Examples** in section "EXECUTE."

Helpful Links

DEALLOCATEand **EXECUTE**

1.13.17.3 PREPARE TRANSACTION

Description

Prepares the current transaction for two-phase commit.

After this statement, the transaction is no longer associated with the current session; instead, its state is fully stored on disk, and there is a high probability that it can be committed successfully, even if a database crash occurs before the commit is requested.

Once prepared, a transaction can be committed using **COMMIT PREPARED** or rolled back using **ROLLBACK PREPARED**. Those statements can be issued from any session, not only the one that executed the original transaction.

From the point of view of the issuing session, **PREPARE TRANSACTION** is not unlike a **ROLLBACK** statement: after executing it, there is no active current transaction, and the effects of the prepared transaction are no longer visible. (The effects will become visible again if the transaction is committed.)

If the **PREPARE TRANSACTION** statement fails for any reason, it becomes a **ROLLBACK** and the current transaction is canceled.

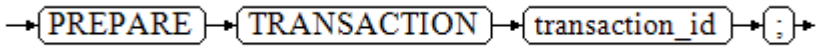
Precautions

- The transaction function is maintained automatically by the database, and should be not visible to users.
- Currently, the distributed system does not allow users to call the user-defined PREPARE TRANSACTION operation. This syntax is reserved only for local two-phase cleanup on CNs in maintenance mode (the GUC parameter **xc_maintenance_mode** is set to **on**) and does not ensure the global transaction capability.
- When running the **PREPARE TRANSACTION** statement, increase the value of **max_prepared_transactions** in configuration file **gaussdb.conf**. You are advised to set it to a value not less than that of **max_connections** so that one pending prepared transaction is available for each session.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax

```
PREPARE TRANSACTION transaction_id;
```



Parameters

transaction_id

Specifies an arbitrary identifier that later identifies this transaction for **COMMIT PREPARED** or **ROLLBACK PREPARED**. The identifier must be different from those for current prepared transactions.

Value range: The identifier must be written as a string literal, and must be less than 200 bytes long.

Examples

```
-- Start a transaction in maintenance mode.
gaussdb=# SET XC_MAINTENANCE_MODE = ON;
SET

gaussdb=# BEGIN;
BEGIN

-- Prepare a transaction whose identifier is trans_test.
gaussdb=# PREPARE TRANSACTION 'trans_test';
PREPARE TRANSACTION

-- Cancel the transaction whose identifier is trans_test.
gaussdb=# ROLLBACK PREPARED 'trans_test';
ROLLBACK PREPARED
```

Helpful Links

COMMIT PREPARED and **ROLLBACK PREPARED**

1.13.17.4 PURGE

Description

The PURGE statement can be used to:

- Clear tables or indexes from the recycle bin and release all space related to the objects.
- Clear the recycle bin.

Precautions

- The PURGE operation supports tables (PURGE TABLE), indexes (PURGE INDEX), and recycle bins (PURGE RECYCLEBIN).
- The permission requirements for performing the PURGE operation are as follows:
  - **PURGE TABLE**: The user must be the owner of the table and must have the USAGE permission on the schema to which the table belongs. When

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

separation of duties is disabled, system administrators have this authorization by default.

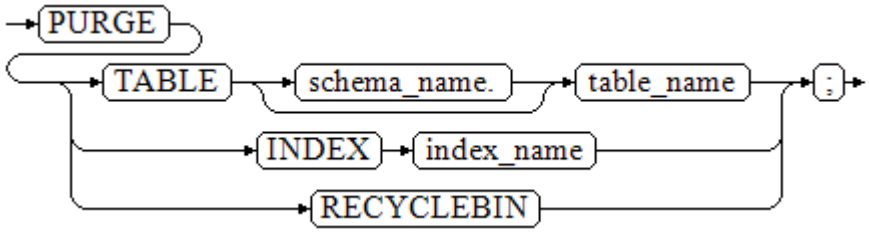
- **PURGE INDEX**: The user must be the owner of the index and must have the USAGE permission on the schema to which the index belongs. When separation of duties is disabled, system administrators have this authorization by default.
- **PURGE RECYCLEBIN**: Only users with the SYSADMIN or SYSDBA permissions can perform this operation. Common users do not have these permissions. Common users can purge only the objects owned by themselves in the recycle bin. In addition, the user must have the USAGE permission of the schema to which the objects belong. When the separation of duties is disabled, system administrators can purge all objects in the recycle bin by default.

Prerequisites

- The **enable_recyclebin** parameter has been enabled to enable the table-level recycle bin. For details about how to use the parameter, contact an administrator.
- The **recyclebin_retention_time** parameter has been set to specify how long objects will remain in the recycle bin before being automatically deleted. Contact an administrator for the parameter usage.

Syntax

```
PURGE { TABLE [schema_name.]table_name
      | INDEX index_name
      | RECYCLEBIN
    };
```



Parameters

- **schema_name**

  Specifies a schema name.
- **TABLE [schema_name.] table_name**

  Clears a specified table in the recycle bin. The table can be schema-qualified.
- **INDEX index_name**

  Clears a specified index in the recycle bin.
- **RECYCLEBIN**

  Clears all objects in the recycle bin.

Examples

```
-- Create the reason_t1 table.
gaussdb=# CREATE TABLE reason_t1(
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    r_reason_sk    integer,
    r_reason_id    character(16),
    r_reason_desc  character(100)
) WITH(STORAGE_TYPE = ustore);

-- Create the reason_t2 table.
gaussdb=# CREATE TABLE reason_t2(
    r_reason_sk    integer,
    r_reason_id    character(16),
    r_reason_desc  character(100)
) WITH(STORAGE_TYPE = ustore);

-- Add indexes to the reason_t1 and reason_t2 tables.
gaussdb=#  CREATE INDEX idx_t1 on reason_t1(r_reason_id);
gaussdb=#  CREATE INDEX idx_t2 on reason_t2(r_reason_id);
gaussdb=#  DROP TABLE reason_t1;
gaussdb=#  DROP TABLE reason_t2;

-- View the recycle bin.
gaussdb=#  SELECT rcyname,rcyoriginname,rcytablespace FROM GS_RECYCLEBIN;
          rcyname            | rcyoriginname | rcytablespace
-----------------------------+---------------+---------------
 BIN$31C94EB4207$8001$0==$0 | reason_t1     |             0
 BIN$31C94EB420D$8001$0==$0 | idx_t1        |             0
 BIN$31C94EB420A$8004$0==$0 | reason_t2     |             0
 BIN$31C94EB420E$8004$0==$0 | idx_t2        |             0
(4 rows)

-- Purge the table.
gaussdb=#  PURGE TABLE reason_t1;
gaussdb=#  SELECT rcyname,rcyoriginname,rcytablespace FROM GS_RECYCLEBIN;
          rcyname            | rcyoriginname | rcytablespace
-----------------------------+---------------+---------------
 BIN$31C94EB420A$8004$0==$0 | reason_t2     |             0
 BIN$31C94EB420E$8004$0==$0 | idx_t2        |             0
(2 rows)

-- Purge the index.
gaussdb=# PURGE INDEX idx_t2;
gaussdb=# SELECT rcyname,rcyoriginname,rcytablespace FROM GS_RECYCLEBIN;
          rcyname            | rcyoriginname | rcytablespace
-----------------------------+---------------+---------------
 BIN$31C94EB420A$8004$0==$0 | reason_t2     |             0
(1 row)

-- Purge all objects in the recycle bin.
gaussdb=#  PURGE recyclebin;
gaussdb=#  SELECT rcyname,rcyoriginname,rcytablespace FROM GS_RECYCLEBIN;
        rcyname         | rcyoriginname | rcytablespace
------------------------+---------------+---------------
(0 rows)
```

1.13.18 R

1.13.18.1 REASSIGN OWNED

Description

Changes the owner of the database object.

REASSIGN OWNED changes the database object owner of an old role to a new role.

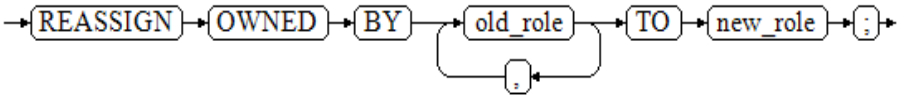
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

- REASSIGN OWNED is often executed before role deletion.
- To run the REASSIGN OWNED statement, you must have the permissions of the original and target roles.

Syntax

```
REASSIGN OWNED BY old_role [, ...] TO new_role;
```



Parameters

- **old_role**

  Specifies the role name of the old owner.

- **new_role**

  Specifies the role name of the new owner. Note: Only the initial user can use the REASSIGN OWNED syntax to change the owner to the initial user.

Examples

```
-- Create the test_jim and test_tom users.
gaussdb=# CREATE USER test_jim PASSWORD '********';
gaussdb=# CREATE USER test_tom PASSWORD '********';

-- View the user with the same name as owner of the automatically created schema.
gaussdb=# \dn test*
   List of schemas
   Name   |  Owner
----------+----------
 test_jim | test_jim
 test_tom | test_tom
(2 rows)

-- Change the owner of all database objects owned by test_jim to test_tom.
gaussdb=# REASSIGN OWNED BY test_jim TO test_tom;

-- View the schema information. The owner of the test_jim schema is changed to test_tom.
gaussdb=# \dn test*
   List of schemas
   Name   |  Owner
----------+----------
 test_jim | test_tom
 test_tom | test_tom
(2 rows)

-- Delete the test_jim and test_tom users.
gaussdb=# DROP USER test_jim, test_tom CASCADE;
```

Helpful Links

DROP OWNED

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.18.2 REFRESH INCREMENTAL MATERIALIZED VIEW

Description

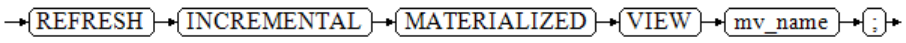
REFRESH INCREMENTAL MATERIALIZED VIEW refreshes a materialized view in incremental mode.

Precautions

- Incremental refresh supports only fast-refresh materialized views.
- To refresh a materialized view, you must have the SELECT permission on the base table.

Syntax

```
REFRESH INCREMENTAL MATERIALIZED VIEW mv_name;
```



Parameters

- **mv_name**

  Name of the materialized view to be refreshed.

Examples

```
-- Create an ordinary table.
gaussdb=# CREATE TABLE my_table (c1 int, c2 int);

-- Create a fast-refresh materialized view.
gaussdb=# CREATE INCREMENTAL MATERIALIZED VIEW my_imv AS SELECT * FROM my_table;

-- Write data to the base table.
gaussdb=# INSERT INTO my_table VALUES(1,1),(2,2);

-- Fast refresh the fast-refresh materialized view my_imv.
gaussdb=# REFRESH INCREMENTAL MATERIALIZED VIEW my_imv;

-- View the fast-refresh materialized view.
gaussdb=# SELECT * FROM my_imv;
 c1 | c2
----+----
  1 |  1
  2 |  2
(2 rows)

-- Delete the fast-refresh materialized view.
gaussdb=# DROP MATERIALIZED VIEW my_imv;

-- Delete the my_table table.
gaussdb=# DROP TABLE my_table;
```

Helpful Links

ALTER MATERIALIZED VIEW, **CREATE INCREMENTAL MATERIALIZED VIEW**, **CREATE MATERIALIZED VIEW**, **CREATE TABLE**, **DROP MATERIALIZED VIEW**, and **REFRESH MATERIALIZED VIEW**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.18.3 REFRESH MATERIALIZED VIEW

Description

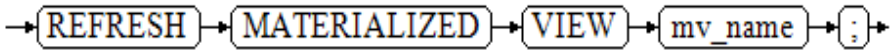
REFRESH MATERIALIZED VIEW refreshes materialized views in complete-refresh mode.

Precautions

- Full refreshing can be performed on both complete- and fast-refresh materialized views.
- To refresh a materialized view, you must have the SELECT permission on the base table.

Syntax

```
REFRESH MATERIALIZED VIEW mv_name;
```



Parameters

mv_name

Name of the materialized view to be refreshed.

Examples

```
-- Create an ordinary table.
gaussdb=# CREATE TABLE my_table (c1 int, c2 int);

-- Create a complete-refresh materialized view.
gaussdb=# CREATE MATERIALIZED VIEW my_mv AS SELECT * FROM my_table;

-- Create a fast-refresh materialized view.
gaussdb=# CREATE INCREMENTAL MATERIALIZED VIEW my_imv AS SELECT * FROM my_table;

-- Write data to the base table.
gaussdb=# INSERT INTO my_table VALUES(1,1),(2,2);

-- Completely refresh the complete-refresh materialized view my_mv.
gaussdb=# REFRESH MATERIALIZED VIEW my_mv;

-- Completely refresh the fast-refresh materialized view my_imv.
gaussdb=# REFRESH MATERIALIZED VIEW my_imv;

-- Delete the fast-refresh materialized view.
gaussdb=# DROP MATERIALIZED VIEW my_imv;

-- Delete the complete-refresh materialized view.
gaussdb=# DROP MATERIALIZED VIEW my_mv;

-- Delete the my_table table.
gaussdb=# DROP TABLE my_table;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

ALTER MATERIALIZED VIEW, **CREATE INCREMENTAL MATERIALIZED VIEW**, **CREATE MATERIALIZED VIEW**, **CREATE TABLE**, **DROP MATERIALIZED VIEW**, and **REFRESH INCREMENTAL MATERIALIZED VIEW**

1.13.18.4 REFRESH SYSTEM OBJECT

Description

Updates the version information of system objects to the system catalog **GS_SYSTEM_OBJECTS_VERSION** to determine the changes of system objects during the upgrade.

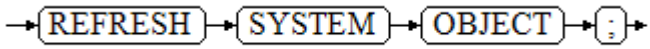
It can be called during the upgrade from a version earlier than 507.0. This is an internal syntax; its use is therefore discouraged.

Precautions

- This statement can be called only in the upgrade phase. That is, this statement can be executed only when the GUC parameter **upgrade_mode** is not set to **0**.
- Only the initial user has permission to execute this statement.

Syntax

```
REFRESH SYSTEM OBJECT;
```



1.13.18.5 REINDEX

Description

Rebuilds an index using the data stored in the index's table, replacing the old copy of the index.

There are several scenarios in which REINDEX can be used:

- An index has become corrupted, and no longer contains valid data.
- An index has become "bloated", that is, it contains many empty or nearly-empty pages.
- You have altered a storage parameter (such as fillfactor) for an index, and wish that the change takes full effect.
- After an index becomes invalid, its statistics are not updated. You are advised to reindex it.

Precautions

- REINDEX DATABASE and REINDEX SYSTEM cannot be performed in transaction blocks.
- Currently, only REINDEX INDEX and REINDEX TABLE are supported for global secondary indexes.

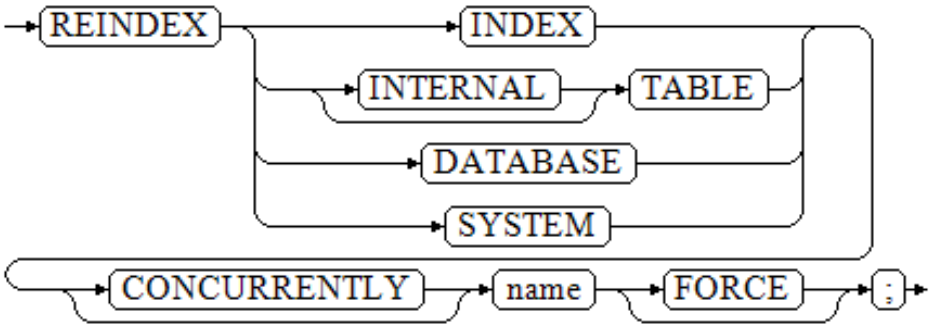
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- When REINDEX is executed, if the index option **lpi_parallel_method** is set to **PARTITION** and the table option **parallel_workers** is set to a value greater than 0, indexes are rebuilt in parallel by partition. If the **lpi_parallel_method** option is not set or is set to **AUTO**, indexes are rebuilt in parallel by page by default.
- When REINDEX CONCURRENTLY is used to rebuild indexes online, the index sequence in the table changes, and the query plan may change.
- REINDEX [CONCURRENTLY] does not support the modification of the RCR and PCR types of UB-tree indexes. That is, the index type after REINDEX is the same as that when the index is created.
- REINDEX CONCURRENTLY rebuilds indexes online. Indexes based on the **rowid** system column cannot be rebuilt.

Syntax

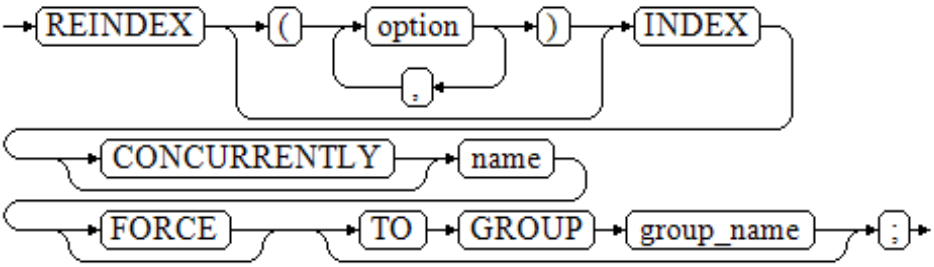
- Rebuild an ordinary index.

```
REINDEX { INDEX | [INTERNAL] TABLE | DATABASE | SYSTEM } [ CONCURRENTLY ] name [ FORCE ];
```



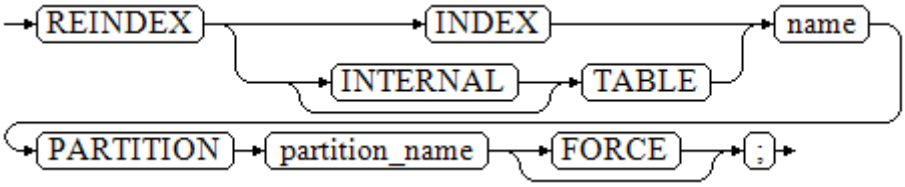
- Rebuild the index and convert the type.

```
REINDEX [ ( option [, ...] ) ] { INDEX } [ CONCURRENTLY ] name [ FORCE ] [TO GROUP group_name];
```



- Rebuild an index partition.

```
REINDEX { INDEX | [INTERNAL] TABLE } name
   PARTITION partition_name [ FORCE ];
```



Parameters

- **INDEX**

  Rebuilds the specified index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **TABLE**

  Rebuilds all indexes of a specified table. The TOAST table (if any) of the table is reindexed as well. If an index on the table has been invalidated by running **alter unusable**, the index cannot be rebuilt. Indexes in the TOAST table cannot be rebuilt when specifying the **CONCURRENTLY** option.

- **DATABASE**

  Rebuilds all indexes within the current database. Indexes in the TOAST table within the current database cannot be rebuilt when specifying the CONCURRENTLY option.

- **SYSTEM**

  Rebuilds all indexes on system catalogs within the current database. Indexes on user tables are not processed.

- **option**

  Currently, **CROSSBUCKET** and **TABLEOIDTRANS** are supported. For **CROSSBUCKET**, the value can only be **ON** or **OFF**. For **TABLEOIDTRANS**, the value cannot be specified.

  a. **CROSSBUCKET** controls whether the index of a hash bucket table is converted to a cross-bucket index (CBI) or local-bucket index (LBI) during the REINDEX process. This conversion supports only the index of the distributed hash bucket table and does not support the GSI index. In the current version, only the session of the scale-out process can use the online index type conversion.

  b. **TABLEOIDTRANS** controls whether the internal data format of the GSI index on a hash bucket table is converted during index rebuilding.

- **CONCURRENTLY**

  Rebuilds an index (with ShareUpdateExclusiveLock) in non-blocking DML mode. When an index is rebuilt, other statements cannot access the table on which the index depends. If this keyword is specified, DML is not blocked during the recreation. Indexes in system catalogs cannot be rebuilt online. REINDEX INTERNAL TABLE CONCURRENTLY, REINDEX SYSTEM CONCURRENTLY, and REINDEX INVALID INDEX CONCURRENTLY are not supported. When REINDEX DATABASE CONCURRENTLY is executed, all indexes on user tables in the current database are rebuilt online (indexes on system catalogs are not processed). REINDEX CONCURRENTLY cannot be executed within a transaction. Online index rebuilding supports only B-tree and UB-tree indexes, ordinary indexes, global indexes, and local indexes. PCR UB-tree indexes and subpartitions are not supported. GSIs can be rebuilt online using REINDEX INDEX CONCURRENTLY, but expression GSIs and partial GSIs cannot be rebuilt online. During online GSI rebuild, do not execute DDL statements concurrently with other DDL statements. Otherwise, deadlocks or errors may occur. You can reconnect to the client and try again. Online parallel index rebuilding supports only ordinary indexes, global indexes, local indexes, and GSIs of Astore and Ustore. Other online index rebuilding specifications are inherited from the current version. If online index rebuilding fails, the system automatically clears the new indexes to prevent resource occupation in scenarios such as manual cancellation, duplicate unique index key values, insufficient resources, thread startup failure, and lock timeout. The automatic index clearing process includes a phase where the system must wait for other database transactions to end. As a result, issues such as lock timeout and deadlock may occur, causing index clearing failures. If the system

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

cannot automatically clear invalid new indexes (for example, due to database crashes, FATAL, or PANIC), you need to manually clear invalid new indexes (using the DROP INDEX statement) and temporary tables (using the DROP TABLE statement) as soon as possible to prevent further resource occupation. Generally, the suffix of an invalid index is **_ccnew**. For GSIs, the prefix of an invalid new index is **ccgsi_tmp_index**_. In the distributed system, failed indexes in DNs are automatically cleared and those in CNs need to be manually cleared. If a critical error occurs, the metadata of the failed indexes cannot be found on some nodes in the distributed system. The DROP INDEX statement cannot be used to drop these indexes. You need to use the DROP INDEX IF EXISTS statement to drop them. The execution of REINDEX INDEX CONCURRENTLY adds a four-level session lock to the table and its first several phases are similar to those of CREATE INDEX CONCURRENTLY. Therefore, the execution may be suspended or deadlocked, which is similar to that of CREATE INDEX CONCURRENTLY. For example, if two sessions perform the REINDEX CONCURRENTLY operation on the same index or table at the same time, a deadlock occurs. For details, see **CONCURRENTLY** in **CREATE INDEX**.

NOTE

This keyword is specified when an index is rebuilt. For Astore, you need to complete two full table scans for building. During the first scan, a new index is created without blocking read and write operations. During the second scan, the changes in the first scan are merged and updated. For Ustore, you need to complete a full table scan. During the scan, data generated by concurrent DML operations is inserted into the temporary table named **index_oid_cctmp**. After the scan is complete, you merge the temporary table to the new index suffixed with **_ccnew{n}** (for GSIs, the prefix is **ccgsi_tmp_index**_), delete the temporary table, exchange the old and new indexes, mark the old index as dead, enable the new index, and rebuild the index.

If an online index rebuild operation is performed during the generation of a WDR snapshot, the error message "Could not open relation with OID %u." may be displayed, causing the WDR snapshot generation to fail. This is an expected behavior. After the index rebuild is complete, generate the WDR snapshot again.

- **name**

  Specifies the name of the index, table, or database whose index needs to be rebuilt. Tables and indexes can be schema-qualified.

  NOTE

  REINDEX DATABASE and SYSTEM can create indexes for only the current database. Therefore, **name** must be the same as the current database name.

- **FORCE**

  It is provided solely for syntax compatibility and does not serve any practical purpose.

- **TO GROUP group_name**

  Rebuilds the index to the specified node group named **group_name**. This option is used only in the scale-out process.

- **partition_name**

  Name of the index or table to be reindexed.

  The options are as follows:

  – If REINDEX INDEX is used, specify the name of an index.

  – If REINDEX TABLE is used, specify the name of a table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- REINDEX DATABASE and REINDEX SYSTEM cannot be performed in transaction blocks.
- REINDEX and REINDEX CONCURRENTLY do not support separate operations on TOAST tables or TOAST indexes.
- During the execution of REINDEX CONCURRENTLY, an error may be reported for the stream plan involving indexes. Before executing REINDEX CONCURRENTLY, you can use an SQL patch or add hints to SQL statements to disable streaming and use the PGXC plan.

Examples

```
-- Create the table tbl_test and insert data into the tables.
gaussdb=# CREATE TABLE tbl_test(c1 int,c2 varchar) WITH (storage_type = ustore);
gaussdb=# INSERT INTO tbl_test VALUES (1, 'AAAAAAA'),(5, 'AAAAAAB'),(10, 'AAAAAAC');

-- Create an index and check the index size.
gaussdb=# CREATE INDEX idx_test_c1 ON tbl_test(c1);
gaussdb=# SELECT pg_size_pretty(pg_total_relation_size('idx_test_c1')) AS size;
  size
--------
 176 kB
(1 row)

-- Insert 10,000 data records and then delete the data. It is found that the index becomes larger.
gaussdb=# INSERT INTO tbl_test VALUES (generate_series(1,10000),'test');
gaussdb=# DELETE FROM tbl_test WHERE c2 = 'test';
gaussdb=# SELECT pg_size_pretty(pg_total_relation_size('idx_test_c1')) AS size;
  size
--------
 504 kB
(1 row)

-- After an independent index is rebuilt, the index size is restored to the initial size.
gaussdb=# REINDEX INDEX idx_test_c1;
gaussdb=# SELECT pg_size_pretty(pg_total_relation_size('idx_test_c1')) AS size;
  size
--------
 176 kB
(1 row)

-- Rebuild a single index online.
gaussdb=# REINDEX INDEX CONCURRENTLY idx_test_c1;

-- Rebuild all indexes in the table tbl_test.
gaussdb=# REINDEX TABLE tbl_test;

-- Rebuild all indexes in the table tbl_test online.
gaussdb=# REINDEX TABLE CONCURRENTLY tbl_test;

-- Drop the tbl_test table.
gaussdb=# DROP TABLE tbl_test;
```

Suggestions

- INTERNAL TABLE

  This scenario is used for fault recovery. You are advised not to perform concurrent operations.
- DATABASE

  You are advised not to reindex a database in a transaction.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- SYSTEM

  You are advised not to reindex a system catalog in a transaction.

1.13.18.6 RELEASE SAVEPOINT

Description

RELEASE SAVEPOINT destroys a savepoint previously defined in the current transaction.

Destroying a savepoint makes it unavailable as a rollback point, but it has no other user visible behavior. It does not undo the effects of statements executed after the savepoint was established. To do that, use **ROLLBACK TO SAVEPOINT**. Destroying a savepoint when it is no longer needed allows the system to reclaim some resources earlier than transaction end.

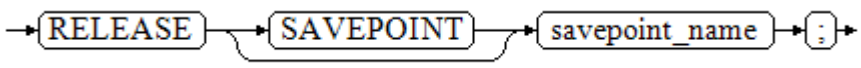
RELEASE SAVEPOINT also deletes all savepoints that were established after the named savepoint was established.

Precautions

- Releasing a savepoint name that was not previously defined will cause an error.
- It is not possible to release a savepoint when the transaction is in an aborted state.
- If multiple savepoints have the same name, only the one that was most recently defined is released.

Syntax

```
RELEASE [ SAVEPOINT ] savepoint_name;
```



Parameters

savepoint_name

Specifies the name of the savepoint you want to destroy.

Examples

```
-- Create a schema.
gaussdb=# CREATE SCHEMA tpcds;

-- Create a table.
gaussdb=# CREATE TABLE tpcds.table1(a int);

-- Start a transaction.
gaussdb=# START TRANSACTION;

--Insert data.
gaussdb=# INSERT INTO tpcds.table1 VALUES (3);

-- Create a savepoint.
gaussdb=# SAVEPOINT my_savepoint;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--Insert data.
gaussdb=# INSERT INTO tpcds.table1 VALUES (4);

-- Delete the savepoint.
gaussdb=# RELEASE SAVEPOINT my_savepoint;

-- Query the table content. The table should contain both 3 and 4.
gaussdb=# SELECT * FROM tpcds.table1;
 a
---
 3
 4
(2 rows)

-- Commit the transaction.
gaussdb=# COMMIT;

-- Query the table content again. You can see 3 and 4 at the same time.
gaussdb=# SELECT * FROM tpcds.table1;
 a
---
 3
 4
(2 rows)

-- Drop the table.
gaussdb=# DROP TABLE tpcds.table1;

-- Delete the schema.
gaussdb=# DROP SCHEMA tpcds CASCADE;
```

Helpful Links

SAVEPOINT and **ROLLBACK TO SAVEPOINT**

1.13.18.7 RENAME TABLE

Description

In the same statement, changing the name of a single table or multiple tables does not affect the stored data.

Precautions

- To change the name of a single table, this syntax is equivalent to RENAME in ALTER TABLE.
- Changing the names of multiple tables is equivalent to running RENAME in ALTER TABLE for multiple times. However, changing the names of local and non-local temporary tables at the same time is not supported.

Syntax

Modify the table name.

```
RENAME { TABLE | TABLES } { table_name TO new_table_name } [,...];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

If you run this command when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** set to **'5.7'**, and **b_format_dev_version** set to **'s2'**, the following situations may occur:

- If the new table name starts with "#mysql50#" and is followed by other characters, "#mysql50#" will be ignored.
- If the old and new table names are the same, no error is reported.

Parameters

- **TABLE | TABLES**

  TABLE and TABLES can be used interchangeably, regardless of the number of tables operated in the statement.

- **table_name TO new_table_name**

  table_name indicates the name of the table to be modified.

  new_table_name indicates the new table name.

  TO is the connection word.

Examples of Modifying Tables

- Rename a single table.

```
gaussdb=# CREATE TABLE aa(c1 int, c2 int);
gaussdb=# RENAME TABLE aa TO test_alt1;
gaussdb=# DROP TABLE test_alt1;
```

- Rename multiple tables.

```
gaussdb=# CREATE TABLE aa(c1 int, c2 int);
gaussdb=# CREATE TABLE bb(c1 int, c2 int);
gaussdb=# RENAME TABLE aa TO test_alt1, bb TO test_alt2;
gaussdb=# DROP TABLE test_alt1,test_alt2;
```

Helpful Links

ALTER TABLE

1.13.18.8 REPLACE

Description

Inserts data into a table or replaces existing data in a table. If the data to be inserted has the primary key or unique key conflicts with the existing data, the REPLACE statement deletes the existing data and then inserts the new data.

You can use REPLACE in the following ways:

- Replace or insert values. That is, VALUES or VALUE is used to construct a row of records and insert the row into the table.
- Replace or insert the query. One or more rows of records are constructed based on the result set returned by SELECT and inserted into a table.
- Set the value of a specified column. Similar to value insertion, the default value is used for columns that are not specified.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Precautions

- To execute this statement, you must have the DELETE and INSERT permissions on the table.
- If the primary key or unique key does not conflict, you can directly insert the data. If the primary key or unique key conflicts, delete the original data and then insert the new data.
- The return format of the REPLACE operation is **REPLACE 0** *X*, where *X* indicates the number of DELETE and INSERT operations.
- In the REPLACE...SELECT syntax, the number of columns in **select_list** must be the same as the number of columns to be inserted.
- In the REPLACE...SET syntax:
  - If **col_name** has a default value, **SET col_name = col_name + 1** is equivalent to **SET col_name =** *Default value of col_name* **+ 1**.
  - If **col_name** has neither default value nor NOT NULL constraint, **SET col_name = col_name + 1** is equivalent to **col_name = NULL**.
  - If **col_name** does not have a default value but has a NOT NULL constraint, the data types that support a default value are timestamp, timestamp with time zone, time, time with time zone, interval, tinterval, smalldatetime, date, datea (valid only in ORA-compatible mode), uuid, name, point, polygon, circle, lseg, box, json, jsonb, xml, varbit, numeric, cidr, inet, macaddr, numrange, int8range, int4range, tsrange, tstzrange, daterange, hash16, hash32, bool, bytea, char, bigint, int, smallint, tinyint, text, raw, blob, clob, float4, float8, abstime, reltime, bpchar, varchar, nvarchar2, money, uint1, uint2, uint4, uint8, and enum. The default values are described in the following table.

Data Type	Default Value of col_name
int8, int4, int2, int1, float8, float4, numeric, uint1, uint2, uint4, and uint8	**0**. If **numeric** specifies a decimal place, the decimal place is displayed. For example, **NUMERIC (10, 3): 0.000**
text, clob, bpchar, varchar, char, nvarchar2, name, blob, raw, and varbit	Empty string.
numrange, int8range, int4range, tsrange, tstzrange, and daterange	empty
bytea	\x
money	$0.00
json, jsonb, and xml	'null'
macaddr	00:00:00:00:00:00
inet	0.0.0.0
cidr	0.0.0.0/32
point	(0,0)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Data Type	Default Value of col_name
lseg	[(0,0),(0,0)]
box	(0,0),(0,0)
path	((0,0))
polygon	((0,0))
circle	<(0,0),0>
uuid	00000000-0000-0000-0000-000000000000
hash16	0000000000000000
hash32	00000000000000000000000000000000
bool	f
abstime	abstime '1970-01-01 00:00:00'
reltime	reltime '00:00:00'
interval	interval '00:00:00'
tinterval	tinterval(abstime '1970-01-01 00:00:00', abstime '1970-01-01 00:00:00')
timestamp	timestamp '1970-01-01 00:00:00'
timestamp with time zone	timestamptz '1970-01-01 00:00:00'
date	date '1970-01-01'
datea	datea '01-JAN-70'
time	time '00:00:00'
time with time zone	timetz '00:00:00'
smalldatetime	smalldatetime '1970-01-01 00:00:00'

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The default value of the enumerated type (ENUM) is the first element. If the first element does not exist, **NULL** is returned.
- In ORA-compatible mode (**sql_compatibility = 'ORA'**), an empty string of the text, clob, blob, raw, bytea, varchar, nvarchar2, bpchar, char, name, byteawithoutorderwithqualcol, or byteawithoutordercol type is equivalent to null. If a column has a NOT NULL constraint, an error is reported when a reference column is used to insert data into a table.
- In MySQL-compatible mode (**sql_compatibility = 'MYSQL'**), if **b_format_version** is set to **'5.7'**, **b_format_dev_version** is set to **'s1'**, and **sql_mode** does not contain **strict_trans_tables**, **only_full_group_by**, **no_zero_in_date**, **no_zero_date**, or **error_for_division_by_zero**, then the default values of **timestamp** and **datetime** are **0000-00-00 00:00:00**, and the default value of **DATE** is **0000-00-00** under the NOT NULL constraint.
- The uint1, uint2, uint4, and uint8 data types are supported only in the MySQL-compatible mode (**sql_compatibility = 'MYSQL'**).
- If the default value of the function expression can be calculated during parsing, the default value is the calculated constant. Otherwise, the value is **NULL**.
- In other scenarios, the default value is **NULL**.
- When the datea type is entered in literal format, the entered character string must comply with the **nls_date_format** format; otherwise, an error may be reported or incorrect data may be generated.

- In the REPLACE... SET syntax, the current **col_name** value depends on the previous **col_name** value. If there is no previous **col_name** value, the default value is used. For example, in the **SET f1 = f1 + 1, f2 = f1** scenario, **f1** is equal to the default value of **f1** (assumed to **1**) plus 1, and **f2** is equal to the calculated value of **f1**, that is **2**.
- The REPLACE INTO statement does not support ROW triggers. For STATEMENT triggers, only INSERT STATEMENT is supported, and DELETE STATEMENT is not supported.
- Deferrable UNIQUE constraints or primary keys are not supported.
- If a table has multiple unique constraints and the inserted data violates multiple unique constraints, all the data that violates the constraints is deleted and new data is inserted. In this scenario, data may be deleted by mistake. Therefore, exercise caution when performing this operation.
- If multiple rows are inserted and these rows have unique constraint conflicts with data in the same row in the table, the REPLACE operation is performed in sequence.
- In **SET col_name = col_name + 1**, the length of **col_name** cannot exceed 1. For example, in **SET col_name = table_name.col_name + 1**, the length of **table_name.col_name** is 2. Formats such as B.A, C.B.A, and D.C.B.A are not supported.
- Tables containing global secondary indexes do not support the REPLACE syntax.
- When comparing floating-point data, note that precision loss may occur.
- If a row-level security policy is created on a table, REPLACE INTO is not supported.
- Foreign tables are not supported.
- Encrypted tables are not supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- MOTs are not supported.

Syntax

- Replace or insert values.

```
REPLACE [ INTO ] table_name
    [ PARTITION ( partition_name [, ... ] ) ]
    [ ( col_name [, ... ] ) ]
    { VALUES | VALUE } ( value [, ... ] ) [, ... ];
```

  The format of **value** is as follows:

```
{ expression | DEFAULT }
```

- Replace or insert the query.

```
REPLACE [ INTO ] table_name
    [ PARTITION ( partition_name [, ... ] ) ]
    [ ( col_name [, ... ] ) ]
    query;
```

- Set the value of a specified column.

```
REPLACE [INTO] table_name
    [ PARTITION ( partition_name [, ... ] ) ]
    SET col_name = value [, ... ];
```

  The format of **value** is as follows:

```
{ expression | DEFAULT }
```

Parameters

- **table_name**

  Specifies the name of the target table where data will be inserted.

  Value range: existing table names.

- **col_name**

  Specifies the name of a column in a table.

  - The column name can be qualified with a subcolumn name or array index, if needed.
  - Each column not present in the column list will be filled with a default value, either its declared default value or **NULL** if there is none. Inserting data into only some columns of a composite type leaves the other columns **NULL**.
  - The target column names (specified by **col_name**) can be listed in any order. If no list of column names is given at all, the default is all the columns of the table in their declared order.
  - The target columns are the first *N* column names, if there are only *N* columns provided by the VALUE clause and QUERY.
  - The values provided by the VALUE clause and QUERY are joined with the corresponding columns from left to right in the table.

  Value range: existing column names.

- **PARTITION (partition_name [, ...])**

  Inserts data to a specified partition. **partition_name** indicates the partition name.

  If the value of the VALUE clause is inconsistent with that of the specified partition, an exception is displayed.

- **value**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the value to be inserted. The value format is as follows:

```
{ expression | DEFAULT }
```

- **expression** indicates that a valid expression or value is assigned to the corresponding column.
  - If single quotation marks are inserted into a column, they must be escaped by using an additional pair of single quotation marks.
  - If the expression for any column is not of the correct data type, automatic type conversion will be attempted. If the attempt fails, data insertion fails, and the system returns an error message.
- **DEFAULT** indicates the default value of the corresponding column name. The value is **NULL** if there is no default value.

- **query**

  Specifies a query statement (SELECT statement) that uses the query result as the inserted data.

Examples

```
-- Create table test.
gaussdb=# CREATE TABLE test(f1 int primary key, f2 int, f3 int);

-- Insert three data records.
gaussdb=# INSERT INTO test VALUES(1, 1, 1), (2, 2, 2), (3, 3, 3);
INSERT 0 3

-- Replace or insert values.
gaussdb=# REPLACE INTO test VALUES(1, 11, 11);
REPLACE 0 2

-- Query the result.
gaussdb=# SELECT * FROM test WHERE f1 = 1;
 f1 | f2 | f3
----+----+----
  1 | 11 | 11
(1 row)

-- Replace or insert the query.
gaussdb=# REPLACE INTO test SELECT 2, 22, 22;
REPLACE 0 2

-- Query the result.
gaussdb=# SELECT * FROM test WHERE f1 = 2;
 f1 | f2 | f3
----+----+----
  2 | 22 | 22
(1 row)

-- Replace or insert the specified column.
gaussdb=# REPLACE INTO test SET f1 = f1 + 3, f2 = f1 * 10 + 3, f3 = f2;
REPLACE 0 2

-- Query the result.
gaussdb=# SELECT * FROM test WHERE f1 = 3;
 f1 | f2 | f3
----+----+----
  3 | 33 | 33
(1 row)

-- Drop table test.
gaussdb=# DROP TABLE test;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.18.9 RESET

Description

Restores GUC parameters to their default values. The default values are parameter default values complied in the **gaussdb.conf** configuration file.

RESET is an alternative spelling for:

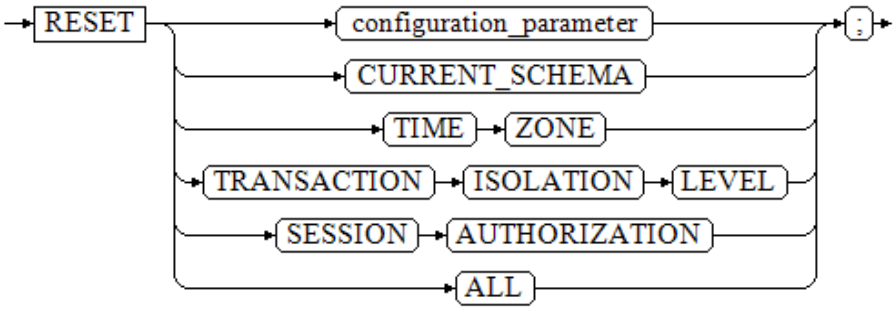
```
SET configuration_parameter TO DEFAULT;
```

Precautions

RESET and SET have the same transaction behavior. Their impact will be rolled back.

Syntax

```
RESET {configuration_parameter | CURRENT_SCHEMA | TIME ZONE | TRANSACTION ISOLATION LEVEL |
SESSION AUTHORIZATION | ALL };
```



Parameters

- **configuration_parameter**

  Specifies the name of a GUC parameter.

  Value range: GUC parameters. You can view them by running the **SHOW ALL** command.

  NOTE

  Some parameters that viewed by running **SHOW ALL** cannot be set by running **SET**, for example, **max_datanodes**.

- **CURRENT_SCHEMA**

  Specifies the current schema.

- **TIME ZONE**

  Specifies the time zone.

- **TRANSACTION ISOLATION LEVEL**

  Specifies the transaction isolation level.

- **SESSION AUTHORIZATION**

  Specifies the session authorization.

- **ALL**

  Resets all settable run-time parameters to default values.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

```
-- Set the time zone to Italy.
gaussdb=# SET timezone TO 'Europe/Rome';
-- View the current time zone.
gaussdb=# SHOW timezone;
  TimeZone
-------------
 Europe/Rome
(1 row)

-- Set the time zone to the default value.
gaussdb=# RESET timezone;
-- View the current time zone.
gaussdb=# SHOW timezone;
 TimeZone
----------
 PRC
(1 row)

-- The preceding SQL statement is equivalent to the following two SQL statements. Set the time zone to the default value.
gaussdb=# SET timezone TO DEFAULT;
gaussdb=# ALTER SESSION SET timezone to DEFAULT;
```

Helpful Links

SET and **SHOW**

1.13.18.10 REVOKE

Description

Revokes permissions from one or more roles.

Precautions

If a non-owner user of an object attempts to REVOKE permission on the object, the statement is executed based on the following rules:

- If the user has no permissions whatsoever on the object, the statement will fail outright.
- If an authorized user has some permissions, only the permissions with authorization options are revoked.
- If the authorized user does not have the authorization option, the REVOKE ALL PRIVILEGES form will issue an error message. For other forms of statements, if the permission specified in the statement does not have the corresponding authorization option, the statement will issue a warning.

Syntax

- Revoke the permission on a specified table or view.

```
REVOKE [ GRANT OPTION FOR ]
    { { SELECT | INSERT | UPDATE | DELETE | TRUNCATE | REFERENCES | ALTER | DROP | COMMENT | INDEX | VACUUM }[, ...]
    | ALL [ PRIVILEGES ] }
    ON { [ TABLE ] table_name [, ...]
        | ALL TABLES IN SCHEMA schema_name [, ...] }
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT ];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Revoke the permission on a specified column in a table.

```
REVOKE [ GRANT OPTION FOR ]
    { {{ SELECT | INSERT | UPDATE | REFERENCES | COMMENT } ( column_name [, ...] )}[, ...]
    | ALL [ PRIVILEGES ] ( column_name [, ...] ) }
    ON [ TABLE ] table_name [, ...]
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT ];
```

- Revoke the permission on a specified sequence.

```
REVOKE [ GRANT OPTION FOR ]
    { { SELECT | UPDATE | ALTER | DROP | COMMENT }[, ...]
    | ALL [ PRIVILEGES ] }
    ON { [ SEQUENCE ] sequence_name [, ...]
       | ALL SEQUENCES IN SCHEMA schema_name [, ...] }
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT ];
```

- Revoke the permission on a specified database.

```
REVOKE [ GRANT OPTION FOR ]
    { { CREATE | CONNECT | TEMPORARY | TEMP | ALTER | DROP | COMMENT } [, ...]
    | ALL [ PRIVILEGES ] }
    ON DATABASE database_name [, ...]
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT ];
```

- Revoke the specified CMK permission.

```
REVOKE [ GRANT OPTION FOR ]
    { { USAGE | DROP } [, ...] | ALL [PRIVILEGES] }
    ON CLIENT_MASTER_KEYS client_master_keys_name [, ...]
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT ];
```

- Revoke the specified CEK permission.

```
REVOKE [ GRANT OPTION FOR ]
    { { USAGE | DROP } [, ...] | ALL [PRIVILEGES]}
    ON COLUMN_ENCRYPTION_KEYS column_encryption_keys_name [, ...]
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT ];
```

- Revoke the permission on a specified directory.

```
REVOKE [ GRANT OPTION FOR ]
    { { READ | WRITE | ALTER | DROP } [, ...] | ALL [ PRIVILEGES ] }
    ON DIRECTORY directory_name [, ...]
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT ];
```

- Revoke the permission on a specified external data source.

```
REVOKE [ GRANT OPTION FOR ]
   { USAGE | ALL [ PRIVILEGES ] }
    ON FOREIGN DATA WRAPPER fdw_name [, ...]
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT ];
```

- Revoke the permission on a specified external server.

```
REVOKE [ GRANT OPTION FOR ]
    { { USAGE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
    ON FOREIGN SERVER server_name [, ...]
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT ];
```

- Revoke the permission on a specified function.

```
REVOKE [ GRANT OPTION FOR ]
    { { EXECUTE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
    ON { FUNCTION {function_name ( [ {[ argmode ] [ arg_name ] arg_type} [, ...] ] )} [, ...]
       | ALL FUNCTIONS IN SCHEMA schema_name [, ...] }
    FROM { [ GROUP ] role_name | PUBLIC } [, ...]
    [ CASCADE | RESTRICT ];
```

- Revoke the permission on a specified stored procedure.

```
REVOKE [ GRANT OPTION FOR ]
    { { EXECUTE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
    ON { PROCEDURE {proc_name ( [ {[ argmode ] [ arg_name ] arg_type} [, ...] ] )} [, ...]
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    | ALL PROCEDURE IN SCHEMA schema_name [, ...] }
  FROM { [ GROUP ] role_name | PUBLIC } [, ...]
  [ CASCADE | RESTRICT ];
```

- Revoke the permission on a specified procedural language.

```
REVOKE [ GRANT OPTION FOR ]
  { USAGE | ALL [ PRIVILEGES ] }
   ON LANGUAGE lang_name [, ...]
   FROM { [ GROUP ] role_name | PUBLIC } [, ...]
   [ CASCADE | RESTRICT ];
```

- Revoke the permission on a specified schema.

```
REVOKE [ GRANT OPTION FOR ]
   { { CREATE | USAGE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
   ON SCHEMA schema_name [, ...]
   FROM { [ GROUP ] role_name | PUBLIC } [, ...]
   [ CASCADE | RESTRICT ];
```

- Revoke the permission on a specified tablespace.

```
REVOKE [ GRANT OPTION FOR ]
   { { CREATE  | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
   ON TABLESPACE tablespace_name [, ...]
   FROM { [ GROUP ] role_name | PUBLIC } [, ...]
   [ CASCADE | RESTRICT ];
```

- Revoke the permission on a specified type.

```
REVOKE [ GRANT OPTION FOR ]
  { { USAGE | ALTER | DROP | COMMENT } [, ...] | ALL [ PRIVILEGES ] }
   ON TYPE type_name [, ...]
   FROM { [ GROUP ] role_name | PUBLIC } [, ...]
   [ CASCADE | RESTRICT ];
```

- Revoke the permission on a specified sub-cluster.

```
REVOKE [ GRANT OPTION FOR ]
   { { CREATE | USAGE | COMPUTE  | ALTER | DROP } [, ...] | ALL [ PRIVILEGES ] }
   ON NODE GROUP group_name [, ...]
   FROM { [ GROUP ] role_name | PUBLIC } [, ...]
   [ CASCADE | RESTRICT ];
```

NOTE

When the CREATE permission on a sub-cluster is revoked, the USAGE and COMPUTE permissions are revoked by default.

- Revoke the permission on a package object.

```
REVOKE [ GRANT OPTION FOR ]
  { { EXECUTE | ALTER | DROP | COMMENT } [, ...] | ALL [PRIVILEGES] }
  ON PACKAGE package_name [, ...]
  FROM {[GROUP] role_name | PUBLIC} [, ...]
  [ CASCADE | RESTRICT ];
```

- Revoke permissions from a role.

```
REVOKE [ ADMIN OPTION FOR ]
   role_name [, ...] FROM role_name [, ...]
   [ CASCADE | RESTRICT ];
```

- Revoke the SYSADMIN permission from a role.

```
REVOKE ALL { PRIVILEGES | PRIVILEGE } FROM role_name;
```

- Revoke the ANY permissions.

```
REVOKE [ ADMIN OPTION FOR ]
 { CREATE ANY TABLE | ALTER ANY TABLE | DROP ANY TABLE | SELECT ANY TABLE | INSERT ANY
TABLE | UPDATE ANY TABLE |
 DELETE ANY TABLE | TRUNCATE ANY TABLE | CREATE ANY SEQUENCE | CREATE ANY INDEX |
CREATE ANY FUNCTION | EXECUTE ANY FUNCTION |
 CREATE ANY PACKAGE | EXECUTE ANY PACKAGE | CREATE ANY TYPE | ALTER ANY TYPE | DROP ANY
TYPE | ALTER ANY SEQUENCE | DROP ANY SEQUENCE |
 SELECT ANY SEQUENCE | ALTER ANY INDEX | DROP ANY INDEX | CREATE ANY SYNONYM | DROP
ANY SYNONYM | CREATE ANY TRIGGER | ALTER ANY TRIGGER |
 DROP ANY TRIGGER
} [, ...]
 FROM [ GROUP ] role_name [, ...];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Revoke the permission on database links.
  ```
  REVOKE { CREATE | ALTER | DROP } [PUBLIC] DATABASE LINK FROM role_name;
  ```
- Revoke the permission on PUBLIC synonyms.
  ```
  REVOKE { CREATE | DROP } PUBLIC SYNONYM FROM role_name;
  ```
  Built-in roles (gs_role_public_synonym_create and gs_role_public_synonym_drop) can also be used to revoke the permission on PUBLIC synonyms.
  - Revoke the permission to create PUBLIC synonyms.
    ```
    REVOKE gs_role_public_synonym_create FROM role_name;
    ```
  - Revoke the permission to delete PUBLIC synonyms.
    ```
    REVOKE gs_role_public_synonym_drop FROM role_name;
    ```

Parameters

The keyword PUBLIC indicates an implicitly defined group that has all roles.

For details about the permission types and parameters, see **Parameters**.

Permissions of a role include the permissions directly granted to the role, permissions inherited from the parent role, and permissions granted to PUBLIC. Therefore, revoking the SELECT permission on an object from PUBLIC users does not necessarily mean that the SELECT permission on the object has been revoked from all roles because the SELECT permission directly granted to roles and inherited from parent roles remains. Similarly, if the SELECT permission is revoked from a user but is not revoked from PUBLIC users, the user can still run the SELECT statement.

If GRANT OPTION FOR is specified, the permission cannot be granted to others, but permission itself is not revoked.

If user A holds the UPDATE permission on a table and the **WITH GRANT OPTION** option and has granted them to user B, the permission that user B holds is called dependent permission. If the permissions or the grant option held by user A is revoked, the dependent permissions still exist. Those dependent permissions are also revoked if CASCADE is specified.

A user can only revoke permissions that were granted directly by that user. For example, if user A has granted permission with grant option (**WITH ADMIN OPTION**) to user B, and user B has in turn granted it to user C, then user A cannot revoke the permission directly from C. However, user A can revoke the grant option held by user B and use CASCADE. In this way, the permission of user C is automatically revoked. For another example, if both user A and user B have granted the same permission to C, A can revoke A's own grant but not B's grant, so C will still effectively have the permission.

If the role executing REVOKE holds permissions indirectly by using more than one role membership path, it is unspecified which containing role will be used to execute the statement. In such cases, you are advised to use SET ROLE to become the specific role, and then execute REVOKE. Failure to do so may lead to deleting permissions not intended to delete, or not deleting any permissions at all.

Examples

- Revoke the permission of role **jerry** from user **joe**.
  ```
  -- Create role jerry.
  gaussdb=# CREATE role jerry PASSWORD 'xxxxxxxxxx';
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Grant the CREATE ANY TABLE permission to jerry and allow the role to grant the permission to
others.
gaussdb=# GRANT create any table TO jerry with admin option;

-- Create user joe and grant the permission of role jerry to it.
gaussdb=#  CREATE user joe PASSWORD 'xxxxxxxxxx';
gaussdb=#  GRANT jerry TO joe;

-- Revoke the permission of role jerry from user joe.
gaussdb=#  REVOKE jerry FROM joe;
```

- Revoke system permissions from user **tom**.

```
-- Create user tom.
gaussdb=# CREATE USER tom PASSWORD 'xxxxxxxxxx';

-- Grant system permissions to user tom.
gaussdb=# GRANT ALL PRIVILEGES TO tom;

-- Revoke system permissions from user tom.
gaussdb=# REVOKE ALL PRIVILEGES FROM tom;
```

- Revoke the SELECT permission on the **reason** table in schema **tpcds** from user **joe**.

```
-- Create schema tpcds.
gaussdb=# CREATE SCHEMA tpcds;

-- Create table reason in schema tpcds.
gaussdb=# CREATE TABLE tpcds.reason (
r_reason_sk         INTEGER      NOT NULL,
r_reason_id         CHAR(16)     NOT NULL,
r_reason_desc       VARCHAR(20)
);

-- Grant the query permission on the table tpcds.reason to joe.
gaussdb=# GRANT select ON tpcds.reason TO joe;

-- Revoke the SELECT permission on the reason table in schema tpcds from user joe.
gaussdb=# REVOKE SELECT ON TABLE tpcds.reason FROM joe;
```

- Revoke the ALTER permission on the fun1() function in schema **tpcds** from user **joe**.

```
-- Create function fun1().
gaussdb=# CREATE or replace FUNCTION tpcds.fun1() RETURN boolean AS
BEGIN
SELECT current_user;
RETURN true;
END;
/
-- Grant the ALTER permission on the fun1() function to user joe.
gaussdb=# GRANT ALTER ON FUNCTION tpcds.fun1() TO joe;

-- Revoke the ALTER permission on the fun1() function in schema tpcds from user joe.
gaussdb=#  REVOKE ALTER ON FUNCTION tpcds.fun1() FROM joe;
```

- Revoke the CONNECT permission on database **testdb** from user **joe**.

```
-- Create database testdb.
gaussdb=# CREATE DATABASE testdb;

-- Grant the CONNECT permission on database testdb to user joe.
gaussdb=# GRANT connect on database testdb TO joe WITH GRANT OPTION;

-- Revoke the CONNECT permission on database testdb from user joe.
gaussdb=# REVOKE CONNECT ON database testdb FROM joe;
```

- Clear data.

```
gaussdb=# DROP TABLE tpcds.reason;
gaussdb=# DROP FUNCTION tpcds.fun1();
gaussdb=# DROP SCHEMA tpcds CASCADE;
gaussdb=# DROP USER joe;
gaussdb=# DROP USER tom;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# REVOKE create any table FROM jerry;
gaussdb=# DROP ROLE jerry;
gaussdb=# DROP DATABASE testdb;
```

Helpful Links

GRANT

1.13.18.11 ROLLBACK

Description

Rolls back the current transaction and backs out all updates in the transaction.

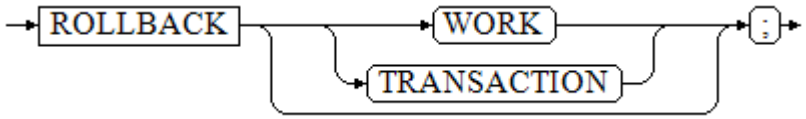
If a fault occurs during the running of a transaction, the transaction cannot be executed. The system cancels all completed operations on the database in the transaction, and the database status returns to the time when the transaction starts.

Precautions

If a ROLLBACK statement is executed out of a transaction, no error occurs, but a notice is displayed.

Syntax

```
ROLLBACK [ WORK | TRANSACTION ];
```



Parameters

WORK | TRANSACTION

Specifies the optional keyword that more clearly illustrates the syntax.

Examples

```
-- Create a table.
gaussdb=# CREATE TABLE test (id int, name text);

-- Start a transaction.
gaussdb=# START TRANSACTION;

-- Alter the table definition.
gaussdb=# ALTER TABLE test ADD COLUMN score int;

-- View the table definition.
gaussdb=# \d test;
     Table "public.test"
 Column |  Type   | Modifiers
--------+---------+-----------
 id     | integer |
 name   | text    |
 score  | integer |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Perform rollback.
gaussdb=# ROLLBACK;

-- The table definition is restored to the initial state.
gaussdb=# \d test;
     Table "public.test"
 Column |  Type   | Modifiers
--------+---------+-----------
 id     | integer |
 name   | text    |


-- Drop the test table.
gaussdb=# DROP TABLE test;
```

Helpful Links

COMMIT | END

1.13.18.12 ROLLBACK PREPARED

Description

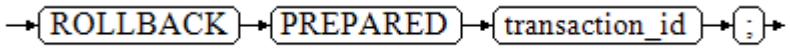
Cancels a transaction ready for two-phase commit.

Precautions

- The function is only available in maintenance mode (when the GUC parameter **xc_maintenance_mode** is set to **on**). Exercise caution when enabling the mode. It is used by maintenance engineers for troubleshooting. Common users should not use the mode.
- Only the user who initiates a transaction or a system administrator can roll back the transaction.
- The transaction function is maintained automatically by the database, and should be not visible to users.

Syntax

```
ROLLBACK PREPARED transaction_id;
```



NOTE

The distributed environment does not support the ROLLBACK PREPARED syntax.

Parameters

transaction_id

Specifies the identifier of the transaction to be committed. The identifier must be different from those for current prepared transactions.

Examples

```
-- Start a transaction in maintenance mode.
gaussdb=# SET XC_MAINTENANCE_MODE = ON;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
SET

gaussdb=# BEGIN;
BEGIN

-- Prepare a transaction whose identifier is trans_test.
gaussdb=# PREPARE TRANSACTION 'trans_test';
PREPARE TRANSACTION

-- Cancel the transaction whose identifier is trans_test.
gaussdb=# ROLLBACK PREPARED 'trans_test';
ROLLBACK PREPARED
```

Helpful Links

COMMIT PREPARED and **PREPARE TRANSACTION**.

1.13.18.13 ROLLBACK TO SAVEPOINT

Description

ROLLBACK TO SAVEPOINT rolls back operations to a specified savepoint. Operations after the savepoint do not take effect.

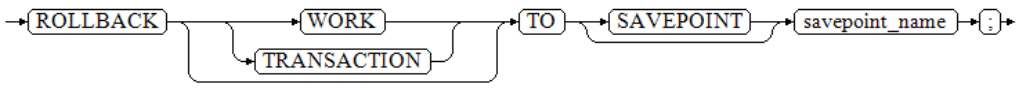
Rolls back all statements that were executed after the savepoint was established. The savepoint remains valid and can be rolled back to again later, if needed.

Precautions

- Rolling back to a savepoint name that has not been specified is an error.
- Cursors have somewhat non-transactional behavior with respect to savepoints. Any cursor that is opened inside a savepoint will be closed when the savepoint is rolled back. If a previously opened cursor is affected by a FETCH statement inside a savepoint that is later rolled back, the cursor remains at the position that FETCH left it pointing to (that is, the cursor motion caused by FETCH is not rolled back). Closing a cursor is not undone by rolling back, either. A cursor whose execution causes a transaction to abort is put in a cannot-execute state, so while the transaction can be restored using ROLLBACK TO SAVEPOINT, the cursor can no longer be used.
- Use ROLLBACK TO SAVEPOINT to roll back to a savepoint. Use RELEASE SAVEPOINT to destroy a savepoint but keep the results of the statements executed after the savepoint was established.

Syntax

```
ROLLBACK [ WORK | TRANSACTION ] TO [ SAVEPOINT ] savepoint_name;
```



Parameters

- **WORK | TRANSACTION**

  Specifies the optional keyword that more clearly illustrates the syntax.
- **savepoint_name**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Rolls back to a savepoint.

Examples

```
-- Start a transaction.
gaussdb=# START TRANSACTION;

-- Set the savepoint my_savepoint.
gaussdb=# SAVEPOINT my_savepoint;

-- Roll back to the savepoint my_savepoint.
gaussdb=# ROLLBACK TO SAVEPOINT my_savepoint;

-- Cursor positions are not affected by savepoint rollback.
gaussdb=# DECLARE foo CURSOR FOR SELECT 1 UNION SELECT 2;

-- Set the savepoint foo.
gaussdb=# SAVEPOINT foo;

-- Read a data record, which is the first one.
gaussdb=# FETCH 1 FROM foo;
 ?column?
----------
        1
-- Roll back to the savepoint foo.
gaussdb=# ROLLBACK TO SAVEPOINT foo;

-- Read a data record, which is the second one.
gaussdb=# FETCH 1 FROM foo;
 ?column?
----------
        2
-- Release a savepoint.
gaussdb=# RELEASE SAVEPOINT my_savepoint;

-- End the transaction.
gaussdb=# COMMIT;
```

Helpful Links

SAVEPOINT and **RELEASE SAVEPOINT**

1.13.19 S

1.13.19.1 SAVEPOINT

Description

SAVEPOINT establishes a new savepoint in the current transaction.

A savepoint is a special mark in a transaction. It allows all commands that are executed after the savepoint was established to be rolled back, restoring the transaction state to what it was at the time of the savepoint.

Precautions

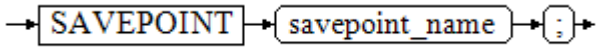
- Use ROLLBACK TO SAVEPOINT to roll back to a savepoint. Use RELEASE SAVEPOINT to destroy a savepoint but keep the effects of the commands executed after the savepoint was established.
- Savepoints can only be established when inside a transaction block. Multiple savepoints can be defined within a transaction.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- In the case of an unexpected termination of a distributed thread or process caused by a node or connection failure, or of an error caused by the inconsistency between source and destination table definitions in a COPY FROM operation, the transaction cannot be rolled back to the established savepoint. Instead, the entire transaction will be rolled back.
- According to the SQL standard, when a savepoint with the same name is created, the previous savepoint with the same name is automatically deleted. In GaussDB, the old savepoint is retained, but only the latest one is used during rollback or release. If the latest savepoint is released, the previous savepoint will again become accessible to ROLLBACK TO SAVEPOINT and RELEASE SAVEPOINT. Except for this, SAVEPOINT fully complies with the SQL standard.

Syntax

```
SAVEPOINT savepoint_name;
```



Parameter

savepoint_name

Specifies the name of the new savepoint.

NOTICE

When using SAVEPOINT, you are advised to release SAVEPOINT promptly to avoid too many nested subtransactions. It is recommended that the number of nested subtransactions be less than or equal to 10,000. If the number of nested subtransactions is too large, the current transaction performance may deteriorate.

Examples

```
-- Create a table.
gaussdb=# CREATE TABLE table1(a int);

-- Start a transaction.
gaussdb=# START TRANSACTION;

-- Insert data.
gaussdb=# INSERT INTO table1 VALUES (1);

-- Create a savepoint.
gaussdb=# SAVEPOINT my_savepoint;

-- Insert data.
gaussdb=# INSERT INTO table1 VALUES (2);

-- Roll back to the savepoint.
gaussdb=# ROLLBACK TO SAVEPOINT my_savepoint;

-- Insert data.
gaussdb=# INSERT INTO table1 VALUES (3);

-- Commit the transaction.
gaussdb=# COMMIT;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Query the table content. The table should contain both 1 and 3 but not 2 because 2 is rolled back.
gaussdb=# SELECT * FROM table1;

-- Drop the table.
gaussdb=# DROP TABLE table1;
```

Helpful Links

RELEASE SAVEPOINT and **ROLLBACK TO SAVEPOINT**

1.13.19.2 SECURITY LABEL ON

Description

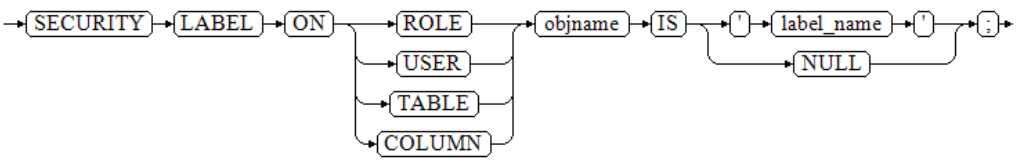
Applies, updates, or cancels a security label.

Precautions

An initial user, a user with the SYSADMIN permission, or a user who inherits the permissions of the built-in role gs_role_seclabel can update or cancel security labels.

Syntax

```
SECURITY LABEL ON { ROLE | USER | TABLE | COLUMN } objname IS {'label_name' | NULL};
```



Parameters

- **objname**
  - For **ROLE** and **USER**, **objname** indicates the user/role name.
  - For **TABLE**, **objname** indicates the table name, which can be prefixed with a schema name.
  - For **COLUMN**, **objname** indicates the name in the format of "table name.column name", which can be prefixed with a schema name.
- **label_name**

  Specifies the security label name.
- **NULL**

  Specifies that the security label is canceled.

Examples

```
-- Create a security label.
gaussdb=# CREATE SECURITY LABEL sec_label 'L1:G4';

-- Create a table.
gaussdb=# CREATE TABLE tbl(c1 int, c2 int);

-- Create a user.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CREATE USER bob WITH PASSWORD '********';

-- Apply a security label to the user.
gaussdb=# SECURITY LABEL ON ROLE bob IS 'sec_label';

-- Apply a security label to the table.
gaussdb=# SECURITY LABEL ON TABLE tbl IS 'sec_label';

-- Apply a security label to a column of the table.
gaussdb=# SECURITY LABEL ON COLUMN tbl.c1 IS 'sec_label';

-- Cancel the security label of the user.
gaussdb=# SECURITY LABEL ON ROLE bob IS NULL;
-- Cancel the security label of the table.
gaussdb=# SECURITY LABEL ON TABLE tbl IS NULL;

-- Cancel the security label of the column of the table.
gaussdb=# SECURITY LABEL ON COLUMN tbl.c1 IS NULL;

-- Delete table tb1.
gaussdb=# DROP TABLE tbl;

-- Delete user bob.
gaussdb=# DROP USER bob;

-- Delete the existing security label sec_label.
gaussdb=# DROP SECURITY LABEL sec_label;
```

Helpful Links

CREATE SECURITY LABEL and **DROP SECURITY LABEL**

1.13.19.3 SELECT

Description

Queries data from a table or view. The SELECT operation is akin to a filter layered over a database table. By specifying query criteria, you can sift through the database table to extract the desired data.

Precautions

- The owner of a table, users with the SELECT permission on the table, and users with the SELECT ANY TABLE permission can perform SELECT to read data in the table or view. When separation of duties is disabled, system administrators have this permission by default.
- SELECT supports JOIN between ordinary tables but does not support JOIN between ordinary tables and GDS foreign tables. That is, ordinary tables and GDS foreign tables cannot coexist in the same SELECT statement.
- Users performing a SELECT operation must have SELECT permission on the columns included in the SELECT statement.
- Only users with both SELECT and UPDATE permissions on a table can perform the FOR UPDATE or FOR SHARE operation on the table.
- Currently, the statement specifying a partition cannot perform global index scan.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax

Query data.

```
[ WITH [ RECURSIVE ] with_query [, ...] ]
SELECT [/*+ plan_hint */] [ ALL | DISTINCT [ ON ( expression [, ...] ) ] ]
   { * | {expression [ [ AS ] output_name ]} [, ...] }
   [ FROM from_item [, ...] ]
   [ WHERE condition ]
   [ [ START WITH condition ] CONNECT BY [NOCYCLE]
   condition [ ORDER SIBLINGS BY expression ] ]
   [ GROUP BY grouping_element [, ...] ]
   [ HAVING condition [, ...] ]
   [ WINDOW {window_name AS ( window_definition )} [, ...] ]
   [ { UNION | INTERSECT | EXCEPT | MINUS } [ ALL | DISTINCT ] select ]
   [ ORDER BY {expression [ [ ASC | DESC | USING operator ] | nlssort_expression_clause ] [ NULLS { FIRST |
LAST } ]} [, ...] ]
   [ LIMIT { [offset,] count | ALL } ]
   [ OFFSET start [ ROW | ROWS ] ]
   [ FETCH { FIRST | NEXT } [ count ] { ROW | ROWS } ONLY [OFFSET offset_count]]
   [ {FOR { UPDATE | SHARE } [ OF table_name [, ...] ] [ NOWAIT | WAIT n | SKIP LOCKED ]} [...] ];
```

NOTICE

- When the SELECT statement plan is cached, the number of conditions in the WHERE IN candidate subset should be less than or equal to 100 to prevent high dynamic memory.
  - If the WHERE IN candidate subset is too large, the memory usage of the generated plan increases.
  - If the WHERE IN subsets constructed by concatenated SQL statements are different, the SQL template of the cache plan cannot be reused. A large number of different plans are generated, and the plans cannot be shared. As a result, a large amount of memory is occupied.
- In condition and expression, you can use the aliases of expressions in the **targetlist** (the list returned by the query) in compliance with the following rules:
  - Reference only within the same level.
  - Only reference aliases in **targetlist**.
  - Reference a prior expression in a subsequent expression.
  - Volatile functions cannot be used.
  - Window functions cannot be used.
  - Aliases cannot be referenced in the condition of JOIN ON.
  - An error is reported if the target list contains multiple aliases to be referenced.

- The subquery **with_query** is as follows:

  ```
  with_query_name [ ( column_name [, ...] ) ]
      AS [ [ NOT ] MATERIALIZED ] ( {select | values | insert | update | delete} )
  ```

- The specified query source **from_item** is as follows:

  ```
  {[ ONLY ] table_name [ * ] [ partition_clause ] [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
  [ TABLESAMPLE sampling_method ( argument [, ...] ) [ REPEATABLE ( seed ) ] ]
  [ TIMECAPSULE {TIMESTAMP | CSN} expression ]
  |( select ) [ AS ] alias [ ( column_alias [, ...] ) ]
  |with_query_name [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
  |function_name ( [ argument [, ...] ] ) [ AS ] alias [ ( column_alias [, ...] | column_definition [, ...] ) ]
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
|function_name ( [ argument [, ...] ] ) AS ( column_definition [, ...] )
|xmltable_clause
|from_item unpivot_clause
|from_item pivot_clause
|from_item [ NATURAL ] join_type from_item [ ON join_condition | USING ( join_column [, ...] ) ]}
```

  - Specify partition_clause in the query source **from_item** as follows:

    ```
    {PARTITION { ( { partition_name | subpartition_name } [, ...] ) | FOR (  partition_value [, ...] ) }
    | SUBPARTITION { ( subpartition_name ) | FOR (  subpartition_value [, ...] )}}
    ```

    NOTE

    - The specified partition applies only to partitioned tables.
    - If PARTITION specifies multiple partition names, the partition names can be the same. The union set of the partition ranges is used.

  - Specify xmltable_clause in the query source **from_item** as follows:

    ```
    XMLTABLE(
    xmlnamespaces_clause
    row_expression
    passing_clause
    columns_clause
    )
    ```

    - For xmlnamespaces_clause:

      ```
      {
         XMLNAMESPACES(
         {string AS identifier
         |
         DEFAULT string }
         [, { string AS identifier  |  DEFAULT string } ][...]
         ),
      }
      ```

    - For passing_clause:

      ```
      [
         PASSING [BY { REF | VALUE }] document_expression [BY { REF | VALUE }]
      ]
      ```

    - For columns_clause:

      ```
      [
         COLUMNS
         name { type [PATH column_expression] [DEFAULT default_expression] [NOT NULL |
      NULL] | FOR ORDINALITY }
         [, ...]
      ]
      ```

- The GROUP BY clause is as follows:

  ```
  {()
  |expression
  |( expression [, ...] )
  |ROLLUP ( { expression | ( expression [, ...] ) } [, ...] )
  |CUBE ( { expression | ( expression [, ...] ) } [, ...] )
  |GROUPING SETS ( grouping_element [, ...] )}
  ```

- The sorting order nlssort_expression_clause is as follows:

  ```
  NLSSORT ( column_name, ' NLS_SORT = { SCHINESE_PINYIN_M | generic_m_ci } ' )
  ```

- Simplified query syntax, equivalent to SELECT * FROM table_name.

  ```
  TABLE { ONLY {(table_name)| table_name} | table_name [ * ]};
  ```

Parameters

- **WITH [RECURSIVE] with_query [, ...]**

  Specifies one or more subqueries that can be referenced by name in the main query, which is equivalent to a temporary table. This subquery statement structure is called the common table expression (CTE) structure. When this

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

structure is used, the execution plan contains the CTE SCAN content. If RECURSIVE is declared, the SELECT subquery can reference itself by name.

Format of **with_query**:

```
with_query_name [ ( column_name [, ...] ) ] AS [ [ NOT ] MATERIALIZED ] ( {select | values | insert | update | delete} )
```

- **with_query_name** specifies the name of the result set generated by a subquery. Such names can be used to access the result sets of subqueries in a query.
- **column_name** specifies the column name displayed in the subquery result set.
- Each subquery can be a SELECT, VALUES, INSERT, UPDATE, or DELETE statement.
- To use the CTE structure, RECURSIVE must be declared after WITH. If there are multiple CTEs, you only need to declare RECURSIVE at the first CTE.
- You can use [NOT] MATERIALIZED to modify the CTE. Currently, only inline execution is supported for stream plans. In this case, this syntax does not take effect.
  - If **MATERIALIZED** is specified, the WITH query will be materialized, and a copy of the subquery result set is generated. The copy is directly queried at the reference point. Therefore, the WITH subquery cannot be jointly optimized with the SELECT statement trunk (for example, predicate pushdown and equivalence class transfer). When using **NOT MATERIALIZED** for modification, if the WITH query can be semantically executed as a subquery inline, it can be jointly optimized with the trunk SELECT statement (such as predicate pushdown and equivalence class transfer).
  - If the user does not explicitly declare the materialized attribute, comply with the following rules: If the CTE is referenced only once in the SELECT statement trunk to which it belongs and semantically supports inline execution, it will be rewritten as subquery inline execution. Otherwise, the materialized execution will be performed in CTE SCAN mode.

- **plan_hint clause**

  Optimizes the plan generated by the SELECT statement block. The hint is declared in the /*+ */ format after the SELECT keyword. For details about how to use hints for optimization, see "Hint-based Optimization" in *SQL Optimization Guide*. In each statement, only the first **/*+** *plan_hint* ***/** comment block takes effect as a hint. Multiple hints can be written in the comment block.

- **ALL**

  (Optional) Returns all rows that meet the SELECT query conditions. This is the default behavior of SELECT query.

- **DISTINCT [ON (expression [, ...])]**

  Removes all duplicate rows from the result set of the SELECT statement so that each row in the result set is unique.

  ON (expression[, ...]): retains only the data from the first row of the result set that complies with the expression operation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2022

NOTICE

- Specifying **DISTINCT ON** retains only the first row of the set of rows that comply with the expression and does not sort the set, so the result is not fixed. To return a unique row as expected, you are advised to also specify **ORDER BY** for data filtering.
- The DISTINCT ON expression does not support sorting of non-integer constants.

- **SELECT list**

  Specifies the name of a column in the table to be queried. The value can be a part of the column name or all of the column names. The wildcard (*) is used to represent the column name.

  You may use the AS *output_name* clause to assign an alias to the output column. The alias is usually used for displaying the output column.

  Column names can be expressed in the following ways:

  – Manually input column names which are separated by commas (,)
  – Use the results or columns calculated by the FROM clause as column names.
  – Use the keywords NAME, VALUE, and TYPE as column aliases.

- **FROM clause**

  Specifies one or more data source tables for a SELECT query.

  The FROM clause contains the following elements:

  – table_name

    Specifies the name of a table or view. The schema name can be added before the table name or view name as a limit, for example, schema_name.table_name.

    NOTE

    You can use database links to perform operations on remote tables and synonyms. For details, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.

  – alias

    Assigns a temporary alias to a table or a complex table reference so that the reference of the table by other queries is easier.

    An alias is used for easier reference or to eliminate ambiguity for self-joins. If an alias is provided, it can be referenced directly instead of using the actual table name.

    NOTE

    To add an alias to **joined_table** generated by the JOIN operation and specified by parentheses in the format of **(joined_table)**, do not use non-reserved keywords UNPIVOT and PIVOT as the table alias.

  – TABLESAMPLE sampling_method (argument [, ...]) [REPEATABLE (seed)]

    Specifies a sampling method to retrieve a subset of rows in a table specified by **table_name**. The available sampling methods are as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2023

- SYSTEM(percent)

  Random sampling by data block. The value range of **percent** is [0,100], indicating the probability (in percentage) that each data block is selected.

- BERNOULLI(percent)

  Independent random sampling by row. The value range of **percent** is [0,100], indicating the probability (in percentage) that each row is selected.

- HYBRID(p1,p2)

  Hybrid random sampling. It consists of two steps: First, random sampling is performed on data blocks with probability **p1**. Second, independent random sampling by row is performed on the sampled data content with probability **p2**. The value range of both **p1** and **p2** is [0,100].

The REPEATABLE clause is optional and serves as an additional condition for **sampling_method**, and **seed** is the seed value used for generating random numbers within the sampling method, which can be any non-null constant value. When the table definition/data remains unchanged and multiple sampling operations are performed, if **REPEATABLE (seed)** is not specified, the system acquires a fresh random sample for each query using the automatically generated seed value, so the sample is not fixed. Once **REPEATABLE (seed)** is specified, a unique and fixed sample can be obtained.

– TIMECAPSULE { TIMESTAMP | CSN } expression

  Specifies a CSN or time point as a flashback query identifier where the table content is queried.

  - TIMECAPSULE TIMESTAMP

    Searches for the result set of a specified time point based on the date. *date* must be a valid past timestamp.

  - TIMECAPSULE CSN

    Queries the result set for a specified CSN in a table flashback. The CSN can be obtained from **snpcsn** recorded in gs_txn_snapshot.

  - expression

    Constants, functions, or SQL expressions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2024

NOTE

- Currently, flashback query cannot be performed on system catalogs, DFS tables, global temporary tables, local temporary tables, unlogged tables, views, sequence tables, shared tables, and inherited tables.
- After a statement (TRUNCATE, DDL, DCL, or VACUUM FULL) that modifies the table definition or affects physical storage of a table is executed between the flashback point and the current point, a flashback query cannot be performed on the table again because such attempt will fail. If a flashback is performed on tables that have undergone DDL operations, the error message "ERROR: The table definition of %s has been changed." is displayed. Similarly, when a flashback is performed on DDL operations involving table definition changes, the error message "ERROR: recycle object %s desired does not exist." is displayed.
- Flashback query supports scanning PCR UB-tree indexes. If no such indexes, flashback query supports only seq scan for full table scan.
- When the flashback point is too old, the source version cannot be obtained because the flashback version is recycled. As a result, the flashback fails and the error message "Restore point too old" is displayed.
- The flashback point is specified by TIMESTAMP. The maximum difference between the flashback point and the actual time is 3 seconds.
- After TRUNCATE is performed on a table, a flashback query or table flashback operation performed using TIMESTAMP will result in an error (Snapshot too old) ; while a flashback operation by using CSN will fail to find data or report an error (Snapshot too old).
- In the GTM-free scenario, each node uses the local CSN and does not have a globally unified CSN. Therefore, flashback by using CSN is not supported.

– column_alias

  Specifies the column alias.

– PARTITION

  Queries data in the specified partition in a partitioned table.

– partition_name

  Specifies the name of a partition.

– partition_value

  Specifies the value of the specified partition key. If a partitioned table has multiple partition keys, you can use PARTITION FOR to uniquely identify a partition.

– subquery

  Creates a temporary table to store the subquery's output.

– with_query_name

  It can be referenced by the name of the WITH query.

– function_name

  Function name.

– join_type

  ▪ [INNER] JOIN

    Inner join (equi-join). It returns only the rows that match the join condition in the join table.

  ▪ LEFT [OUTER] JOIN

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Left (outer) join. Two tables are concatenated, with the left table serving as the main table. All main table data is displayed, while child table data is shown depending on whether the join conditions are met. Unmatched data will be replaced with **NULL** values.

- RIGHT [OUTER] JOIN

  Right (outer) join. Two tables are concatenated, with the right table serving as the main table. All main table data is displayed, while child table data is shown depending on whether the join conditions are met. Unmatched data will be replaced with **NULL** values.

  You can convert the right (outer) join to a left (outer) join by exchanging the inputs on the left and right.

- FULL [OUTER] JOIN

  Returns all result rows of an inner join and all rows that do not match the join condition in the join table.

- CROSS JOIN

  CROSS JOIN is equivalent to INNER JOIN ON(TRUE), that is, rows that are not deleted by the conditions. These join types are just a notational convenience, since they do nothing you could not do with plain FROM and WHERE.

NOTE

You must declare a join condition (**ON join_condition** or **USING (join_column [,...])**) for INNER JOIN and OUTER JOIN, while it is not required for CROSS JOIN.

CROSS JOIN and INNER JOIN produce a simple Cartesian product, the same result as you get from listing the two items at the top level of FROM.

– ON join_condition

  Join condition, which specifies the matching rule of the joined objects, for example, **ON left_table.a = right_table.a**. You are advised not to use numeric types such as int for **join_condition** because such types can be implicitly converted to bool values (non-zero values are implicitly converted to **true** and **0** is implicitly converted to **false**), which may cause unexpected results.

– USING(join_column[, ...])

  ON left_table.a = right_table.a AND left_table.b = right_table.b ... abbreviation. Corresponding columns must have the same name.

– NATURAL

  Shorthand for a USING list that contains all columns in the two tables that have the same names.

– from item

  Specifies the name of the query source object connected.

– xmltable_clause

  Currently, xmltable_clause can be used only in ORA- or PG-compatible mode. xmltable_clause produces a virtual table based on data of the given XMLType (in ORA-compatible mode) or XML (in PG-compatible mode) type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- xmlnamespaces_clause

  (Optional) It consists of XMLNAMESPACES followed by comma-separated XML namespace declaration clause. **string** indicates the full name of the namespace for which single quotation marks are used as boundary characters. **identifier** indicates the namespace alias for which double quotation marks are used as boundary characters. The alias can be used in row_expression and columns_clause. In the current version, DEFAULT cannot be used to declare the default namespace.

- row_expression

  row_expression is an XPath 1.0 expression with English single quotation marks as boundary characters. The document_expression in passing_clause is used as its context to obtain a group of XML nodes. These XML nodes are processed by columns_clause to generate each row of the virtual table.

  In ORA-compatible mode, **row_expression** can pass a sequence consisting of numbers or XML text, with single quotation marks (') as boundary characters and commas (,) as separators.

- passing_clause

  passing_clause starts with PASSING and its document_expression is the data of the XMLType (in ORA-compatible mode) or XML (in PG-compatible mode) type that is passed to XMLTABLE for processing. Currently, only single data records are supported. In ORA-compatible mode, an error is reported regardless of whether BY VALUE or BY REF is used. In PG-compatible mode, although BY VALUE or BY REF in the first and second places can be identified and received, no functional processing is performed.

- columns_clause

  (Optional) Specifies the information required for the columns to be generated in the virtual table, as shown in **Table 1-405**.

Table 1-405 COLUMNS information description

Information	Description
name	Column name.
type	Column type.
PATH	(Optional) column_expression is an expression of XPath 1.0. A node in the node set obtained by row_expression is used as the context of the node, and the data required by the value of the corresponding column in the row data generated by the node is obtained through processing by the node. There is an implicit conversion when it is converted to a result of the TYPE type. If the **PATH** part is not specified, **name** is regarded as column_expression.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Information	Description
DEFAULT	(Optional) If the value obtained from column_expression processing is **NULL**, the result obtained from calculating default_expression is used to replace **NULL**. default_expression is calculated only when necessary. If the expression is fixed, it is calculated only once.
NOT NULL | NULL	(Optional) In PG-compatible mode, specifies whether the column can be **NULL**. An error is reported if the column value is not the specified option.
FOR ORDINALITY	Indicates that the column is a row number column. Row numbers starting from 1 are filled for generated rows. Only one column can be marked as **FOR ORDINALITY**.

In ORA-compatible mode, columns_clause can be omitted. After columns_clause is omitted, **COLUMNS column_value XMLTYPE PATH '.'** is generated by default for subsequent processing.

In PG-compatible mode, columns_clause cannot be omitted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

Currently, xmltable_clause has the following restrictions:

- The involved XPath expressions are XPath 1.0.
- Currently, XPath expressions such as **"namespace::*"** cannot be used in **row_expression**.
- Currently, XPath expressions such as **".."** cannot be used in **column_expression**.
- When the data obtained through column_expression is converted to the data of the TYPE type, in ORA-compatible mode, the data exceeds the typmod of the type will be truncated. Currently, this truncation occurs only when the typmod of the CHAR, CHARACTER, NCHAR, BPCHAR, VARCHAR, CHARACTER VARYING, VARCHAR2, and NVARCHAR2 data types, including those with (n), is greater than 0. In PG-compatible mode, an error is reported when the data exceeds the typmod of the type.
- Currently, the maximum size of the input XMLType (in ORA-compatible mode) or XML (in PG-compatible mode) type data is 1 GB.
- For upgrade from a version that does not support XMLTABLE to a version that supports XMLTABLE, the XMLTABLE syntax cannot be used during the upgrade observation period.
- Only ORA-compatible mode supports "*:nodename" within XPath, which means the namespace of *nodename* is ignored when selected.
- The non-reserved keyword XMLTABLE cannot be used as the name of a function of the Functions as Table Sources type.
- In ORA-compatible mode, when the input is a sequence consisting of numbers and XML text, the restrictions are as follows:
  - It cannot be used together with **passing_clause** or **columns_clause**.
  - The number of elements separated by commas (,) in the sequence ranges from 1 to 1024.
  - Expressions cannot be used as sequence elements.
  - The ampersands (>), quotation marks ("), and ampersands (&) in XML texts are not escaped to ">", """, and "&", respectively.
  - The XML text node name can contain only one or more spaces between the node name and the greater-than sign (>), and the output result does not delete the spaces between the node name and the greater-than sign (>).
  - This feature is not supported during the upgrade observation period.
- When the returning sequence of the xmlquery is nested in the xmltable, the XPath in **column_clause** can only be a period ('.').
- The period ('.') in the XPath cannot be followed by an index expression, for example, '.[6]', or a predicate expression, for example, '.[.>5]'.
- When ill-formed XMLType data in ORA-compatible mode is entered, spaces and carriage return characters used to control the writing format between nodes after the ill-formed part are parsed as text nodes, for example, the spaces and carriage return characters between the end tag of **node2** and the start tag of **node3** are parsed as text nodes.

  Input:

```
<root>
<node1>node1</node1>
malform
<node2>node2</node2>
<node3>node3</node3>
</root>
```

  Actual situation:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
<root><node1>node1</node1>
malform
<node2>node2</node2>
<node3>node3</node3>
</root>
```

Example:

```
-- Create an ORA-compatible database.
gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA' ENCODING = 'utf8';
gaussdb=# \c ora_db
-- A sequence consisting of numbers and XML text is supported.
ora_db=# SELECT * FROM XMLTABLE('1, 2, <a>12</a>');
 column_value
--------------
 1
 2
 <a>12</a>
(3 rows)

ora_db=# \c postgres
gaussdb=# DROP DATABASE ora_db;
```

– unpivot_clause

Converts a column to a row.

```
UNPIVOT [ {INCLUDE | EXCLUDE} NULLS ]
   (
      unpivot_col_clause
      unpivot_for_clause
      unpivot_in_clause
   )
```

- {INCLUDE | EXCLUDE} NULLS

  (Optional) Specifies whether the transposition result contains rows with **NULL** values. If this clause is not specified, the unpivot operation filters rows with **NULL** values from the result set by default.

  INCLUDE NULLS: retains the rows with **NULL** values in the result set.

  EXCLUDE NULLS: filters rows with **NULL** values from the result set.

- unpivot_col_clause

  ```
  unpivot_col_element
  ```

  unpivot_col_element specifies the output column names. It stores the values of the columns to be transposed.

- unpivot_col_element

  ```
  { column | ( column [, column]... ) }
  ```

- unpivot_for_clause

  ```
  FOR { unpivot_col_element }
  ```

  unpivot_col_element specifies the output column names. It stores the alias and names of the column to be transposed.

- unpivot_in_clause

  ```
  IN ( unpivot_in_element [,unpivot_in_element...] )
  ```

  unpivot_in_clause specifies the columns to be transposed. The column names and values are stored in unpivot_col_element.

  unpivot_in_element

  ```
  { unpivot_col_element }[  AS { unpivot_alias_element } ]
  ```

  unpivot_col_element specifies the column to be transposed. If (column [, column]...) is used to specify the column to be

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

transposed, the *column* names are connected by underscores (_) and saved in the output columns. For example, IN ((col1, col2)) generates the column name **col1_col2** and saves it in the output column specified by unpivot_for_clause. The AS keyword can be used to specify an alias for the column to be transposed. After an alias is specified, the alias is saved in the output column instead of the name of column to be transposed.

- unpivot_alias_element

```
{ alias | ( alias [, alias]... )}
```

alias indicates the specified alias.

NOTE

Currently, unpivot_clause has the following restrictions:

- This parameter can be used only in ORA-compatible databases.
- The unpivot_clause clause cannot be used with hints.
- For unpivot_col_clause, the number of output columns specified by unpivot_col_element must be the same as that of unpivot_col_element contained in unpivot_in_clause.
- For unpivot_for_clause, the number of output columns specified by unpivot_col_element must be the same as the number of aliases specified by unpivot_alias_element contained in unpivot_in_clause.
- For unpivot_in_clause, the alias must be a constant or an expression that can be converted to a constant.
- For **unpivot_in_clause**, constant expressions support only IMMUTABLE functions.
- For all unpivot_col_element parameters contained in unpivot_in_clause, if the column types in the same position of these unpivot_col_element parameters are different, UNPIVOT attempts to convert the column types in order to convert the values of columns to be transposed to a public type. Similarly, for all unpivot_alias_element parameters, if the alias types of these unpivot_alias_element parameters in the same position are different, UNPIVOT attempts to do the similar type conversion.

– pivot_clause

Converts a row into a column.

```
PIVOT [ XML ]
( aggregate_function ( expr ) [[AS] alias ]
    [, aggregate_function ( expr ) [[AS] alias ] ]...
    pivot_for_clause
    pivot_in_clause
)
```

- aggregate_function (expr) [[AS] alias]

  Aggregates calculation on a given expression. The calculation result is saved in the output column specified by pivot_in_clause.

  [AS] alias specifies an alias for aggregate_function. The alias is recorded in the format of _alias after the output column name specified by pivot_in_clause.

- pivot_for_clause

```
FOR { column
    | ( column [, column]... )
    }
```

pivot_for_clause specifies the row to be transposed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

column: represents the column to which the row will be transposed.

- pivot_in_clause

```
IN ( { { { expr
        | ( expr [, expr]... )
        } [ [AS] alias]
     }...
   }
 )
```

pivot_in_clause specifies the name of the output column. The column name can consist of one or more expressions. When a column name consists of multiple expressions, these expressions are connected by underscores (_) in sequence. That is, the output column name corresponding to (expr1, expr2) is "expr1_expr2". These expressions not only generate output column names, but also determine the time when the aggregate function is triggered. If the values of row to be transposed is the same as the value of these expressions, the results calculated by aggregate_function are saved in the output column names that consist of these expressions.

NOTE

Currently, pivot_clause has the following restrictions:

- This parameter can be used only in ORA-compatible mode.
- The pivot_clause clause cannot be used with hints.
- If more than one aggregate_function is specified, at most one aggregate_function is allowed to have no alias, the rest of the aggregate_function functions are required to specify an alias.
- XML supports only syntax but does not support functions.
- The expression in pivot_in_clause can be a constant or an expression that can be converted to a constant. If the expression is not a unary expression, specify an alias for the expression.
- For pivot_in_clause, constant expressions support only IMMUTABLE functions.
- When an alias is specified for expressions in pivot_in_clause through AS, non-reserved keywords can be used as aliases; otherwise, non-reserved keywords cannot be used as aliases.
- If the length of an output column name exceeds 63 characters, an error is reported.

- **WHERE clause**

  The WHERE clause forms an expression for row selection to narrow down the query range of SELECT.

  condition indicates any expression that returns a value of Boolean type. Only rows that meet this condition will be displayed. Do not use numeric types such as int for **condition** because such types can be implicitly converted to bool values (non-zero values are implicitly converted to **true** and **0** is implicitly converted to **false**), which may cause unexpected results.

  In a WHERE clause, you can use the operator (+) to convert a table join to an outer join. This operation does not comply with standard SQL syntax. During platform migration, syntax compatibility issues may occur. You are advised not to perform this operation. Using (+) simultaneously comes with the following restrictions:

  – (+) can appear only in the WHERE clause.
  – If the table join relationship has been specified in the FROM clause, then (+) cannot be used in the WHERE clause.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The operator (+) can work only on columns of tables or views, instead of on expressions.
- If tables have multiple join conditions, (+) must be specified in all the conditions. Otherwise, (+) will not take effect, and the table join will be converted into an inner join without displaying any prompt information.
- Tables specified in a join condition where the operator (+) works cannot cross queries or subqueries. If tables where (+) works are not in the FROM clause of the current query or subquery, an error will be reported. If a peer table for the operator (+) does not exist, no error will be reported and the table join will be converted into an inner join.
- Except for the outer join expressions of the same single table, expressions where (+) is used cannot be directly connected through the OR operator.
- If a column where the operator (+) works is compared with a constant, the expression becomes a part of the JOIN condition.
- When a table is used as the outer table for multiple tables, if there is no join relationship between the corresponding inner tables, a Cartesian product is performed between the inner tables, and then an outer join is done between the inner tables and the outer table.
- "(+)" can appear only in the following expressions: comparison, BETWEEN AND/NOT BETWEEN AND, NOT, ANY/SOME, ALL, IN/NOT IN, NULLIF, IS DISTINCT FROM/IS NOT DISTINCT FROM, IS NULL/IS NOT NULL, or IS OF. (+) is not allowed in other types of expressions. In addition, these expressions cannot be connected through AND or OR.
- The operator (+) can be used to convert a table join only to a left or right outer join, instead of a full join. That is, the operator (+) cannot be specified on both tables of an expression.
- In ORA-compatible mode, spaces are allowed within the parentheses of (+), for example, "(+)" and "(+)".
- The "(+)" operator does not support automatic parameterization.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- For the WHERE clause, if special character "%", "_", or "\" is queried in LIKE, add the slash "\" before each character. In ORA-compatible mode, when the GUC parameter **standard_conforming_strings** is set to **on**, you can set the **behavior_compat_options** parameter to **ignore_like_default_escape** to use the backslash (\) as a common character for matching.

```
gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA';
gaussdb=# \c ora_db;
ora_db=# SET standard_conforming_strings = on;
ora_db=# SET behavior_compat_options = ignore_like_default_escape;
ora_db=# SELECT 'aa\a' LIKE 'aa\a';
 ?column?
----------
 t
(1 row)
ora_db=# SELECT 'aaa' LIKE 'aa\a';
 ?column?
----------
 f
(1 row)
ora_db=# SET behavior_compat_options = '';
ora_db=# SELECT 'aa\a' LIKE 'aa\a';
 ?column?
----------
 f
(1 row)
ora_db=# SELECT 'aaa' LIKE 'aa\a';
 ?column?
----------
 t
(1 row)
ora_db=# \c postgres;
gaussdb=# DROP DATABASE ora_db;
```

- If the IN clause after a WHERE clause has no following parentheses, it supports only one element to be added after IN. The restriction only applies in ORA-compatible mode. In addition, the WHERE clause can contain only this IN condition. The element has the following restrictions:
  - Constant: such as character strings, integers, and floating points
  - Column name
  - CASE WHEN clause
  - Function
- The "(+)" operator supports validity check after the common expression is extracted. A common expression is defined as an expression with the same structure, including the sequence within the expression and the sequence number of placeholders within the expression.

```
SELECT * FROM a,a2 WHERE a.id(+) IS NULL AND a.id(+) = a2.id OR a.id(+) = a2.id AND a.id(+) =
1;
```

  After the common expression a.id (+) = a2.id is extracted, it can be converted into:

```
SELECT * FROM a,a2 WHERE a.id(+) = a2.id AND (a.id(+) IS NULL OR a.id(+) = 1);
```

  The converted expression meets the requirement that the left and right sides of the OR operator are outer join expressions of the same single table. Therefore, the verification is successful.

```
SELECT * FROM a,a2 WHERE a.id(+) IS NULL AND a2.id = a.id(+) OR a.id(+) = a2.id AND a.id(+) =
1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The expression sequences of a2.id = a.id(+) and a.id(+) = a2.id are different, and the common expression cannot be extracted. Therefore, the verification fails.

When using placeholders to bind parameters in a PREPARE statement or dynamically executing a query statement through a cursor, ensure that the placeholder sequence numbers in the common expression are the same.

```
PREPARE p1 AS SELECT * FROM a,a2 WHERE a.id IS NULL AND a.id(+) = $1 OR a.id(+) = $2 AND a.id = 1;
```

The placeholder sequence numbers of a.id(+) = $1 and a.id(+) = $2 are different ($1≠$2), which does not comply with the definition of the common expression. Therefore, the verification fails.

- **START WITH clause**

  The START WITH clause is usually used together with the CONNECT BY clause and indicates the initial condition of recursion: Data is traversed recursively and hierarchically. If this clause is omitted and the CONNECT BY clause is used alone, all rows in the table are used as the initial set. For details, see **CONNECT BY**.

- **CONNECT BY clause**

  CONNECT BY indicates a recursive join condition. It is used together with START WITH to implement data traversal and recursion.

  In the CONNECT BY condition, the PRIOR keyword can be specified for a column to indicate that the column is recursive. If NOCYCLE is added before the recursive join condition, recursion stops when a cyclic record is encountered.

NOTICE

A SELECT statement containing the START WITH *condition* CONNECT BY clause does not support the FOR SHARE or UPDATE lock.

The process of executing the START WITH statement is as follows:

a. The initial data set is selected based on the condition in the START WITH clause. For example, select ('A', 1, 0) first, and then set the initial data set as the working set.

b. If the working set is not empty, the data in the working set is used as the input for the next query. The filter criteria are specified in the CONNECT BY clause. The keyword PRIOR indicates the current record. For example, **prior id = fatherid** in the example indicates that the ID of the current record is **fatherid** of the next record.

c. Set the data set filtered out in **the previous step** as the working set and repeat **the previous step**.

```
gaussdb=#  CREATE TABLE test(name varchar, id int, fatherid int);
gaussdb=#  INSERT INTO test VALUES('A', 1, 0), ('B', 2, 1),('C',3,1),('D',4,1),('E',5,2);
gaussdb=#  SELECT * FROM test START WITH id = 1 CONNECT BY prior id = fatherid ORDER SIBLINGS BY id DESC;
 name | id | fatherid
------+----+----------
 A    | 1 |        0
 D    | 4 |        1
 C    | 3 |        1
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 B    | 2 |       1
 E    | 5 |       2
(5 rows)
```

Currently, the default behavior of START WITH is width-first search. However, depth-first search can be implemented by working with **pseudocolumns**. Example:

```
gaussdb=# SELECT sys_connect_by_path(name,'-') as path, *, LEVEL FROM test START WITH id = 1
CONNECT BY fatherid=prior id ORDER BY path;
  path   | name | id | fatherid | level
---------+------+----+----------+-------
 -A      | A    |  1 |        0 |     1
 -A-B    | B    |  2 |        1 |     2
 -A-B-E  | E    |  5 |        2 |     3
 -A-C    | C    |  3 |        1 |     2
 -A-D    | D    |  4 |        1 |     2
gaussdb=# DROP TABLE test;
```

If a cycle exists in the data set, the database provides cycle detection. By default, an error is reported and no data is returned when a cycle is detected. If the NOCYCLE keyword is specified, the query can be executed normally, and recursion stops and exits when a cyclic record is encountered.

– Pseudocolumns: The database adds pseudocolumns to each selected data record so that users can learn about the position of the data in the recursive or tree structure.

  - **LEVEL**: level of a node. The level of the root node is **1**.
  - **CONNECT_BY_ISCYCLE**: returns whether the current node forms a cycle with its parent or child row.
  - **CONNECT_BY_ISLEAF**: specifies whether a node is a leaf node. If a node does not have any subnode, the node is a leaf node and **CONNECT_BY_ISLEAF** is set to **1**. Otherwise, it is set to **0**.

  For example (Assume that table **T1** already exists with the data structure as shown in **Table 1-406**):

```
SELECT * FROM T1 CONNECT BY PRIOR B=A AND LEVEL<=3;
```

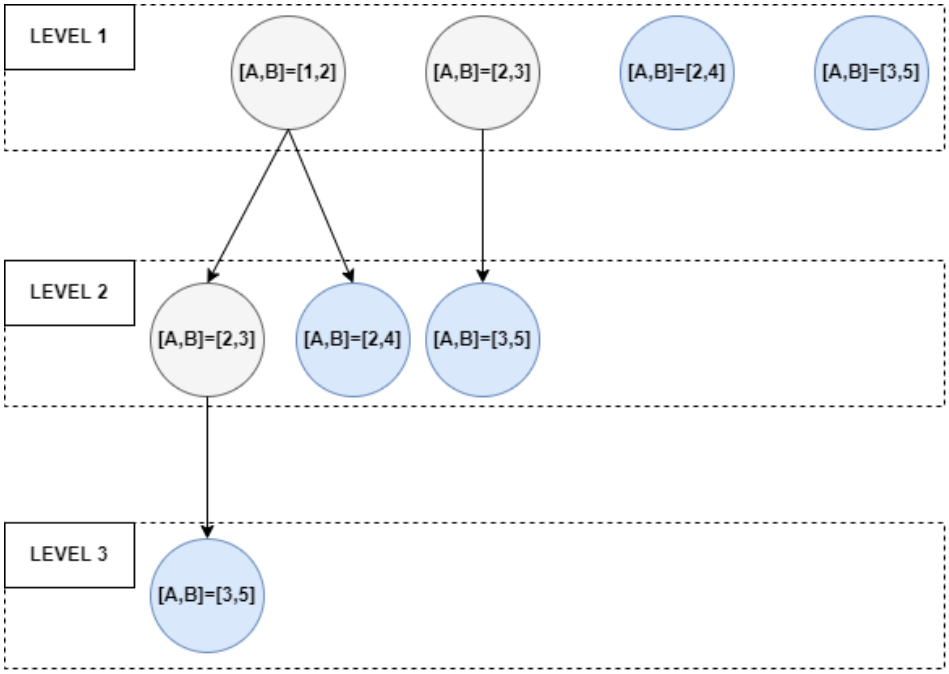
Table 1-406 T1 data structure

A	B
1	2
2	3
2	4
3	5

The tree structure of the result is shown in **Figure 1-15**, where the blue nodes are leaf nodes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 1-15 Logical structure of the execution



- Query functions: In addition to pseudocolumns, the database also provides query functions (for details, see **Hierarchical Recursion Query Functions**).
  - **sys_connect_by_path(col, separator)**: returns the connection path from the root node to the current row. **col** indicates the name of the column displayed in the path, and **separator** indicates the connector.
  - **connect_by_root(col)**: displays the top-level node of the node. **col** indicates the name of the output column.
- In addition to pseudocolumns and query functions, the PRIOR keyword can be used outside the CONNECT BY clause to modify columns. It can be used in the SELECT query column, DISTINCT clause, WHERE clause, GROUP BY clause, HAVING clause, WINDOW clause, and ORDER BY clause to display the parent row column value.
- In the hierarchical query, the search is performed strictly according to the depth-first order. If **ROWNUM** is used as the filter condition in START WITH or CONNECT, the value of **ROWNUM** is increased by 1 for each record to be returned. Then, the record is verified based on **ROWNUM** conditions. Records that do not satisfy the conditions are discarded and the value of **ROWNUM** is decreased by 1.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- The PRIOR keyword cannot be specified in the START WITH statement.
- In addition to **targetlist**, "prior (single column)" is parsed as "prior single column". User-defined functions named **prior** are not perceived.
- PRIOR can be specified only for columns in the table. It cannot be specified for expressions, pseudocolumns, or type conversions.
- In the CONNECT BY statement, columns modified by PRIOR cannot be in the same condition as pseudocolumns like level and rownum (such as **PRIOR a = level**), but can be in different conditions (such as **PRIOR a = b and level = 1**). Different conditions refer to the conditions joined by AND at the top layer of the CONNECT BY statement.
- In the START WITH and CONNECT BY clauses, pseudocolumns cannot be used for subconnection, for example, "rownum = (subquery)" or "rownum in (subquery)".
- You are advised not to use ROWNUM in the CONNECT BY clause. If ROWNUM is required, test it thoroughly to avoid unexpected results.
- When START WITH or CONNECT BY are called on the CTE defined by WITH AS, if there are multiple CTEs, ensure that the definition of each CTE does not depend on other CTEs.
- If no cycle exists in the data but the error message "runs into cycle" is reported, increase the value of **max_recursive_times**.
- **connect_by_isleaf**, **connect_by_iscycle** and **level** are of the int type.
- In hierarchical queries, **connect_by_isleaf**, **connect_by_iscycle**, and **level** will be parsed as pseudocolumns and will not be parsed as columns whose alias names in projection columns are the same as pseudocolumns.
- You are advised not to reference other views when defining views with hierarchical queries.
- Optimization suggestions for the START WITH clause:
  - Create indexes based on the conditions in the CONNECT BY clause to improve the performance of the START WITH clause.
  - Identify bottlenecks based on the plan collected by running EXPLAIN PERFORMANCE or in WDR. If the recursive operator (inner plan) of RECURSIVE UNION is the Hash JOIN operator, but the hash table is created for the temporary table **tmp_result** or the hash table in plan is materialized (that is, the batch size is greater than 1), the possible cause is that the value of **work_mem** is too small. As a result, the hash table cannot be created for the outer data table. You can increase the value of **work_mem** to improve performance.

    GaussDB optimizes tables with a small volume of data and caches table results in hash tables to improve performance. In this case, indexes are not required. However, if the data volume exceeds the limit specified by **work_mem**, the optimization becomes invalid. In this case, you can create indexes for optimization.
- If no value or **NULL** is used in the START WITH statement, the initial recursive condition is empty and an empty result is returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If no value is used in the CONNECT BY statement, the recursive condition is empty and all results that meet the condition of the START WITH clause are returned.
- PRIOR-qualified columns support only the following types: INT8, INT1, INT2, OID, INT4, BOOL, CHAR, NAME, FLOAT4, FLOAT8, ABSTIME, RELTIME, DATE, DATEA (only effective in ORA-compatible mode), CASH, TIME, TIMESTAMP, TIMESTAMPTZ, SMALLDATETIME, UUID, INTERVAL, TIMETZ, INT2VECTOR, CLOB, NVARCHAR2, VARCHAR, TEXT, VECTOR, BPCHAR, RAW, BYTEA, NUMERIC, XID, CID, and TID. The operation of forcibly converting a type to any aforementioned type cannot avoid the whitelist restriction, for example, by running **PRIOR col::text** (where col is not a type in the aforementioned whitelist).
- When the PRIOR keyword is used to modify a column outside the CONNECT BY clause, the statement behavior is displayed based on the value of the **enable_startwith_prior_enhance** parameter. If this parameter is disabled, using the PRIOR keyword to modify a column is equivalent to not adding the PRIOR keyword. In this case, the current row and column values are displayed. If this parameter is enabled and the PRIOR keyword is used to modify a column in a clause (SELECT query column, DISTINCT clause, WHERE clause, GROUP BY clause, HAVING clause, WINDOW clause, or ORDER BY clause), the parent row column values are displayed. In addition, an error is reported in the following scenarios:
  - In a clause (SELECT query column, DISTINCT clause, WHERE clause, GROUP BY clause, HAVING clause, WINDOW clause, or ORDER BY clause) that supports PRIOR, PRIOR is used to modify the pseudo columns **LEVEL**, **CONNECT_BY_ISCYCLE**, and **CONNECT_BY_ISLEAF**.
  - In the SELECT statement with the CONNECT BY clause, PRIOR is used to modify columns in the ON condition and ORDER SIBLINGS BY clause.
  - In the statement without the CONNECT BY clause, the PRIOR keyword is used to modify columns.
- If PRIOR is used to modify different columns for multiple times in the same query statement, the data is considered duplicate only when the values of all modified columns are the same. That is, a cycle exists.
- If the **level**, **connect_by_isleaf**, or **connect_by_iscycle** column is used as the input parameter of the hierarchical recursive query function sys_connect_by_path or connect_by_root, it will be parsed as a pseudocolumn and an error will be reported.

- ORDER SIBLINGS BY clause

  The output of the START WITH statement is returned level by level. However, within each layer, the database automatically selects the optimal execution path, so the implementation sequence cannot be guaranteed. If you have requirements on the final output sequence, you can use ORDER SIBLINGS BY. The usage of ORDER SIBLINGS BY is the same as that of ORDER BY. ORDER SIBLINGS BY is used for sorting at each level during recursion.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

The expression following ORDER SIBLINGS BY only supports sorting by calling non-aggregate and window functions for ordinary columns, column name offsets, and column names. It does not support calling system functions related to START WITH for column names or using pseudocolumns related to START WITH.

- **GROUP BY clause**

  Condenses query results into a single row all selected rows that share the same values for the grouped expressions.

  – CUBE ({ expression | (expression [, ...]) } [, ...])

    A CUBE grouping is an extension to the GROUP BY clause that creates subtotals for all of the possible combinations of the given list of grouping columns (or expressions). In terms of multidimensional analysis, CUBE generates all the subtotals that could be calculated for a data cube with the specified dimensions. For example, given three expressions (n=3) in the CUBE clause, the operation results in $2^n = 2^3 = 8$ groupings. Rows grouped on the values of *n* expressions are called regular rows, and the rest are called superaggregate rows.

  – GROUPING SETS (grouping_element [, ...])

    GROUPING SETS is an advanced extension of the GROUP BY clause. It allows you to explicitly define multiple GROUP BY options in a single query. Unlike CUBE or ROLLUP, GROUPING SETS precisely specifies the required grouping sets, and the database does not need to execute the complete aggregation set generated by CUBE or ROLLUP, thereby significantly improving query efficiency.

    NOTE

    When the GROUPING SETS clause is used, an error may be reported if the SQL query statement contains logical statements such as DISTINCT ON, ORDER BY, or multi-table equi-joins. You are advised to encapsulate the calculation logic of the GROUPING SETS clause into the inner subquery so that the grouping set calculation can be executed independently first.

    ```
    SELECT DISTINCT ON (a) a, b FROM t1 GROUP BY GROUPING SETS ((a, b), (b)) ORDER BY a, b DESC;
    ```

    It can be rewritten as:

    ```
    SELECT DISTINCT ON (a) a, b FROM (SELECT a, b FROM t1 GROUP BY GROUPING SETS ((a, b), (b))) AS sub_t ORDER BY a, b DESC;
    ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- If **sql_mode** contains the **ONLY_FULL_GROUP_BY** option, the following situations occur:
  - If the non-aggregate function column in the SELECT list is consistent with the column for GROUP BY, no error is reported.
  - If the non-aggregate function column in the SELECT list is inconsistent with the column for GROUP BY, the following situations may occur:
    - If the GROUP BY list contains a primary key or a unique non-null key, the SQL statement does not report an error.
    - Non-aggregate function columns in the SELECT list are also included in the GROUP BY or WHERE list. If included in the WHERE list, the columns in the WHERE clause must be in the form of a constant, and then no error will be reported.
    - If both the JOIN and GROUP BY clauses are contained, the following situations may occur:
      - LEFT JOIN

        In the syntax, when the ON clause is used for equality check between two columns, no error is reported if the GROUP BY clause contains columns from the LEFT JOIN table.

        For example, when LEFT JOIN operations are performed on tables **t1** and **t2**, if the condition in the ON clause is **t1.col1=t2.col2 ...**, the GROUP BY clause only needs to contain **t1.col1**.
      - RIGHT JOIN

        In the syntax, when the ON clause is used for equality check between two columns, no error is reported if the GROUP BY clause contains columns from the RIGHT JOIN table.

        For example, when RIGHT JOIN operations are performed on tables **t1** and **t2**, if the condition in the ON clause is **t1.col1=t2.col2 ...**, the GROUP BY clause only needs to contain **t2.col2**.
      - INNER JOIN/CROSS JOIN/STRAIGHT_JOIN

        In the syntax, when the ON clause is used for equality check between two columns, no error is reported if the GROUP BY clause contains either of the columns.
- If **sql_mode** does not contain the **ONLY_FULL_GROUP_BY** option, no restriction exists and no error will occur in the preceding error scenarios.
  - If SELECT contains both non-aggregate function columns and aggregate functions, it must be used together with GROUP BY. Otherwise, it will result in an error.
  - If a non-aggregate function column beyond the columns for GROUP BY contains multiple data records after it is grouped, due to differences in execution plans and DN data distribution, the queried column data is a random data record in the grouped data.
- The GROUP BY clause does not support sorting of non-integer constants.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **HAVING clause**

  Used together with the GROUP BY clause to select special groups. The HAVING clause compares some attributes of groups with a constant. Only groups that match the logical expression in the HAVING clause are displayed.

- **WINDOW clause**

  ```
  WINDOW window_name AS ( window_definition ) [, ...]
  ```

  – **window_name**: name referenced by **window_definition**.

  – window_definition:

  ```
  [ existing_window_name ]
  [ PARTITION BY expression [, ...] ]
  [ ORDER BY expression [ ASC | DESC | USING operator ] [ NULLS { FIRST | LAST } ] [, ...] ]
  [ frame_clause ]
  ```

  frame_clause: defines a window frame for window functions. Window functions (not all) depend on the window frame.

  window frame: indicates a group of related rows of the current query row.

  The format of frame_clause is as follows:

  ```
  [ RANGE | ROWS ] frame_start
  [ RANGE | ROWS ] BETWEEN frame_start AND frame_end
  ```

  Formats of **frame_start** and **frame_end**:

  ```
  UNBOUNDED PRECEDING
  VALUE PRECEDING
  CURRENT ROW
  VALUE FOLLOWING
  UNBOUNDED FOLLOWING
  ```

  NOTE

  Restrictions on scenarios where VALUE is used:

  - The table columns operated by the aggregate function or window function support only the INT1, INT2, INT4, INT8, FLOAT4, FLOAT8, NUMERIC, INTERVAL, TIME, TIMETZ, TIMESTAMP, TIMESTAMPTZ, DATEA (only effective in ORA-compatible mode), and DATE types.
  - The VALUE type supports only the INT1, INT2, INT4, INT8, INT16, FLOAT4, FLOAT8, NUMERIC, INTERVAL, and character string (only character strings that can be implicitly converted to numbers are supported) types.
  - The value of **VALUE** can only be a non-negative number.
  - Currently, **VALUE** only supports constants instead of variables and expressions as the upper and lower boundaries.

- **UNION clause**

  ```
  select_statement UNION [ALL] select_statement
  ```

  – **select_statement**: SELECT statement that does not contain the ORDER BY, LIMIT, or FOR UPDATE clause.

  – UNION operator: calculates the union of multiple row sets returned by **select_statement**.

  – **ALL**: optional. If it is not specified, the UNION result excludes duplicate rows.

  The UNION clause has the following restrictions:

  – The number of columns in multiple SELECT statements must be the same, and the data type and sequence of each column must be the same.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Multiple UNION operators in the same SELECT statement have a default priority of executing from left to right. To alter the priority, use parentheses to specify the order.
  - FOR UPDATE cannot be specified either for a UNION result or for any input of a UNION.
  - A UNION subexpression can contain an ORDER BY or LIMIT clause enclosed in parentheses.

NOTICE

- It is recommended that the number of UNION ALL clauses be less than 100. If the number of UNION ALL clauses exceeds 100, ensure that the instance memory is sufficient.
- The following constraints apply when you perform set operations with **NULL**:
  - **NULL** can only be a projection column itself or a projection column in a subquery of the FROM statement.
  - In CTEs and recursive queries, complex subqueries (DISTINCT, GROUP BY, ORDER BY, and PARTITION BY statements) are not supported.

- **INTERSECT clause**

```
select_statement INTERSECT select_statement
```

  - **select_statement**: SELECT statement that does not contain the FOR UPDATE clause.
  - INTERSECT operator: calculates the intersection of multiple row sets returned by **select_statement** and deduplicates rows.

  The INTERSECT clause has the following restrictions:

  - Multiple INTERSECT operators in the same SELECT statement have a default priority of executing from left to right. To alter the priority, use parentheses to specify the order.
  - For UNION and INTERSECT operations on the execution results of multiple SELECT statements, INTERSECT has a higher priority than UNION.

- **EXCEPT clause**

```
select_statement EXCEPT [ ALL ] select_statement
```

  - **select_statement**: SELECT expressions that do not contain the FOR UPDATE clause.
  - The EXCEPT operator calculates the rows that are in the output of the left **select_statement** but not in the output of the right **select_statement**. Multiple EXCEPT operators in the same SELECT statement have a default priority of executing from left to right. To alter the priority, use parentheses to specify the order.
  - **ALL**: (Optional) If **ALL** is not specified, the result of EXCEPT does not contain any duplicate rows; if it is specified, a row with *m* duplicate rows in the left table and *n* duplicate rows in the right table appears max(*m* – *n*,0) times in the result.

  Currently, the FOR UPDATE clause cannot be declared for the results of EXCEPT or for any inputs of EXCEPT.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **MINUS clause**

  Has the same function and syntax as EXCEPT clause.

- **ORDER BY clause**

  Sorts data retrieved by SELECT in ascending or descending order. The ORDER BY clause does not support sorting of non-integer constants.

  - If an ORDER BY expression contains multiple columns, the columns are sorted based on the following behavior:
    - The expression is executed from left to right. The leftmost column is sorted first. If two columns are equal according to the leftmost expression, they are compared according to the next expression.
    - If they are equal according to all specified expressions, they are returned in an implementation-dependent order.
    - When used with the DISTINCT keyword, the columns to be sorted in ORDER BY must be included in the columns of the result set retrieved by the SELECT statement.
    - When used with the GROUP BY clause, the columns to be sorted in ORDER BY must be included in the columns of the result set retrieved by the SELECT statement.
  - When the clause is not used with the GROUP BY clause and the columns of the result set retrieved by the SELECT statement contain aggregate functions:
    - The ORDER BY clause is ignored if the columns of the retrieved result set do not contain any aggregate functions.
    - Only the sorting of the columns that contain aggregate functions is retained if the columns of the retrieved result set contain aggregate functions.
  - In an ORA-compatible database, when the **DISTINCT** keyword is specified, the columns or expressions referenced in the ORDER BY clause must be in the DISTINCT list.
    - The expression range supported by the ORDER BY clause must comply with the constraints of the ORDER BY clause.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

1. ORDER BY supports only operations on constants and entire expressions. It may not guarantee normal operations on constants and non-entire expressions.

```
-- Supported.
-- It is recommended that an entire expression be enclosed by parentheses for the operation on a constant and the entire expression.
SELECT DISTINCT c1*c2+c2*c1 FROM t1 ORDER BY c2*c1+c1*c2+1;
SELECT DISTINCT c1*c2+c2*c1 FROM t1 ORDER BY (c2*c1+c1*c2)+1;
SELECT DISTINCT c1*c2+c2*c1 FROM t1 ORDER BY 1+(c2*c1+c1*c2);

-- Not supported.
-- In terms of the operation on a constant and a non-entire expression, the position of the constant affects the structure verification logic of the expression, which may affect the normal operation.
SELECT DISTINCT c1*c2+c2*c1 FROM t1 ORDER BY 1+c2*c1+c1*c2;
SELECT DISTINCT c1*c2+c2*c1 FROM t1 ORDER BY c2*c1+1+c1*c2;
```

2. If the ORDER BY and DISTINCT expressions contain multiple forcible type conversions, the database parses and converts the forcibly converted expressions. If the parsed expression of ORDER BY is in the parsed list of DISTINCT, it passes the verification. Otherwise, it fails the verification. If the DISTINCT expression is inconsistent with the ORDER BY expression, the parsed results may be inconsistent.

```
-- Not supported.
-- The DISTINCT expression is inconsistent with the original expression of ORDER BY, and the parsed results are also inconsistent.
SELECT DISTINCT c1::int::char::int,c2::integer::int3::int4 FROM t1 ORDER BY c2::char::int + c1;

-- Supported.
-- The DISTINCT expression is inconsistent with the original expression of ORDER BY, but the parsed results are consistent.
SELECT DISTINCT c1::int::char::int,c2::integer::int3::int4 FROM t1 ORDER BY c2::char::int + c1::char::int;
SELECT DISTINCT c1::int::int::char::int::int,c2::integer::int3::int4 FROM t1 ORDER BY c2::char::int + c1::int::char::int;
SELECT DISTINCT c1::int::int::char::char::char::int::int,c2::integer::int3::int4 FROM t1 ORDER BY c2::char::int + c1::int::char::int;
```

3. The addition and multiplication operations in functions do not support swapping.

```
SELECT DISTINCT BIT_LENGTH(c1+c2) FROM t1 ORDER BY BIT_LENGTH(c2+c1);
```

4. If both DISTINCT and ORDER BY expressions contain function aliases, only aliases whose function attributes are STABLE and IMMUTABLE are supported.

```
-- Not supported.
-- The function attribute is VOLATILE.
SELECT DISTINCT DBE_RANDOM.GET_VALUE(c1+c2,100) AS c0 FROM t1 ORDER BY c0+1;

-- Supported.
-- The function attribute is STABLE.
SELECT DISTINCT add_numbers(c1,c2) AS c0 FROM t1 ORDER BY c0+1;
```

5. If ORDER BY is followed by only a constant, the constant indicates the position of a DISTINCT expression and must meet the limitations of ORDER BY.

```
-- Supported.
-- 1 indicates the first expression in the DISTINCT list.
SELECT DISTINCT c1,c2 FROM test_table ORDER BY 1;

-- Not supported.
-- 2 indicates the second expression in the DISTINCT list. Since the DISTINCT list contains only one expression, the system reports the error: "The expression at position 2 does not exist."
SELECT DISTINCT 1 FROM test_table ORDER BY 2;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Addition and multiplication operations only support swapping when left and right expressions are consistent.

  NOTE

  To support Chinese pinyin order, set the encoding format to **UTF-8**, **GB18030**, **GB18030_2022**, **GBK**, or **ZHS16GBK** when initializing the database. The statements are as follows:

  ```
  initdb –E UTF8 –D ../data –locale=zh_CN.UTF-8
  initdb -E GB18030 -D ../data -locale=zh_CN.GB18030
  initdb -E GB18030_2022 -D ../data -locale=zh_CN.GB18030
  initdb –E GBK –D ../data –locale=zh_CN.GBK
  initdb –E ZHS16GBK –D ../data –locale=zh_CN.GBK
  ```

- **LIMIT clause**

  Consists of two independent sub-clauses:

  LIMIT { count | ALL } limits the number of rows to be returned. **count** specifies the number of rows to be returned. The effect of **LIMIT ALL** is the same as that of omitting the LIMIT clause.

  OFFSET start count specifies the maximum number of rows to be returned. **count** specifies the number of rows, and **start** specifies the number of rows to be ignored before the first row is returned. If both parameters are specified, the **count** of returned rows starts being calculated after the **start** row.

  ROWNUM cannot be used as count or offset in the LIMIT clause.

- **OFFSET clause**

  The SQL: 2008 standard has introduced a different clause:

  ```
  OFFSET start { ROW | ROWS }
  ```

  start specifies the number of rows to be ignored before starting to return rows.

- **FETCH { FIRST | NEXT } [count] { ROW | ROWS } ONLY [OFFSET offset_count]**

  If **count** is omitted in a **FETCH** clause, it defaults to **1**.

- **FOR UPDATE | SHARE clause**
  - FOR UPDATE: adds an exclusive lock to the rows retrieved by SELECT to prevent the retrieved rows from being modified or deleted by other transactions before the current transaction ends. This blocks other UPDATE, DELETE, and SELECT FOR UPDATE operations until the current transaction ends.
  - FOR SHARE: adds a shared lock to the rows retrieved by SELECT to prevent them from being modified or deleted by other transactions before the current transaction ends. This blocks other UPDATE, DELETE, and SELECT operations (except for SELECT FOR SHARE), until the current transaction ends.
  - **[OF table_name [, ...]]**: object to which the lock is added. If a table name is explicitly specified, only the specified table is locked, while others used in SELECT remain unlocked; otherwise, all tables employed in the command will be locked. If applied to a view or subquery, it locks all specified tables used in the view or subquery.

  To prevent the operation from waiting for the commit of other transactions, you can set the following options:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **NOWAIT**: If the selected row cannot be locked immediately, an error is reported immediately after **SELECT FOR UPDATE NOWAIT** is executed.
- **WAIT** *n*: If the selected row cannot be locked immediately, wait for *n* seconds (*n* is of the int type and the value range is $0 \leq n \leq 2147483$). If the lock is obtained within *n* seconds, the operation is performed normally. Otherwise, an error is reported.
- **SKIP LOCKED**: skips locked rows when adding locks to a table. **SKIP LOCKED** skips only row locks and does not skip scenarios where locks do not block each other in different transactions, such as in scenario of SELECT FOR SHARE - SELECT FOR SHARE SKIP LOCKED.

NOTICE

- Multiple FOR UPDATE and FOR SHARE clauses can be written if it is necessary to specify different locking behaviors for different tables.
- If the same table is mentioned (or implicitly affected) by both FOR UPDATE and FOR SHARE clauses, it is processed as FOR UPDATE. Similarly, a table is processed as **NOWAIT** if that is specified in any of the clauses affecting it.
- For SQL statements containing FOR UPDATE or FOR SHARE, their execution plans will be pushed down to DNs. If the pushdown fails, an error will be reported.
- The query of row number contained in the projection column or WHERE conditions does not support FOR UPDATE or FOR SHARE.
- For the FOR UPDATE or FOR SHARE operation whose subquery is a stream plan, the same locked row cannot be concurrently updated.
- For the ORDER BY FOR UPDATE or ORDER BY FOR SHARE operation, the execution order of the sort operator and lock operator in the stream plan differs from that in other plans. In the stream plan, the lock operation is performed before the sort operation. In other plans, the sort operation is performed before the lock operation. The reason is that if the data in the stream plan is not on the current DN, the data needs to be redistributed and locked on the original DN. After redistribution, data becomes disordered. Therefore, the SORT operator needs to be added. If SORT is performed before LOCK, the original ordered data becomes disordered again. In this case, the SORT operator is meaningless and can be eliminated. The final plan execution sequence is changed from **sort > lock > sort** to **lock > sort**.
- When FOR UPDATE or FOR SHARE operations are performed concurrently, adding ORDER BY to sort data can help prevent issues like no response. However, this approach is inappropriate in distributed systems because the locking order of DNs cannot be determined by ORDER BY, and specifying ORDER BY also leads to performance overhead. You are advised not to use ORDER BY to address this issue.
- The DATABASE LINK function does not support the SKIP LOCKED syntax.
- In distributed scenarios, FOR UPDATE/SHARE cannot be performed on multiple tables.
- The hierarchical query statement cannot be used together with this clause.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- NLS_SORT

  Sorts data according to the specified mode.

  Value range:

  - **SCHINESE_PINYIN_M**, sorted by Pinyin order.
  - **generic_m_ci**: sorted in case-insensitive order (optional; only English characters are supported in the case-insensitive order.)

  NOTICE

  Currently, only Chinese Pinyin and case-insensitive sorting are supported. When creating a database, you need to specify the encoding format as **UTF8**, **GB18030**, **GB18030_2022**, **GBK**, or **ZHS16GBK**. If you specify another encoding format, such as **SQL_ASCII**, an error may occur or the sorting may be invalid.

- **PARTITION clause**

  Queries data in a partition of a partitioned table.

1.13.19.3.1 Simple Queries

A simple query retrieves one or more columns of data from one or more tables or views.

Example:

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE student(
    sid INT PRIMARY KEY,
    class INT,
    name  VARCHAR(50),
    gender INT CHECK(gender = 0 OR gender = 1) -- Gender. 1: male; 0: female
);
gaussdb=# INSERT INTO student (sid, class, name, gender) VALUES (1, 1, 'michael', 0);
gaussdb=# INSERT INTO student (sid, class, name, gender) VALUES (2, 2, 'bob', 1);
gaussdb=# INSERT INTO student (sid, class, name, gender) VALUES (3, 2, 'gary', 0);

-- Query some columns.
gaussdb=# SELECT sid, name FROM student;
 sid |  name
-----+---------
   1 | michael
   2 | bob
   3 | gary
(3 rows)

-- Query all columns.
gaussdb=# SELECT * FROM student;
 sid | class |  name   | gender
-----+-------+---------+-----
   1 |     1 | michael |   0
   2 |     2 | bob     |   1
   3 |     2 | gary    |   0
(3 rows)

-- Specify the column alias.
gaussdb=# SELECT sid student_id, name FROM student;
 student_id |  name
------------+---------
          1 | michael
          2 | bob
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
        3 | gary
(3 rows)

-- Drop the table.
gaussdb=# DROP TABLE student;
```

1.13.19.3.2 Conditional Queries

A conditional query retrieves data that meets the conditions.

Example:

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE test_grade(
    id INT,
    name VARCHAR(20),
    score FLOAT
);
gaussdb=# INSERT INTO test_grade VALUES (1,'Scott',90),(2,'Jack',87.5),(3,'Ben',48);

-- Query information about students who fail to pass the test.
gaussdb=# SELECT * FROM test_grade WHERE score < 60;
 id | name | score
----+------+-------
  3 | Ben  |    48
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE test_grade;
```

1.13.19.3.3 Group Queries

A group query is usually used together with aggregate functions to query category statistics. Common aggregate functions include count(), sum(), avg(), min(), and max(). GROUP BY is often used in group queries.

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE student (id INT,name VARCHAR(20),class INT);
gaussdb=# INSERT INTO student VALUES (1, 'Scott', 1), (2, 'Ben', 1);
gaussdb=# INSERT INTO student VALUES (3, 'Jack', 2),(4, 'Anna', 2),(5, 'Judy', 2);
gaussdb=# INSERT INTO student VALUES (6, 'Sally', 3), (7, 'Jordan', 3);

-- Use the aggregate function COUNT() to count the number of people in each class.
gaussdb=# SELECT class, count(*) AS cnt FROM student GROUP BY class;
 class | cnt
-------+-----
     2 |   3
     1 |   2
     3 |   2
(3 rows)

-- Use the HAVING clause to retrieve data of classes with fewer than three people.
gaussdb=# SELECT class, COUNT(*) AS num FROM student GROUP BY class HAVING COUNT(*) < 3;
 class | num
-------+-----
     1 |   2
     3 |   2
(2 rows)

-- Drop the table.
gaussdb=# DROP TABLE student;
```

sql_mode contains the **ONLY_FULL_GROUP_BY** option, which is used to verify the GROUP BY list and SELECT list.

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE test (
        id int  primary key NOT NULL,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
        id_2 int unique NOT NULL,
        s1 varchar(10),
        s2 int,
        s3 varchar(12)
      );
gaussdb=# INSERT INTO test value(1,101,'a',1000,'aa');
gaussdb=# INSERT INTO test value(2,102,'b',2000,'aaa');
gaussdb=# INSERT INTO test value(3,103,'c',3000,'aaaa');
gaussdb=# INSERT INTO test value(4,104,'b',4000,'aaa');
gaussdb=# INSERT INTO test value(5,105,'c',5000,'aaaaa');
gaussdb=# CREATE TABLE tb1(a int, b int, c int);
gaussdb=# INSERT INTO tb1 VALUES(1, 4, 11);
gaussdb=# INSERT INTO tb1 VALUES(2, 5, 12);
gaussdb=# INSERT INTO tb1 VALUES(2, 6, 13);
gaussdb=# INSERT INTO tb1 VALUES(2, 2, 13);
gaussdb=# CREATE TABLE tb2(a int, b int);
gaussdb=# INSERT INTO tb2 VALUES(2, 7);
gaussdb=# INSERT INTO tb2 VALUES(2, 8);
gaussdb=# INSERT INTO tb2 VALUES(3, 9);

-- Set <b>sql_mode</b>.
gaussdb=# SET sql_mode =
'ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_
BY_ZERO,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION';

-- If the non-aggregate function column in the SELECT list is consistent with the column for GROUP BY, no
error is reported.
gaussdb=# SELECT s1, s3 FROM test GROUP BY s1,s3 ORDER BY s1,s3;
 s1 |  s3
----+-------
 a  | aa
 b  | aaa
 c  | aaaa
 c  | aaaaa
(4 rows)

gaussdb=# SELECT s1, s3 FROM test GROUP BY s3 ORDER BY s3;
ERROR:  Column "test.s1" must be used in the GROUP BY clause or aggregate function.
LINE 1: SELECT s1, s3 FROM test GROUP BY s3 ORDER BY s3;
               ^
-- If the non-aggregate function column in the SELECT list is inconsistent with the column for GROUP BY,
the following two cases exist:
-- Situation 1: If the GROUP BY list contains a primary key or a unique not-null key, the SQL statement does
not report an error.
gaussdb=# SELECT s1, MAX(LENGTH(s3)) FROM test GROUP BY id ORDER BY id;
 s1 | max
----+-----
 a  |   2
 b  |   3
 c  |   4
 b  |   3
 c  |   5
(5 rows)

-- Situation 2: All non-aggregate function columns in the SELECT list are displayed in the GROUP BY or
WHERE list. Columns in the WHERE clause must be equal to a constant.
gaussdb=# SELECT s1, SUM(s2) FROM test WHERE s1 = 'b' GROUP BY s1;
 s1 | sum
----+------
 b  | 6000
(1 row)

-- Disable ONLY_FULL_GROUP_BY in sql_mode.
gaussdb=# SET SQL_MODE =
'STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_AUTO_CR
EATE_USER,NO_ENGINE_SUBSTITUTION';

-- If the non-aggregate function column in the SELECT list is consistent with the column for GROUP BY, no
error is reported.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT s1, s3 FROM test GROUP BY s1,s3 ORDER BY s1,s3;
 s1 |  s3
----+-------
 a  | aa
 b  | aaa
 c  | aaaa
 c  | aaaaa
(4 rows)

gaussdb=# SELECT s1, s3 FROM test GROUP BY s3 ORDER BY s3;
 s1 |  s3
----+-------
 a  | aa
 b  | aaa
 c  | aaaa
 c  | aaaaa
(4 rows)

-- If the non-aggregate function column in the SELECT list is inconsistent with the column for GROUP BY, the following two cases exist:
-- Situation 1: If the GROUP BY list contains a primary key or a unique not-null key, the SQL statement does not report an error.
gaussdb=# SELECT s1, MAX(LENGTH(s3)) FROM test GROUP BY id ORDER BY id;
 s1 | max
----+-----
 a  |   2
 b  |   3
 c  |   4
 b  |   3
 c  |   5
(5 rows)

-- Situation 2: All non-aggregate function columns in the SELECT list are displayed in the GROUP BY or WHERE list. Columns in the WHERE clause must be equal to a constant.
gaussdb=# SELECT s1, SUM(s2) FROM test WHERE s1 = 'b' GROUP BY s1;
 s1 | sum
----+------
 b  | 6000
(1 row)

-- Drop.
gaussdb=# DROP TABLE test, tb1, tb2;
```

1.13.19.3.4 Pagination Queries

The syntax is as follows:

```
SELECT query_list FROM table_name [ LIMIT { [offset,] count | ALL } ]
SELECT query_list FROM table_name [ FETCH { FIRST | NEXT } [ count ] { ROW | ROWS } ONLY [OFFSET offset_count]]
```

- **offset**: indicates the number of rows to skip before starting to return rows.
- **count**: indicates the number of data records to be retrieved after the skipped rows.
- **ALL**: indicates that all data after the skipped rows is retrieved.
- **FIRST**: queries data from the beginning of the result set. It can be used together with **OFFSET** to start the query from the offset specified by **offset_count**.
- **NEXT**: It can be used together with **OFFSET** to start the query from the offset specified by **offset_count**. If **OFFSET** is not specified, data is queried from the beginning of the result set.
- **offset_count**: offset, indicating that data is queried after the specified offset.

```
-- Create a table and insert 100 data records into the table.
gaussdb=# CREATE TABLE testl(id int PRIMARY KEY, flag varchar(10));
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# INSERT INTO testl (id, flag) VALUES (generate_series(1,100),'flag'||generate_series(1,100));

-- Query the first five data records.
gaussdb=# SELECT * FROM testl ORDER BY 1  LIMIT 5;
 id | flag
----+-------
  1 | flag1
  2 | flag2
  3 | flag3
  4 | flag4
  5 | flag5
(5 rows)

gaussdb=# SELECT * FROM testl ORDER BY 1 FETCH FIRST 5 ROWS ONLY;
 id | flag
----+-------
  1 | flag1
  2 | flag2
  3 | flag3
  4 | flag4
  5 | flag5
(5 rows)

gaussdb=# SELECT * FROM testl ORDER BY 1 FETCH FIRST 2+3 ROWS ONLY;
 id | flag
----+-------
  1 | flag1
  2 | flag2
  3 | flag3
  4 | flag4
  5 | flag5
(5 rows)

gaussdb=# SELECT * FROM testl ORDER BY 1 FETCH FIRST 4.5 ROWS ONLY;
 id | flag
----+-------
  1 | flag1
  2 | flag2
  3 | flag3
  4 | flag4
  5 | flag5
(5 rows)

-- Query four data records after the twentieth record.
gaussdb=# SELECT * FROM testl ORDER BY 1 LIMIT 20,4;
 id |  flag
----+--------
 21 | flag21
 22 | flag22
 23 | flag23
 24 | flag24
(4 rows)

gaussdb=# SELECT * FROM testl ORDER BY 1 FETCH NEXT 4 ROWS ONLY OFFSET 20;
 id |  flag
----+--------
 21 | flag21
 22 | flag22
 23 | flag23
 24 | flag24
(4 rows)

-- Query all data after the ninety-sixth record.
gaussdb=# SELECT * FROM testl ORDER BY 1 LIMIT 96,ALL;
 id  |  flag
-----+---------
  97 | flag97
  98 | flag98
  99 | flag99
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 100 | flag100
(4 rows)

-- Drop the table.
gaussdb=# DROP TABLE testl;
```

1.13.19.3.5 Partition Queries

Queries data in a specified partition.

Example:

```
-- Create a range partitioned table.
gaussdb=# CREATE TABLE test_range1(
    id INT,
    info VARCHAR(20)
) PARTITION BY RANGE (id) (
    PARTITION p1 VALUES LESS THAN (200),
    PARTITION p2 VALUES LESS THAN (400),
    PARTITION p3 VALUES LESS THAN (600),
    PARTITION p4 VALUES LESS THAN (800),
    PARTITION pmax VALUES LESS THAN (MAXVALUE)
);

-- Insert 1000 data records.
gaussdb=# INSERT INTO test_range1 VALUES(GENERATE_SERIES(1,1000),'abcd');

-- Query the number of data records in the p1 partition.
gaussdb=# SELECT COUNT(*) FROM test_range1 PARTITION (p1);
 count
-------
   199
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE test_range1;
```

1.13.19.3.6 Join Queries

A join query may also be referred to as a cross-table query, where multiple tables need to be joined.

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE emp(
    id int,         -- Employee ID.
    name varchar,   -- Employee name.
    deptno int      -- Department ID.
);

gaussdb=# CREATE TABLE dept(
    deptno int,     -- Department ID.
    depname varchar -- Department name.
);

gaussdb=# INSERT INTO emp VALUES (1, 'Joe', 10), (2, 'Scott', 20), (3, 'Ben', 999);   -- The department of Ben has not been confirmed yet. Therefore, the department ID is 999.
gaussdb=# INSERT INTO dept VALUES (10, 'hr'), (20, 'it'), (30, 'sal');                -- There is no employee in the sal department.
```

- INNER JOIN

  ```
  gaussdb=# SELECT t1.id,t1.name,t2.depname FROM emp t1 JOIN dept t2 ON t1.deptno = t2.deptno;
   id | name  | depname
  ----+-------+---------
    1 | Joe   | hr
    2 | Scott | it
  (2 rows)
  ```

- LEFT JOIN

  ```
  gaussdb=# SELECT t1.id,t1.name,t2.depname FROM emp t1 LEFT JOIN dept t2 ON t1.deptno =
  t2.deptno;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 id | name  | depname
----+-------+---------
  1 | Joe   | hr
  2 | Scott | it
  3 | Ben   |
(3 rows)
```

- RIGHT JOIN

```
gaussdb=# SELECT t1.id,t1.name,t2.depname FROM emp t1 RIGHT JOIN dept t2 ON t1.deptno =
t2.deptno;
 id | name  | depname
----+-------+---------
  1 | Joe   | hr
  2 | Scott | it
    |       | sal
(3 rows)
```

- FULL JOIN

```
gaussdb=# SELECT t1.id,t1.name,t2.depname FROM emp t1 FULL JOIN dept t2 ON t1.deptno =
t2.deptno;
 id | name  | depname
----+-------+---------
  1 | Joe   | hr
  2 | Scott | it
    |       | sal
  3 | Ben   |
(4 rows)

-- Drop the table.
gaussdb=# DROP TABLE emp,dept;
```

- Multi-table join

```
gaussdb=# CREATE TABLE t1(a int, b int, c int);
gaussdb=# CREATE TABLE t2(a int, b int, c int);
gaussdb=# CREATE TABLE t3(a int, b int, c int);

gaussdb=# INSERT INTO t1 VALUES(1,generate_series(1,10),generate_series(1,20));
gaussdb=# INSERT INTO t2 VALUES(generate_series(1,5),generate_series(1,10),generate_series(1,20));
gaussdb=# INSERT INTO t3
VALUES(generate_series(6,10),generate_series(1,20),generate_series(1,10));

gaussdb=# SELECT * FROM t1 LEFT JOIN t2 ON t1.a =t2.a INNER JOIN t3 ON t1.b = t3.b AND t2.c =
t3.c;
 a | b | c  | a | b | c | a | b | c
---+---+----+---+---+---+---+---+---
 1 | 1 |  1 | 1 | 1 | 1 | 6 | 1 | 1
 1 | 6 |  6 | 1 | 6 | 6 | 6 | 6 | 6
 1 | 1 | 11 | 1 | 1 | 1 | 6 | 1 | 1
 1 | 6 | 16 | 1 | 6 | 6 | 6 | 6 | 6
(4 rows)

gaussdb=# DROP TABLE t1,t2,t3;
```

1.13.19.3.7 Subqueries

A query can be nested in another query, and its result is used as the data source or condition of the other query. Outer queries are also called parent queries, and inner queries are also called subqueries.

- Subqueries can be classified into single-row subqueries and multi-row subqueries based on the number of records returned.
- Subqueries can be classified into correlated subqueries and non-correlated subqueries based on whether a subquery is executed for multiple times.

Single-Row Subqueries

Single-row subquery operators include >=, >, <=, <, and <>.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a student table and insert data into the table.
gaussdb=# CREATE TABLE student(
    sid VARCHAR(5),      -- Student ID
    grade INT,           -- Grade
    name VARCHAR(20),    -- Name
    height INT           -- Height
);
gaussdb=# INSERT INTO student VALUES ('00001',1,'Scott',135),('00002',1,'Jack',95),('00003',1,'Ben',100);
gaussdb=# INSERT INTO student VALUES ('00004',2,'Henry',115),('00005',2,'Jordan',130),
('00006',2,'Bob',126);
gaussdb=# INSERT INTO student VALUES ('00007',3,'Bella',128),('00008',3,'Alicia',136);
-- Create a teacher table and insert data into the table.
gaussdb=# CREATE TABLE teacher (
    name VARCHAR(20),    -- Teacher name
    grade INT            -- Class
);
-- Insert data.
gaussdb=# INSERT INTO teacher VALUES ('Bill',1),('Sally',2),('Luke',3);

-- Query the students who are taller than Bella.
gaussdb=# SELECT * FROM student
WHERE height > (SELECT height FROM student WHERE name = 'Bella');
  sid  | grade |  name  | height
-------+-------+--------+--------
 00001 |     1 | Scott  |    135
 00005 |     2 | Jordan |    130
 00008 |     3 | Alicia |    136
(3 rows)
```

Multi-Row Subqueries

Multi-row subquery operators are as follows:

- IN: equal to any value in the list.
- ANY: used together with single-row comparison operators to compare with any value returned by the subquery.
- ALL: used together with single-row comparison operators to compare with all values returned by the subquery.
- SOME: another name for ANY and both operators have the same effect.

Example: Query the students of Sally and Luke.

```
gaussdb=# SELECT * FROM student t1 WHERE t1.grade IN (
    SELECT grade FROM teacher WHERE name = 'Sally' OR name = 'Luke'
);
  sid  | grade |  name  | height
-------+-------+--------+--------
 00004 |     2 | Henry  |    115
 00005 |     2 | Jordan |    130
 00006 |     2 | Bob    |    126
 00007 |     3 | Bella  |    128
 00008 |     3 | Alicia |    136
(5 rows)
```

Example: Query the students in grade 2 who are taller than any students in grade 3.

```
gaussdb=# SELECT * FROM student
WHERE grade = 2 AND
      height > ANY (SELECT height FROM student WHERE grade = 3);
  sid  | grade |  name  | height
-------+-------+--------+--------
 00005 |     2 | Jordan |    130
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- The query result is equivalent to that of the following query:
gaussdb=# SELECT * FROM student
WHERE grade = 2 AND
    height > (SELECT MIN(height) FROM student WHERE grade = 3);
```

Example: Query the students in grade 1 who are taller than all students in grade 2.

```
gaussdb=# SELECT * FROM student
WHERE grade = 1 AND
    height > ALL (SELECT height FROM student WHERE grade = 2);
 sid  | grade | name  | height
-------+-------+-------+--------
 00001 |     1 | Scott |    135
(1 row)

-- The query result is equivalent to that of the following query:
gaussdb=# SELECT * FROM student
WHERE grade = 1 AND
    height > (SELECT MAX(height) FROM student WHERE grade = 2);
```

Correlated Subqueries

Characteristics: Subqueries cannot run independently and are related to parent queries. Execute parent queries and then subqueries. Each time a parent query is executed, its subquery is recalculated.

Example: Query the students whose height is greater than the average height of the students in the class.

```
gaussdb=# SELECT * FROM student out
WHERE height > (SELECT AVG(height) FROM student
              WHERE grade = out.grade);
 sid  | grade |  name  | height
-------+-------+--------+--------
 00001 |     1 | Scott  |    135
 00005 |     2 | Jordan |    130
 00006 |     2 | Bob    |    126
 00008 |     3 | Alicia |    136
(4 rows)
```

Non-Correlated Subqueries

Characteristics: A subquery queries the value and then returns the result to the outer layer for query.

Example: Query the students whose height is greater than the average height of the students in the class.

```
gaussdb=# SELECT t1.*
FROM student t1,
    (SELECT grade, AVG(height) avg_hei FROM student GROUP BY grade) t2
WHERE t1.grade = t2.grade AND
    t1.height > t2.avg_hei;
 sid  | grade |  name  | height
-------+-------+--------+--------
 00001 |     1 | Scott  |    135
 00005 |     2 | Jordan |    130
 00006 |     2 | Bob    |    126
 00008 |     3 | Alicia |    136
(4 rows)

-- Drop the table.
gaussdb=# DROP TABLE student;
gaussdb=# DROP TABLE teacher;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.19.3.8 Hierarchical Queries

Syntax

```
SELECT { { column | LEVEL | CONNECT_BY_ISLEAF | CONNECT_BY_ISCYCLE | SYS_CONNECT_BY_PATH(col,
separator) | CONNECT_BY_ROOT(col) } [, ...] }
 FROM table
START WITH condition
CONNECT BY [NOCYCLE] [PRIOR column1 = column2|column1 = prior column2]
[ORDER SIBLINGS BY expression] [ORDER BY expression]
```

Parameters

- **LEVEL**: pseudocolumn that returns the query level.
- **CONNECT_BY_ISLEAF**: pseudocolumn that returns whether the current node is a leaf node.
- **CONNECT_BY_ISCYCLE**: pseudocolumn that returns whether a row in the current node causes a cycle, which means it has a child which is also its parent.
- **NOCYCLE**: specifies whether to stop recursion when a cycle is detected. If it is not specified, an error is reported when a cycle is detected.
- **SYS_CONNECT_BY_PATH(col, separator)**: function used for hierarchical queries. It returns the connection path from the root node to the current row. The **col** parameter indicates the name of the column displayed in the path, and the **separator** parameter indicates the connector.
- **CONNECT_BY_ROOT(col)**: function used for hierarchical queries. It displays the top node above the current node. The **col** parameter indicates the name of the output column.
- **START WITH**: specifies the hierarchy, that is, the root row of the query.
- **CONNECT BY**: specifies the relationship between parent rows and child rows.
- **PRIOR**: specifies the parent level.
- **ORDER SIBLINGS BY**: specifies the sequence in the same level.

Examples

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE area (id INT,name VARCHAR(25),parent_id INT);
gaussdb=# INSERT INTO area VALUES (1,'China',NULL);
gaussdb=# INSERT INTO area VALUES (2,'Beijing',1);
gaussdb=# INSERT INTO area VALUES(3,'Chaoyang',2);
gaussdb=# INSERT INTO area VALUES(4,'Shaanxi',1);
gaussdb=# INSERT INTO area VALUES(5,'Xi'an',4);
gaussdb=# INSERT INTO area VALUES(6,'Yanta',5);
gaussdb=# INSERT INTO area VALUES(7,'Weiyang',5);

-- Query the hierarchy.
gaussdb=# SELECT level,
    name,
    connect_by_isleaf,
    connect_by_iscycle,
    connect_by_root(name),
    sys_connect_by_path(name, '->')
FROM area START WITH (id = 1) CONNECT BY PRIOR id = parent_id;
 level |  name  | connect_by_isleaf | connect_by_iscycle | connect_by_root |      sys_connect_by_path
-------+--------+-------------------+--------------------+-----------------+--------------------------------
     1 | China  |                 0 |                  0 | China           | -> China
     2 | Shaanxi |                0 |                  0 | China           | -> China -> Shaanxi
     3 | Xi'an  |                 0 |                  0 | China           | -> China -> Shaanxi -> Xi'an
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    4 | Weiyang |              1 |              0 | China          | -> China -> Shaanxi -> Xi'an -> Weiyang
    4 | Yanta   |              1 |              0 | China          | -> China -> Shaanxi -> Xi'an -> Yanta
    2 | Beijing |              0 |              0 | China          | -> China -> Beijing
    3 | Chaoyang |             1 |              0 | China          | -> China -> Beijing -> Chaoyang
(7 rows)

-- Use WHERE to filter a node (only Beijing).
gaussdb=# SELECT level, name FROM area
   WHERE id <> 2
   START WITH (id = 1)
   CONNECT BY PRIOR id = parent_id;
 level |  name
-------+--------
     1 | China
     2 | Shaanxi
     3 | Xi'an
     4 | Weiyang
     4 | Yanta
     3 | Chaoyang
(6 rows)

-- Cycle records are found during the hierarchical query.
gaussdb=# INSERT INTO area VALUES(8,'Jiangsu',10);
gaussdb=# INSERT INTO area VALUES(9,'Nanjing',8);
gaussdb=# INSERT INTO area VALUES(10,'Yuhuatai',9);
gaussdb=# SELECT level,
   name,
   connect_by_isleaf,
   connect_by_iscycle,
   connect_by_root(name),
   sys_connect_by_path(name, '->')
FROM area START WITH (id = 8) CONNECT BY NOCYCLE PRIOR id = parent_id;
 level |   name   | connect_by_isleaf | connect_by_iscycle | connect_by_root |    sys_connect_by_path
-------+----------+-------------------+--------------------+-----------------+---------------------------
     1 | Jiangsu  |                 0 |                  0 | Jiangsu         | -> Jiangsu
     2 | Nanjing  |                 0 |                  0 | Jiangsu         | -> Jiangsu -> Nanjing
     3 | Yuhuatai |                 1 |                  1 | Jiangsu         | -> Jiangsu -> Nanjing -> Yuhuatai
(3 rows)

gaussdb=# DELETE FROM area WHERE id >= 8;

-- Add a condition to the CONNECT BY PRIOR clause to filter a branch (Beijing and its sub-regions).
gaussdb=# SELECT level, name FROM area
   START WITH (id = 1)
   CONNECT BY PRIOR id = parent_id AND id <> 2;
 level |  name
-------+--------
     1 | China
     2 | Shaanxi
     3 | Xi'an
     4 | Weiyang
     4 | Yanta
(5 rows)

Add the PRIOR modifier to the id column in the SELECT query to display the parent row tuple.
gaussdb=# SELECT level, PRIOR id as pid, parent_id, name FROM area
   START WITH (id = 1)
   CONNECT BY PRIOR id = parent_id;
 level | pid | parent_id |  name
-------+-----+-----------+--------
     1 |     |           | China
     2 |   1 |         1 | Shaanxi
     3 |   4 |         4 | Xi'an
     4 |   5 |         5 | Weiyang District
     4 |   5 |         5 | Yanta District
     2 |   1 |         1 | Beijing
     3 |   2 |         2 | Chaoyang District
(7 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop the table.
gaussdb=# DROP TABLE area;
```

1.13.19.3.9 Compound Queries

A query that contains a compound operator is a compound query. All compound queries have the same priority. The number of columns and expressions in each query result in a set operation must be the same, and the types must be compatible.

Common set operations include:

- UNION: a union set of the two query result sets where duplicates are removed.
- UNION ALL: a union set of the two queries where the results of the two queries are simply combined.
- INTERSECT: intersection of two queries (only duplicated data is displayed).
- MINUS: difference of two query result sets. Only the data that exists in the first result set but does not exist in the second result set is displayed, and the data is sorted based on the results in the first column.

Examples

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE test1(c11 INT, c12 VARCHAR);
gaussdb=# INSERT INTO test1 VALUES (1,'a'),(2,'b'),(4,'d');

gaussdb=# CREATE TABLE test2(c21 INT, c22 VARCHAR);
gaussdb=# INSERT INTO test2 VALUES (1,'a'),(3,'c');
```

- UNION

```
gaussdb=# SELECT * FROM test1 UNION SELECT * FROM test2;
 c11 | c12
-----+-----
   1 | a
   4 | d
   2 | b
   3 | c
(4 rows)
```

- UNION ALL

```
gaussdb=# SELECT * FROM test1 UNION ALL SELECT * FROM test2;
 c11 | c12
-----+-----
   1 | a
   2 | b
   4 | d
   1 | a
   3 | c
(5 rows)
```

- INTERSECT

```
gaussdb=# SELECT * FROM test1 INTERSECT SELECT * FROM test2;
 c11 | c12
-----+-----
   1 | a
(1 row)
```

- MINUS

```
gaussdb=# SELECT * FROM test1 MINUS SELECT * FROM test2;
 c11 | c12
-----+-----
   4 | d
   2 | b
(2 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop the table.
gaussdb=# DROP TABLE test1,test2;
```

1.13.19.3.10 Rows to Columns and Columns to Rows

- Rows to columns

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE test_p2(id INT, class VARCHAR(20), score INT);
gaussdb=# INSERT INTO test_p2 VALUES (1,'math',64), (1,'english',78);
gaussdb=# INSERT INTO test_p2 VALUES (2,'math',98), (2,'english',82);
gaussdb=# INSERT INTO test_p2 VALUES (3,'math',68), (3,'english',59);
gaussdb=# SELECT * FROM test_p2;
 id |  class  | score
----+---------+-------
  3 | math    |    68
  3 | english |    59
  1 | math    |    64
  1 | english |    78
  2 | math    |    98
  2 | english |    82
(6 rows)

-- Convert rows to columns.
gaussdb=# SELECT * FROM test_p2 PIVOT(MAX(score) FOR class IN('math','english'));
 id | 'math' | 'english'
----+--------+-----------
  3 |     68 |        59
  1 |     64 |        78
  2 |     98 |        82
(3 rows)

-- Drop the table.
gaussdb=# DROP TABLE test_p2;
```

- Columns to rows

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE test_p1(id INT, math INT, english INT);
gaussdb=# INSERT INTO test_p1 VALUES (1,84,78), (2,98,82), (3,68,59);
gaussdb=# SELECT * FROM test_p1;
 id | math | english
----+------+---------
  3 |   68 |      59
  1 |   84 |      78
  2 |   98 |      82
(3 rows)

-- Convert columns to rows.
gaussdb=# SELECT * FROM test_p1 UNPIVOT(score FOR class IN(math, english));
 id |  class  | score
----+---------+-------
  3 | MATH    |    68
  3 | ENGLISH |    59
  1 | MATH    |    84
  1 | ENGLISH |    78
  2 | MATH    |    98
  2 | ENGLISH |    82
(6 rows)

-- Drop the table.
gaussdb=# DROP TABLE test_p1;
```

1.13.19.3.11 Window Function Queries

The window function performs a calculation on the related row set in the data set and returns a result set. Compared with the calculation performed by the aggregate function, the window function does not aggregate multiple rows into a single output row.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE fruit_sale (
  "statistical_date" date,
  "product" varchar(255),
  "year" varchar(5),
  "sales_quantity" numeric(8),
  "amount" numeric(8)
);

gaussdb=# INSERT INTO fruit_sale VALUES ('2024-01-01', 'Watermelon', '2024', 1721, 253541);
gaussdb=# INSERT INTO fruit_sale VALUES ('2024-01-01', 'Apple', '2024', 5559, 269419);
gaussdb=# INSERT INTO fruit_sale VALUES ('2024-02-01', 'Watermelon', '2024', 4711, 129644);

-- Collect statistics on the sales volume of all fruits.
gaussdb=# SELECT *,SUM(sales_quantity) OVER (PARTITION by null) total_qty FROM fruit_sale;
 statistical_date | product | year | sales_quantity | amount | total_qty
------------------+---------+------+----------------+--------+-----------
 2024-01-01       | Watermelon    | 2024 |          1721 | 253541 |     11991
 2024-01-01       | Apple    | 2024 |          5559 | 269419 |     11991
 2024-02-01       | Watermelon    | 2024 |          4711 | 129644 |     11991
(3 rows)

-- Drop the table.
gaussdb=# DROP TABLE fruit_sale;
```

1.13.19.3.12 Simplified Queries

Simplified queries can be performed on tables without the SELECT keyword.

Example:

```
-- Create a table.
gaussdb=# CREATE TABLE t1(c1 int, c2 int, c3 int);

-- Insert data.
gaussdb=# INSERT INTO t1 VALUES (1,2,3);
 c1 | c2 | c3
----+----+----
  1 |  2 |  3
(1 row)

-- Query data in the table.
gaussdb=# TABLE t1;
 c1 | c2 | c3
----+----+----
  1 |  2 |  3
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE t1;
```

1.13.19.3.13 Queries Using Hints

Hints can be used to modify the SQL execution plan.

Example:

```
-- Create a table.
gaussdb=# CREATE TABLE t1(c1 int, c2 int, c3 int);
gaussdb=# CREATE TABLE t2(c1 int, c2 int, c3 int);
gaussdb=# CREATE INDEX it1 ON t1(c1,c2);

-- Specify the scanning mode using a hint.
gaussdb=# EXPLAIN SELECT /*+ indexscan(t1 it1)*/* from t1,t2 WHERE t1.c1=t2.c2;
 id |                operation                 | E-rows | E-width | E-costs
----+------------------------------------------+--------+---------+---------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 1 | ->  Streaming (type: GATHER)               |     30 |      24 | 61.51
 2 |    ->  Hash Join (3,4)                     |     30 |      24 | 60.26
 3 |       ->  Index Scan using it1 on t1       |     30 |      12 | 44.46
 4 |       ->  Hash                             |     29 |      12 | 15.49
 5 |         ->  Streaming(type: REDISTRIBUTE) |     30 |      12 | 15.49
 6 |           ->  Seq Scan on t2               |     30 |      12 | 14.14
(6 rows)

 Predicate Information (identified by plan id)
-----------------------------------------------
   2 --Hash Join (3,4)
         Hash Cond: (t1.c1 = t2.c2)
(2 rows)

-- Specify the number of rows using a hint.
gaussdb=# EXPLAIN SELECT /*+ rows(t1 t2 #5)*/* FROM t1,t2;
 id |             operation              | E-rows | E-width | E-costs
----+------------------------------------+--------+---------+---------
 1 | ->  Streaming (type: GATHER)        |      5 |      24 | 37.60
 2 |    ->  Nested Loop (3,5)            |      5 |      24 | 37.35
 3 |       ->  Streaming(type: BROADCAST) |     90 |      12 | 17.93
 4 |         ->  Seq Scan on t1          |     30 |      12 | 14.14
 5 |       ->  Materialize               |     30 |      12 | 14.21
 6 |         ->  Seq Scan on t2          |     30 |      12 | 14.14
(6 rows)

-- Drop the table.
gaussdb=# DROP TABLE t1;
gaussdb=# DROP TABLE t2;
```

1.13.19.4 SELECT INTO

Description

SELECT INTO creates a table based on the query result and inserts data retrieved by the query into the new table. Different from common SELECT, SELECT INTO does not return data to the client. The columns of the new table have the same names and data types as the output columns of SELECT.

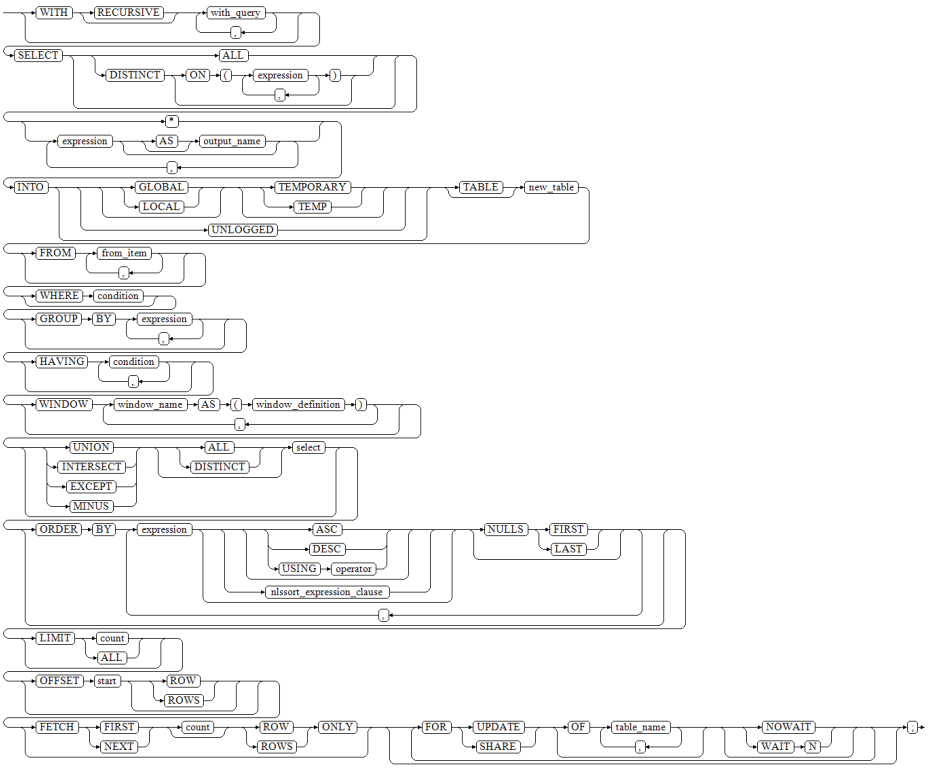
Precautions

CREATE TABLE AS provides functions similar to SELECT INTO in functions and provides a superset of functions provided by SELECT INTO. You are advised to use CREATE TABLE AS because SELECT INTO cannot be used in a stored procedure.

Syntax

```
[ WITH [ RECURSIVE ] with_query [, ...] ]
SELECT [ ALL | DISTINCT [ ON ( expression [, ...] ) ] ]
   { * | {expression [ [ AS ] output_name ]} [, ...] }
   INTO [ [ GLOBAL | LOCAL ] [ TEMPORARY | TEMP ] | UNLOGGED ] [ TABLE ] new_table
   [ FROM from_item [, ...] ]
   [ WHERE condition ]
   [ GROUP BY expression [, ...] ]
   [ HAVING condition [, ...] ]
   [ WINDOW {window_name AS ( window_definition )} [, ...] ]
   [ { UNION | INTERSECT | EXCEPT | MINUS } [ ALL | DISTINCT ] select ]
   [ ORDER BY {expression [ [ ASC | DESC | USING operator ] | nlssort_expression_clause ] [ NULLS { FIRST |
LAST } ]} [, ...] ]
   [ LIMIT { count | ALL } ]
   [ OFFSET start [ ROW | ROWS ] ]
   [ FETCH { FIRST | NEXT } [ count ] { ROW | ROWS } ONLY ]
   [ {FOR { UPDATE | SHARE } [ OF table_name [, ...] ] [ NOWAIT | WAIT N]} [...] ];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- **new_table**

  Specifies the name of the new table.

- **UNLOGGED**

  Specifies that the table is created as an unlogged table. Data written to unlogged tables is not written to the WALs, which makes them considerably faster than ordinary tables. However, in the event of a conflict, OS restart, database restart, primary/standby switchover, power supply interruption, or unclean shutdown, unlogged tables will be cleared, posing a risk of data loss. The contents of an unlogged table are also not replicated to standby servers. The indexes created in unlogged tables are not automatically recorded.

  - Usage scenario: Unlogged tables do not ensure data security. Users can back up data before using unlogged tables; for example, users should back up the data before a system upgrade.
  - Troubleshooting: If data is missing in the indexes of unlogged tables due to some unexpected operations such as an unclean shutdown, users should rebuild the indexes with errors.

NOTE

For details about other parameters of SELECT INTO, see **Parameters**.

Examples

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE tbl_person (
  id     integer,
  name   varchar(20),
  gender varchar(5) CHECK(gender = 'Male' or gender = 'Female')
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
);
gaussdb=# INSERT INTO tbl_person VALUES (1, 'Bob', 'Male'),(2, 'Anne', 'Female'),(3, 'Jack', 'Male'),(4,
'Danny', 'Male'),(5, 'Alice', 'Female'),(6, 'Susan', 'Female');

-- Add information about all persons whose gender is male in the person table to the new table.
gaussdb=# SELECT * INTO tbl_man FROM tbl_person WHERE gender =' Male'

-- Query data in the tbl_man table.
gaussdb=# SELECT * FROM tbl_man;
 id | name  | gender
----+-------+-----
  1 | Bob   | Male
  3 | Jack  | Male
  4 | Danny | Male
(3 rows)

-- Drop the table.
gaussdb=# DROP TABLE tbl_person, tbl_man;
```

Helpful Links

SELECT

1.13.19.5 SET

Description

Modifies GUC parameters.

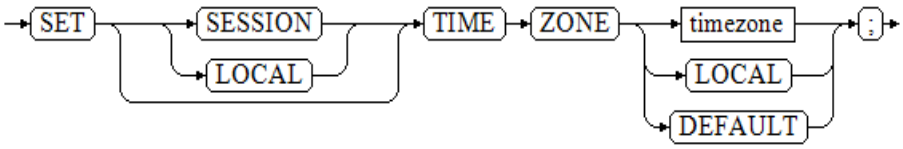
Precautions

Most GUC parameters can be modified at runtime by using SET, but some parameters cannot be modified after a service or session starts.

Syntax

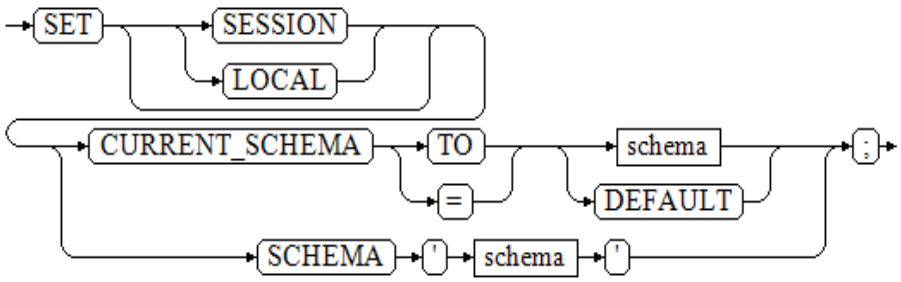
- Set the time zone.

```
SET [ SESSION | LOCAL ] TIME ZONE { timezone | LOCAL | DEFAULT };
```



- Set the schema.

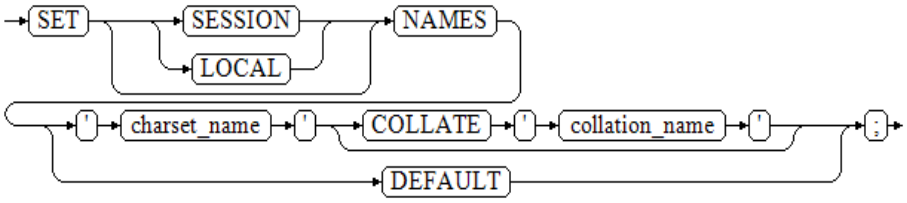
```
SET [ SESSION | LOCAL ]
    {CURRENT_SCHEMA { TO | = } { schema | DEFAULT }
    | SCHEMA 'schema'};
```



- Set the client character set.

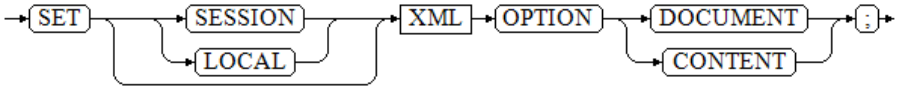
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
SET [ SESSION | LOCAL ] NAMES {'charset_name' [COLLATE 'collation_name'] | DEFAULT};
```



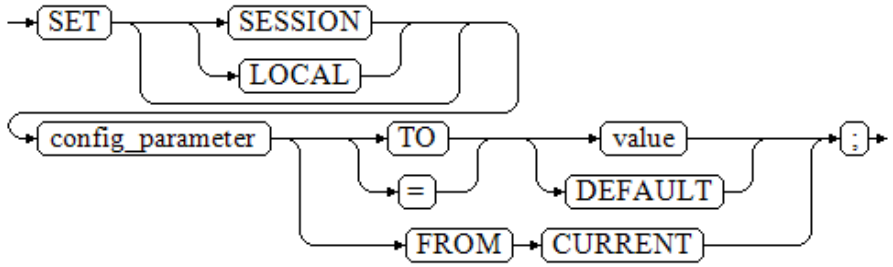
- Set XML parsing mode.

```
SET [ SESSION | LOCAL ] XML OPTION { DOCUMENT | CONTENT };
```



- Set other GUC parameters.

```
SET [ SESSION | LOCAL ]
   {config_parameter { { TO | = } { value | DEFAULT }
                     | FROM CURRENT }};
```



Parameters

- **SESSION**

  Specifies that the specified parameters take effect only for the current session. If neither **SESSION** nor **LOCAL** appears, **SESSION** is the default value.

  If this command is executed in a transaction, the effect of the command disappears after the transaction is rolled back. Once the surrounding transaction is committed, the effect will persist until the end of the session, unless the parameters are reset by another SET statement.

- **LOCAL**

  Specifies that the specified parameters take effect only for the current transaction. After COMMIT or ROLLBACK, session-level settings will take effect again.

  The effect of this command lasts only until the end of the current transaction, whether the transaction is committed or not. A special case is where SET is followed by SET LOCAL within a single transaction. In this case, SET LOCAL will take effect until the end of the transaction, but after the transaction is committed, SET will take effect.

- **TIME ZONE timezone**

  Specifies the local time zone for the current session.

  Value range: valid local time zones. The corresponding GUC parameter is **TimeZone**. The default value is **PRC**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> NOTE
>
> SET TIME ZONE cannot write a time zone in hours and minutes, such as "-12:30", but can write a time zone in English or decimal, for example, "EST", "America/New_York", and "-12.5".

- **CURRENT_SCHEMA schema**

  Specifies the current schema.

  Value range: existing schema names.

- **SCHEMA schema**

  Same as **CURRENT_SCHEMA**. The schema here is a string.

  Example: **set schema 'public';**

- **NAMES {'charset_name' [COLLATE 'collation_name'] | DEFAULT};**
  - The COLLATE clause can be specified when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** set to **'5.7'**, and **b_format_dev_version** set to **'s2'**.

    Sets the client character encoding, character set of a constant string, collation, and character set of the returned result.

    It is equivalent to:

    ```
    SET client_encoding = charset_name;
    SET character_set_connection = charset_name;
    SET collation_connection = collation_name;
    SET character_set_results = charset_name;
    ```

    Value range: character sets and collations supported in MYSQL-compatible mode. Currently, **charset_name** must be one of the character sets specified for the database.
  - The COLLATE clause cannot be specified in other scenarios.

    Specifies the client character encoding.

    It is equivalent to:

    ```
    SET client_encoding TO charset_name;
    ```

    Value range: a valid character encoding name. The default value is **UTF8**.

- **XML OPTION option**

  Specifies the XML parsing mode.

  Value range: **CONTENT** (default) and **DOCUMENT**

- **config_parameter**

  Specifies the name of the configurable GUC parameter. You can use SHOW ALL to view the available GUC parameters. For details about the setting rules, see **Setting Parameters**.

  > NOTE
  >
  > Some parameters viewed by SHOW ALL cannot be set by using SET. For example, **max_datanodes**.

- **value**

  Specifies the new value of **config_parameter**. You can specify the parameter as a string constant, identifier, number, or comma-separated list. When **DEFAULT** is specified, the parameter is set to its default value.

- **FROM CURRENT**

  Uses the value of **config_parameter** of the current session.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

```
-- Set the time zone.
gaussdb=# SET TIME ZONE 'PST8PDT';

-- Set the schema search path.
gaussdb=# SET search_path TO tpcds, public;

-- Set client-side encoding.
gaussdb=# SET client_encoding TO utf8;

-- Set XML parsing mode.
gaussdb=# SET XML OPTION DOCUMENT;

-- Set the date style to the traditional POSTGRES style (date placed before month).
gaussdb=# SET datestyle TO postgres;
```

Helpful Links

RESET and **SHOW**

1.13.19.6 SET CONSTRAINTS

Description

SET CONSTRAINTS sets the behavior of constraint checking within the current transaction.

IMMEDIATE constraints are checked at the end of each statement. DEFERRED constraints are not checked until transaction commit. Each constraint has its own mode.

Upon creation, a constraint is given one of three characteristics **DEFERRABLE INITIALLY DEFERRED**, **DEFERRABLE INITIALLY IMMEDIATE**, or **NOT DEFERRABLE**. The third class is always **IMMEDIATE** and is not affected by the **SET CONSTRAINTS** statement. The first two classes start every transaction in specified modes, but its behaviors can be changed within a transaction by SET CONSTRAINTS.

SET CONSTRAINTS with a list of constraint names changes the mode of just those constraints (which must all be deferrable). If multiple constraints match a name, the name is affected by all of these constraints. **SET CONSTRAINTS ALL** changes the modes of all deferrable constraints.

When **SET CONSTRAINTS** changes the mode of a constraint from **DEFERRED** to **IMMEDIATE**, any data that is modified at the end of the transaction is checked during the execution of the **SET CONSTRAINTS** statement. If any such constraint is violated, the **SET CONSTRAINTS** fails (and does not change the constraint mode). Therefore, **SET CONSTRAINTS** can be used to force checking of constraints to occur at a specific point in a transaction. Check and unique constraints are always checked immediately when a row is inserted or modified.

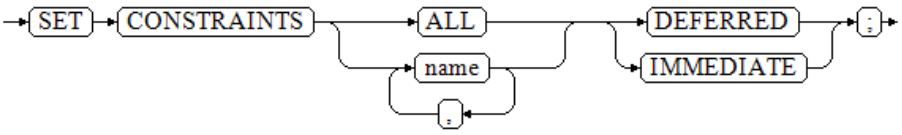
Precautions

SET CONSTRAINTS sets the behavior of constraint checking only within the current transaction. Therefore, if you execute this statement outside of a transaction block (START TRANSACTION/COMMIT pair), it will not appear to have any effect.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax

```
SET CONSTRAINTS { ALL | { name  } [, ...] } { DEFERRED | IMMEDIATE };
```



Parameters

- **name**

  Specifies the constraint name.

  Value range: an existing table name, which can be found in the system catalog pg_constraint.

- **ALL**

  Specifies all constraints.

- **DEFERRED**

  Specifies that constraints are not checked until transaction commit.

- **IMMEDIATE**

  Specifies that constraints are checked at the end of each statement.

Examples

```
-- Set that constraints are checked when a transaction is committed.
gaussdb=# SET CONSTRAINTS ALL DEFERRED;

-- Set that constraints are checked at the end of each statement.
gaussdb=# SET CONSTRAINTS ALL IMMEDIATE;
```

1.13.19.7 SET ROLE

Description

Sets the current user identifier for the current session.

Precautions

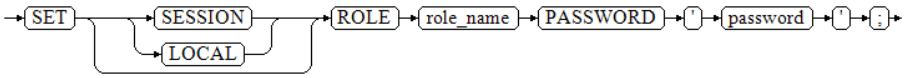
- Users of the current session must be members of specified **rolename**, but the system administrator can choose any roles when separation of duties is disabled.
- This statement may add permissions to or restrict permissions of a user. If the role of a session user has the INHERITS attribute, then the user is automatically granted all permissions of the roles that SET ROLE enables the user to be. In this case, SET ROLE deletes all permissions directly granted to the session user, as well as the permissions of the role it belongs to, that is, the user is left with the permissions of only the specified role. On the other hand, if the role of a session user has the NOINHERITS attribute, SET ROLE deletes permissions directly granted to the session user and obtains permissions of the specified role.
- If a user is configured to use LDAP for login authentication, the user identifier cannot be set to this user by performing this operation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax

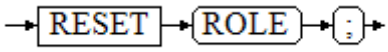
- Set the current user identifier of the current session.

```
SET [ SESSION | LOCAL ] ROLE role_name PASSWORD 'password';
```



- Reset the current user identifier to that of the current session.

```
RESET ROLE;
```



Parameters

- **SESSION**

  Specifies that the command takes effect only for the current session. This parameter is used by default.

- **LOCAL**

  Specifies that the command takes effect only for the current transaction.

- **role_name**

  Indicates the role name.

  Value range: a string. It must be an existing username in the database.

- **password**

  Specifies the password for the role. It must comply with the password convention.

  NOTE

  - The restrictions on using a ciphertext password are as follows:
    - An administrator cannot use a ciphertext password to switch to another administrator but to a user with lower permissions.
  - Ciphertext passwords are usually used in gs_dump and gs_dumpall export scenarios. In other scenarios, you are advised not to use ciphertext passwords directly.

- **RESET ROLE**

  Resets the current user identifier.

Examples

```
-- Query the current session user and the current user.
gaussdb=# SELECT SESSION_USER, CURRENT_USER;
 session_user | current_user
--------------+--------------
 omm          | omm
(1 row)

-- Create a role named paul.
gaussdb=# CREATE ROLE paul PASSWORD '********';

-- Set the current user to paul.
gaussdb=# SET ROLE paul PASSWORD '********';

-- View the current session user and the current user.
gaussdb=> SELECT SESSION_USER, CURRENT_USER;
 session_user | current_user
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------------+--------------
 omm          | paul
(1 row)

-- Reset the current user.
gaussdb=> RESET ROLE;

-- Drop the user.
gaussdb=# DROP USER paul;
```

Helpful Links

SET SESSION AUTHORIZATION

1.13.19.8 SET SESSION AUTHORIZATION

Description

Sets the session user identifier and the current user identifier of the current session to the specified user.

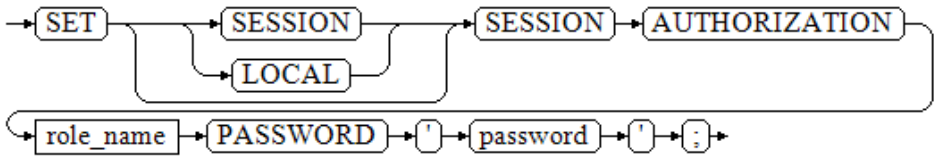
Precautions

- The session identifier can be changed only when the initial session user has the SYSADMIN permission. Otherwise, the system executes the command only when an authenticated username is specified.
- If a user is configured to use LDAP for login authentication, the user identifier cannot be set to this user by performing this operation.

Syntax

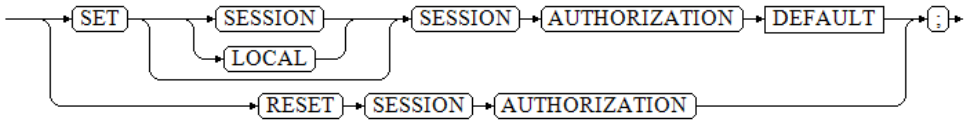
- Set the session user identifier and current user identifier for the current session.

```
SET [ SESSION | LOCAL ] SESSION AUTHORIZATION role_name PASSWORD 'password';
```



- Reset the session and current user identifiers to the originally authenticated username.

```
{SET [ SESSION | LOCAL ] SESSION AUTHORIZATION DEFAULT
   | RESET SESSION AUTHORIZATION};
```



Parameters

- **SESSION**

  Specifies that the command takes effect only for the current session.
- **LOCAL**

  Specifies that the command takes effect only for the current transaction.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **role_name**

  Specifies the username.

  Value range: a string. It must be an existing username in the database.

- **password**

  Specifies the password for the role. The password must comply with the password convention.

  NOTE

  The restrictions on using a ciphertext password are as follows:

  - An administrator cannot use a ciphertext password to switch to another administrator but to a user with lower permissions.

  Ciphertext passwords are usually used in gs_dump and gs_dumpall export scenarios. In other scenarios, you are advised not to use ciphertext passwords directly.

- **DEFAULT**

  Resets the identifiers of the session and current users to the initially authenticated usernames.

Examples

```
-- Query the session user and the current user.
gaussdb=# SELECT SESSION_USER, CURRENT_USER;
 session_user | current_user
--------------+--------------
 omm          | omm
(1 row)

-- Create a role named paul.
gaussdb=# CREATE ROLE paul PASSWORD '********';

-- Set the current user and session user to paul, and query the session user and current user.
gaussdb=# SET SESSION AUTHORIZATION paul PASSWORD '********';
gaussdb=> SELECT SESSION_USER, CURRENT_USER;
 session_user | current_user
--------------+--------------
 paul         | paul
(1 row)

-- Reset the session user and the current user.
gaussdb=> RESET SESSION AUTHORIZATION;
gaussdb=# SELECT SESSION_USER, CURRENT_USER;
 session_user | current_user
--------------+--------------
 omm          | omm
(1 row)

-- Drop the user.
gaussdb=# DROP USER paul;
```

Helpful Links

SET ROLE

1.13.19.9 SET TRANSACTION

Description

Sets the transaction characteristics. Transaction characteristics include transaction isolation level and transaction access mode (read/write or read-only). You can set

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

the current transaction characteristics using LOCAL or the default transaction characteristics of a session using SESSION.

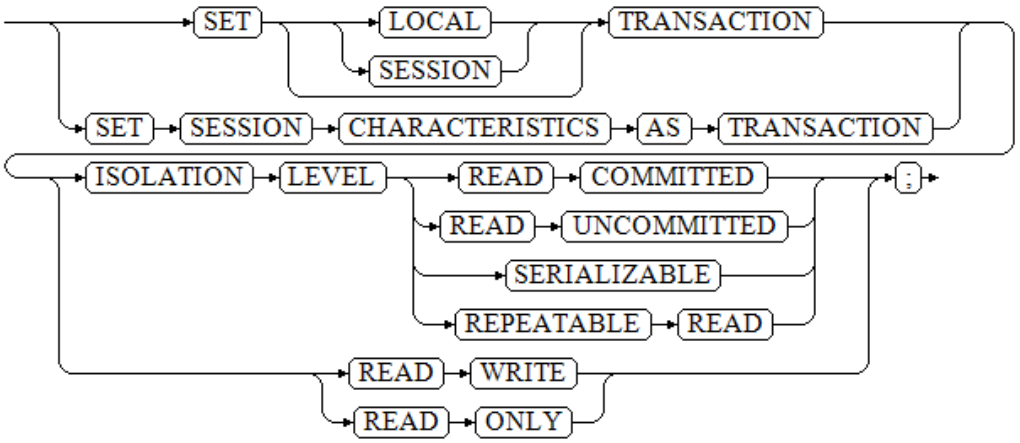
Precautions

The current transaction characteristics must be set in a transaction, that is, **START TRANSACTION** or **BEGIN** must be executed before **SET TRANSACTION** is executed. Otherwise, the setting does not take effect.

Syntax

Set the transaction isolation level and access mode.

```
{ SET [ LOCAL | SESSION ] TRANSACTION | SET SESSION CHARACTERISTICS AS TRANSACTION }
  { ISOLATION LEVEL { READ COMMITTED | READ UNCOMMITTED | SERIALIZABLE | REPEATABLE READ }
  | { READ WRITE | READ ONLY }
  };
```



Parameters

- **LOCAL**

  Specifies that the specified statement takes effect only for the current transaction.

- **SESSION**

  Specifies that the command takes effect only for the current session.

- **SESSION CHARACTERISTICS**

  Specifies that the command takes effect only for the current session.

- **ISOLATION LEVEL**

  Specifies the transaction isolation level. It determines what data a transaction can read when other concurrent transactions are in progress.

  NOTE

  - The isolation level cannot be changed after data is modified using INSERT, DELETE, UPDATE, FETCH, or COPY in the current transaction.

  Value range:

  – **READ COMMITTED**: Only committed data can be read; uncommitted data remains invisible. This is the default.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - **READ UNCOMMITTED**: This isolation level behaves identically to **READ COMMITTED**.
  - **REPEATABLE READ**: The transaction can only read data that was committed before the transaction began. It cannot read uncommitted data or changes committed by other concurrent transactions during its execution.
  - **SERIALIZABLE**: Currently, this isolation level is not supported in GaussDB. It is equivalent to **REPEATABLE READ**.
- **READ WRITE | READ ONLY**

  Specifies the transaction access mode (**READ WRITE** or **READ ONLY**).

Examples

```
-- Start a transaction.
gaussdb=# START TRANSACTION;

-- Set the isolation level of the transaction to READ COMMITTED and the access mode to READ ONLY.
gaussdb=# SET LOCAL TRANSACTION ISOLATION LEVEL READ COMMITTED READ ONLY;

-- Query the current transaction isolation level.
gaussdb=# SHOW transaction_isolation;
 transaction_isolation
-----------------------
 read committed
(1 row)

-- Commit the transaction.

gaussdb=# COMMIT;
```

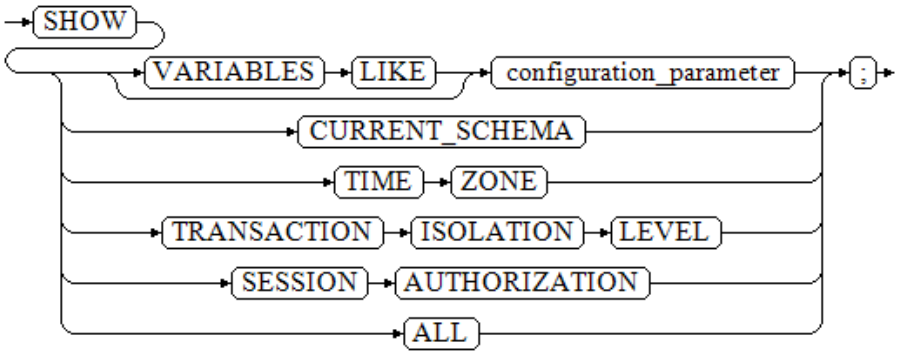
1.13.19.10 SHOW

Description

SHOW shows the current value of a GUC parameter.

Syntax

```
SHOW
 {
   [VARIABLES LIKE] configuration_parameter |
   CURRENT_SCHEMA |
   TIME ZONE |
   TRANSACTION ISOLATION LEVEL |
   SESSION AUTHORIZATION |
   ALL
 };
```



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameters

- **configuration_parameter**

  Specifies the name of the GUC parameter.

  Value range: GUC parameters. You can view the GUC parameters by using the SHOW ALL statement.

  NOTE

  Some parameters viewed by running **SHOW ALL** cannot be set by running **SET**, for example, **max_datanodes**.

- **CURRENT_SCHEMA**

  Specifies the current schema.

- **TIME ZONE**

  Specifies the time zone.

- **TRANSACTION ISOLATION LEVEL**

  Specifies the transaction isolation level.

- **SESSION AUTHORIZATION**

  Specifies the user identifier of the current session.

- **ALL**

  Specifies all GUC parameters.

Examples

```
-- Show the value of timezone.
gaussdb=# SHOW timezone;
TimeZone
----------
 PRC
(1 row)

-- Show the current schema.
gaussdb=# SHOW CURRENT_SCHEMA;
 current_schema
----------------
 "$user",public
(1 row)

-- Show the isolation level of a transaction.
gaussdb=# SHOW TRANSACTION ISOLATION LEVEL;
 transaction_isolation
-----------------------
 read committed
(1 row)

-- Show the user identifier of the current session.
gaussdb=# SHOW SESSION AUTHORIZATION;
 session_authorization
-----------------------
 omm1
(1 row)

-- Show all parameters.
gaussdb=# SHOW ALL;

-- Show all parameters whose names contain var.
gaussdb=# SHOW VARIABLES LIKE var;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Helpful Links

SET and **RESET**

1.13.19.11 SHUTDOWN

Description

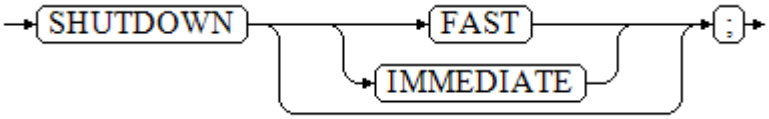
SHUTDOWN shuts down the currently connected database node.

Precautions

- Only the administrator can run this command.
- The distributed database does not support the **SHUTDOWN** command.

Syntax

```
SHUTDOWN [FAST | IMMEDIATE];
```



Parameters

FAST | IMMEDIATE

FAST: Rolls back all active transactions, forcibly disconnects the client, and shuts down the database node without waiting for the client to disconnect.

IMMEDIATE: Shuts down the server forcibly. Fault recovery will occur on the next startup.

If the shutdown mode is not specified, the default value **FAST** is used.

Examples

```
-- Shut down the current database node.
gaussdb=# SHUTDOWN;

-- Shut down the current database node in fast mode.
gaussdb=# SHUTDOWN FAST;
```

1.13.19.12 START TRANSACTION

Description

START TRANSACTION starts a transaction. If the isolation level or read/write mode is specified, a new transaction will have those characteristics. You can also specify them using **SET TRANSACTION**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

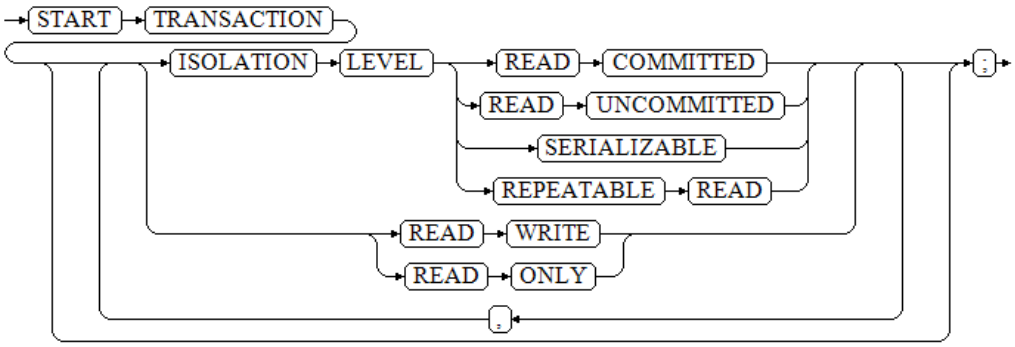
Precautions

When START TRANSACTION is used, **autocommit** remains disabled and is not restored to the initial state until you run COMMIT or ROLLBACK to end a transaction.

Syntax

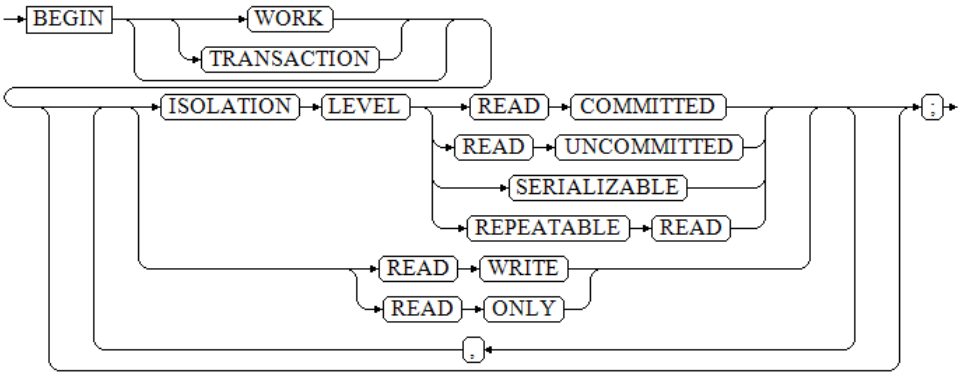
Format 1: START TRANSACTION

```
START TRANSACTION
 [
  {
    ISOLATION LEVEL { READ COMMITTED | READ UNCOMMITTED | SERIALIZABLE | REPEATABLE READ }
    | { READ WRITE | READ ONLY }
  } [, ...]
];
```



Format 2: BEGIN

```
BEGIN [ WORK | TRANSACTION ]
 [
  {
    ISOLATION LEVEL { READ COMMITTED | READ UNCOMMITTED | SERIALIZABLE | REPEATABLE READ }
    | { READ WRITE | READ ONLY }
  } [, ...]
];
```



Parameters

- **WORK | TRANSACTION**

  Serve as optional keywords in the BEGIN statement. However, these keywords currently have no practical effect.

- **ISOLATION LEVEL**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the transaction isolation level. It determines what data a transaction can read when other concurrent transactions are in progress.

> **NOTE**
>
> The isolation level of a transaction cannot be changed after the first clause (INSERT, DELETE, UPDATE, FETCH, COPY) for modifying data is executed in the transaction.

Value range:

- **READ COMMITTED**: Only committed data can be read; uncommitted data remains invisible. This is the default.
- **READ UNCOMMITTED**: This isolation level behaves identically to **READ COMMITTED**.
- **REPEATABLE READ**: The transaction can only read data that was committed before the transaction began. It cannot read uncommitted data or changes committed by other concurrent transactions during its execution.
- **SERIALIZABLE**: Currently, GaussDB does not support this isolation level. Setting this isolation level is equivalent to **REPEATABLE READ**.

- **READ WRITE | READ ONLY**

  Specifies the transaction access mode (**READ WRITE** or **READ ONLY**).

Examples

```
-- Create a schema.
gaussdb=# CREATE SCHEMA tpcds;

-- Create the tpcds.reason table.
gaussdb=# CREATE TABLE tpcds.reason (c1 int, c2 int);

-- Start a transaction in default mode.
gaussdb=# START TRANSACTION;
gaussdb=# SELECT * FROM tpcds.reason;
gaussdb=# END;

-- Start a transaction in default mode.
gaussdb=# BEGIN;
gaussdb=# SELECT * FROM tpcds.reason;
gaussdb=# END;

-- Start a transaction with the isolation level being READ COMMITTED and access mode being READ WRITE.
gaussdb=# START TRANSACTION ISOLATION LEVEL READ COMMITTED READ WRITE;
gaussdb=# SELECT * FROM tpcds.reason;
gaussdb=# COMMIT;

-- Drop the table.
gaussdb=# DROP TABLE tpcds.reason;

-- Drop the schema.
gaussdb=# DROP SCHEMA tpcds CASCADE;
```

Helpful Links

COMMIT | END, **ROLLBACK**, and **SET TRANSACTION**

1.13.20 T

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.20.1 TIMECAPSULE TABLE

Description

The TIMECAPSULE TABLE statement restores a table to an earlier state in the event of human or application errors.

The table can flash back to a past point in time, depending on the source version of the data stored in the system. In addition, GaussDB cannot restore a table to an earlier state through DDL operations that has changed the structure of the table.

Precautions

- The TIMECAPSULE TABLE statement can be used to flash back the data of the source version or the data from the recycle bin.
  - TO TIMESTAMP and TO CSN can flash a table back to an earlier version. Currently, only Ustore is supported.
  - The recycle bin records the objects dropped or truncated by running DROP and TRUNCATE. TO BEFORE DROP and TO BEFORE TRUNCATE can flash data back from the recycle bin. Currently, the Ustore and Astore are supported.
- The following object types do not support flashback: system catalogs, DFS tables, global temporary tables, local temporary tables, unlogged tables, sequence tables, and encrypted tables.
- Flashback is not supported for tables that contain user-defined types.
- After flashback is enabled, tables in the recycle bin can be backed up but cannot be restored.
- If a statement (DDL, DCL, or VACUUM FULL) that modifies the table definition or affects physical storage is executed between the flashback point and the current point, the flashback fails.
- To run DROP, you must have the CREATE or USAGE permission on the schema to which the junk object belongs, and you must be the owner of the schema or the owner of the junk object.

  To run TRUNCATE, you must have the CREATE or USAGE permission on the schema to which the junk object belongs, and you must be the owner of the schema or the junk object. In addition, you must have the TRUNCATE permission on the junk object.
- Scenarios or tables that do not support DROP or TRUNCATE:
  - Scenario where the recycle bin is disabled (**enable_recyclebin** is set to **off**)
  - The system is in the maintenance state (**xc_maintenance_mode** is set to **on**).
  - The version is upgraded from an unsupported baseline version to a supported version and is in the upgrade observation period.
  - Scenario where multiple objects are deleted (The **DROP** or **TRUNCATE TABLE** command is executed to delete multiple objects at the same time.)
  - System catalogs, DFS tables, global temporary tables, local temporary tables, unlogged tables, sequence tables, and encrypted tables

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Objects in the recycle bin are cleared and do not support DROP or TRUNCATE FLASHBACK. The **recyclebin_retention_time** parameter specifies the retention period of objects in the recycle bin.
  - Scaling is not supported. During scale-out and redistribution, data in the recycle bin is forcibly dropped. During scale-out, dropped objects are not moved to the recycle bin.
  - The recycle bin does not support write operations such as DML, DCL, and DDL, and does not support DQL query operations.
  - If a statement (such as DDL, DCL, VACUUM FULL, and partition adding/deleting/splitting/merging) that modifies the table definition or affects physical files are executed between the TRUNCATE TABLE and TRUNCATE flashback operations, the flashback fails.
  - When a table is entirely deleted or truncated, partitions are moved to the recycle bin along with the entire table. A single deleted partition cannot be moved to the recycle bin. This avoids data consistency damage.
  - If there are residual online indexes (a node that is not created exists and the index cannot be found on the node), an error is reported when a table is dropped. You need to clear the residual data to successfully drop the table and place the object in the recycle bin.
  - If the object on which the table depends is an external object (for example, a composite object or a user-defined object), the table is physically deleted and is not moved to the recycle bin.

- DROP FLASHBACK constraints

  You can specify either the original user-defined name of the table or the system-generated name assigned to the object when it was dropped.

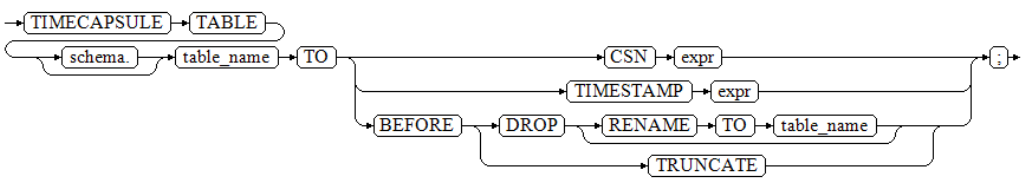
  - System-generated recycle bin object names are unique. Therefore, if you specify the system-generated name, the database retrieves that specified object. To see the content in your recycle bin, run **SELECT * FROM gs_recyclebin;**.
  - If you specify a user-specified name and the recycle bin contains more than one object of that name, the database retrieves the object that was moved to the recycle bin most recently. To retrieve a table of an earlier version, perform the following steps:
    - Specify the system-generated recycle bin name of the table you want to retrieve.
    - Run the **TIMECAPSULE TABLE ... TO BEFORE DROP** statement until you retrieve the table you want.
  - During table restoration, the dropped table along with its original base table name is restored. The subobject names of the table remain the same as those in the recycle bin. You can run the DDL command to manually change the names of subobjects as required.
  - If a table has default values that reference sequences and user-defined functions, flashing back the dropped table will succeed, but the default values will not be restored.
  - Similarly, if a table is referenced by views, dropping the table will cascadingly delete the views. Therefore, flashing back the dropped table will succeed, but the views will not be restored.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - The recycle bin does not support write operations such as DML, DCL, and DDL, and does not support DQL query operations (supported in later versions).
  - The **recyclebin_retention_time** parameter specifies the retention period of objects in the recycle bin. The objects will be automatically deleted after the retention period expires.
- TRUNCATE FLASHBACK constraints
  - After TRUNCATE FLASHBACK is performed, the statistics remain unchanged and are still displayed as **0**. You can modify the statistics by manual ANALYZE during off-peak hours (to reduce the impact on performance).
  - RENAME TO supports only the DROP FLASHBACK operation to specify a new name for the retrieval table, but does not support TRUNCATE FLASHBACK.
  - TRUNCATE FLASHBACK cannot span statements that affect the table definition or physical storage. Otherwise, an error is reported. If a statement (such as DDL, DCL, VACUUM FULL, and partition adding/deleting/splitting/merging) that modifies the table definition or affects physical storages are executed between the flashback point and current point, the flashback fails. The error message "ERROR: The table definition of %s has been modified." is displayed when flashback is performed on a table where DDL operations have been performed. The error message "ERROR: recycle object %s desired does not exist" is displayed when flashback is performed on a table where DDL operations, such as changing namespaces and table names, have been performed.
- When VACUUM is used to update database-level **frozenXid**, the value of **rcyfrozenxid64** in the recycle bin is referenced. If there are tables that are not cleared for a long time in the recycle bin, **frozenXid** may fail to be updated, affecting Clog recycling (Clog files occupy a small amount of storage space and have little impact in actual scenarios). You can clear the tables that do not need to be flashed back in the recycle bin to update **frozenXid**.
- If the recycle bin has been enabled before the upgrade from a version earlier than 503.0 to the latest version, ensure that no partitioned table object exists in the recycle bin before the upgrade. Otherwise, the upgrade will fail.
- During the upgrade from a version earlier than 503.0 to the latest version, no matter whether a partitioned table is in the upgrade uncommitted or upgrade committed state, an error is reported when a flashback table or flashback query operation is performed, indicating that the table structure is changed.

Syntax

```
TIMECAPSULE TABLE
   [schema.]table_name TO { CSN expr | TIMESTAMP expr | BEFORE { DROP [RENAME TO table_name] |
TRUNCATE } };
```



Parameters

- **schema**

  Specifies a schema containing the table to be flashed back. If this parameter is not specified, the current schema is used.

- **table_name**

  Specifies a table name.

- **TO CSN**

  Specifies the CSN corresponding to the time point when the table is to be flashed back. *expr* must be a number representing a valid CSN.

  Note: In the GTM-free scenario, each node uses the local CSN and does not have a globally unified CSN. Therefore, flashback in TO CSN mode is not supported.

- **TO TIMESTAMP**

  Specifies a timestamp value corresponding to the point in time to which you want to flash back the table. The result of *expr* must be a valid past timestamp (convert a string to a time type using the **TO_TIMESTAMP** function). The table will be flashed back to a time within approximately 3 seconds of the specified timestamp.

  Note: When the flashback point is too old, the source version cannot be obtained because it is recycled. As a result, the flashback fails and the error message "Restore point too old" is displayed.

- **TO BEFORE DROP**

  Retrieves dropped tables and their subobjects from the recycle bin.

- **RENAME TO**

  Specifies a new name for the table retrieved from the recycle bin.

- **TO BEFORE TRUNCATE**

  Flashes back to the point in time before the TRUNCATE operation.

Examples

- **Flashing data back to a specified time**

  You need to set the **undo_retention_time** parameter to set the retention period of old undo logs.

  For details about how to use the parameters, contact an administrator.

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE tbl_test(c1 int, c2 int) with(storage_type = ustore) DISTRIBUTE BY
REPLICATION;
gaussdb=# INSERT INTO tbl_test VALUES (1,1),(2,2),(3,3);

-- Query the current time and next_csn on all global nodes.
gaussdb=# SELECT now();
              now
-------------------------------
 2023-11-27 17:06:34.840698+08
(1 row)

gaussdb=# SELECT int8in(xidout(next_csn)) FROM gs_get_next_xid_csn();
 int8in
--------
  25391
  25391
  25391
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 25391
 25391
 25391
(6 rows)

-- Modify data.
gaussdb=# UPDATE tbl_test SET c1=111, c2=222 WHERE c1=1;

-- Query data and flash the data back to the state before UPDATE.
gaussdb=# SELECT * FROM tbl_test;
 c1  | c2
-----+-----
 111 | 222
   2 |   2
   3 |   3
(3 rows)
gaussdb=# TIMECAPSULE table tbl_test TO TIMESTAMP to_timestamp('2023-11-27
17:06:34.840698','YYYY-MM-DD HH24:MI:SS.FF');

-- You can also use the following SQL statements:
gaussdb=# TIMECAPSULE table tbl_test TO CSN 25391;

gaussdb=# SELECT * FROM tbl_test;
 c1 | c2
----+----
  2 |  2
  3 |  3
  1 |  1
(3 rows)

-- Drop.
gaussdb=# DROP TABLE tbl_test PURGE;
```

- **Flashing back data from the recycle bin**

  Prerequisites:

  – The **enable_recyclebin** parameter has been set to enable the recycle bin. For details about how to use the parameter, contact an administrator.

  – The **recyclebin_retention_time** parameter has been set to specify how long objects will remain in the recycle bin before being automatically deleted. Contact an administrator for the parameter usage.

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE tbl_test1(c1 int, c2 varchar(10))with(storage_type = ustore);
gaussdb=# INSERT INTO tbl_test1 VALUES (1,'AAA'),(2,'BBB');

-- Drop the table.
gaussdb=# DROP TABLE tbl_test1;

-- Flash back to the time before the table is deleted.
gaussdb=# TIMECAPSULE TABLE tbl_test1 TO BEFORE DROP;

-- Query data.
gaussdb=# SELECT * FROM tbl_test1;
 c1 | c2
----+-----
  1 | AAA
  2 | BBB
(2 rows)

-- Drop a table. (The PURGE parameter is added to delete the table data from the recycle bin.)
gaussdb=# DROP TABLE tbl_test1 PURGE;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.20.2 TRUNCATE

Description

TRUNCATE quickly removes all rows from a database table.

It has the same effect as an unqualified DELETE on each table, but TRUNCATE is faster because it does not actually scan the tables. This is most useful on large tables.

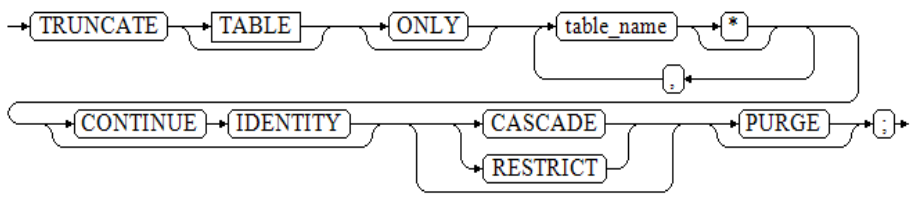
Precautions

- The owner of a table, users granted the TRUNCATE permission on the table, or users granted the TRUNCATE ANY TABLE permission can truncate the specified table. When separation of duties is disabled, system administrators have this permission by default.
- TRUNCATE TABLE has the same function as a DELETE statement with no WHERE clause, emptying a table.
- TRUNCATE TABLE uses less system and transaction log resources as compared with DELETE.
  - DELETE deletes a row each time, and records the deletion of each row in the transaction log.
  - TRUNCATE TABLE deletes all rows in a table by releasing the data page storing the table data, and records the releasing of the data page only in the transaction log.
- The differences between TRUNCATE, DELETE, and DROP are as follows:
  - TRUNCATE TABLE deletes content, releases space, but does not delete definitions.
  - DELETE TABLE deletes content, but does not delete definitions nor release space.
  - DROP TABLE deletes content and definitions, and releases space.

Syntax

- Truncate the data in a table.

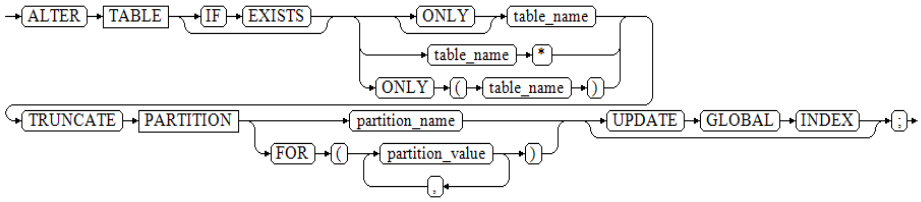
```
TRUNCATE [ TABLE ] [ ONLY ] {table_name [ * ]} [, ... ]
    [ CONTINUE IDENTITY ] [ CASCADE | RESTRICT ] [ PURGE ];
```



- Truncate the data in a partition.

```
ALTER TABLE [ IF EXISTS  ] { [ ONLY  ] table_name
                           | table_name *
                           | ONLY ( table_name )  }
    TRUNCATE PARTITION { partition_name
                       | FOR (  partition_value  [, ...] )  } [ UPDATE GLOBAL INDEX ];
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.



Parameters

- **ONLY**

  If **ONLY** is specified, only the specified table is cleared. If **ONLY** is not specified, the table and all its subtables (if any) are cleared.

- **table_name**

  Specifies the name (optionally schema-qualified) of the target table.

  Value range: an existing table name

- **CONTINUE IDENTITY**

  Does not change the values of sequences. This is the default action.

- **CASCADE | RESTRICT**
  - **CASCADE**: cascadingly truncates all tables added to a group due to CASCADE.
  - **RESTRICT** (default): refuses to truncate if any of the tables have foreign-key references from tables that are not listed in the statement (not supported in distributed scenarios).

- **PURGE**

  Purges table data in the recycle bin by default.

- **partition_name**

  Specifies the partition in the target partitioned table.

  Value range: an existing table name

- **partition_value**

  Specifies the partition key value.

  The value specified by **PARTITION FOR** can uniquely identify a partition.

  Value range: value range of the partition key for the partition to be renamed

  NOTICE

  When the PARTITION FOR clause is used, the entire partition where **partition_value** is located is cleared.

- **UPDATE GLOBAL INDEX**

  Updates all global indexes in the partitioned table to ensure that correct data can be queried using global indexes.

  If this parameter is not used and the GUC parameters **enable_gpi_auto_update** and **enable_drop_truncate_partition_keep_gpi** are set to **off**, all global indexes in the partitioned table will become invalid.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

- Clear table data.

```
-- Create the reason table.
gaussdb=# CREATE TABLE reason (r_reason_sk int,r_reason_id varchar(16),r_reason_desc
varchar(100));

-- Insert multiple records into the table.
gaussdb=# INSERT INTO reason values(1,'AAAAAAAABAAAAAAA','reason 1'),
(5,'AAAAAAAABAAAAAAA','reason 2'),
                                  (15,'AAAAAAAABAAAAAAA','reason 3'), (25,'AAAAAAAABAAAAAAA','reason
4'),
                                  (35,'AAAAAAAABAAAAAAA','reason 5'), (45,'AAAAAAAACAAAAAAA','reason
6');

-- Check the table information. The table size is about 16 KB.
gaussdb=# \d+
                                  List of relations
 Schema |  Name  | Type  | Owner | Size  |             Storage              | Description
--------+--------+-------+-------+-------+----------------------------------+-------------
 public | reason | table | omm   | 16 kB | {orientation=row,compression=no} |
(1 row)

-- Run the DELETE statement without the WHERE condition to clear data in the table and check the
table size.
gaussdb=# DELETE FROM reason;
gaussdb=# \d+
                                  List of relations
 Schema |  Name  | Type  | Owner | Size  |             Storage              | Description
--------+--------+-------+-------+-------+----------------------------------+-------------
 public | reason | table | omm   | 16 kB | {orientation=row,compression=no} |
(1 row)

-- Run the TRUNCATE statement to clear the reason table and check the table size.
gaussdb=# TRUNCATE TABLE reason;
gaussdb=# \d+
                                  List of relations
 Schema |  Name  | Type  | Owner |  Size   |             Storage              | Description
--------+--------+-------+-------+---------+----------------------------------+-------------
 public | reason | table | omm   | 0 bytes | {orientation=row,compression=no} |
(1 row)

-- Drop the table.
gaussdb=# DROP TABLE reason;
```

- Clear data of a partitioned table.

```
-- Create a partitioned table.
gaussdb=# CREATE TABLE reason_p(
    r_reason_sk integer,
    r_reason_id character(16),
    r_reason_desc character(100)
)PARTITION BY RANGE (r_reason_sk)(
    partition p_05_before values less than (05),
    partition p_15 values less than (15),
    partition p_25 values less than (25),
    partition p_35 values less than (35),
    partition p_45_after values less than (MAXVALUE)
);

-- Insert data.
gaussdb=# INSERT INTO reason_p values(1,'AAAAAAAABAAAAAAA','reason 1'),
(5,'AAAAAAAABAAAAAAA','reason 2'),
                                   (15,'AAAAAAAABAAAAAAA','reason 3'), (25,'AAAAAAAABAAAAAAA','reason
4'),
                                   (35,'AAAAAAAABAAAAAAA','reason 5'), (45,'AAAAAAAACAAAAAAA','reason
6');

-- Clear the p_05_before partition.
gaussdb=# ALTER TABLE reason_p TRUNCATE PARTITION p_05_before UPDATE GLOBAL INDEX;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Clear the partitioned table.
gaussdb=# TRUNCATE TABLE reason_p;

-- Delete the reason_p table.
gaussdb=# DROP TABLE reason_p;
```

1.13.21 U

1.13.21.1 UPDATE

Description

Updates specified columns in all rows that meet the specified conditions. You can use WHERE to declare conditions. All columns specified by the SET clause will be updated. Other columns retain their original values.

Precautions

- The owner of a table, users with the UPDATE permission on the table, or users with the UPDATE ANY TABLE permission can update data in the table. When separation of duties is disabled, system administrators have this permission by default.
- UPDATE..... LIMIT row_count supports only some scenarios of execution plan pushdown. (Table replication is not supported.) The prerequisites are that the filter criteria must contain the equivalent distribution key and the filter criteria should be relatively simple. Do not use forcible type conversion. If the command fails to be executed, simplify the filter criteria.
- You must have the SELECT permission on all tables involved in the expressions or conditions.
- The distribution keys can be updated only when **enable_update_distkey** is set to **on** and constraints are met. Otherwise, distribution keys do not support the UPDATE operation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

The constraints on updating distribution keys are as follows:

- Distribution keys can be updated only when **enable_update_distkey** is set to **on**.
- The UPDATE statement cannot be pushed down to DNs for execution. The PGXC plan is directly generated and is not changed based on the values before and after the distribution key is updated.
- Tables with row-level UPDATE TRIGGER are not supported. Otherwise, the execution fails and an error message is displayed. The row-level INSERT/DELETE TRIGGER does not take effect, and the update-statement-level TRIGGER is executed normally.
- Concurrent update of the same row is not supported. You need to obtain a lock first. The result (returning **0** or reporting an error) of obtaining a lock on DNs depends on the setting of the GUC parameter **concurrent_dml_mode**. An error will be reported in the following cases: (1) The error message "update distribute column conflict" is displayed. (2) When the time spent for obtaining a lock exceeds the threshold, an error message is displayed, indicating that the lock times out.
- Tables with global secondary indexes (GSIs) are not supported. Otherwise, an error is reported.
- Only hash distribution is supported. LIST/RANGE distributed tables are not supported. Otherwise, an error is reported.
- MERGE INTO and UPSERT cannot be used to update distribution keys. Otherwise, an error is reported.
- GTM_FREE is not supported. Otherwise, an error is reported.
- UPDATE RETURNING is not supported. Otherwise, an error is reported.
- Statements with joined tables are not supported. Otherwise, an error is reported.
- UPDATE+LIMIT is not supported. Otherwise, an error is reported.
- In ORA-compatible and MYSQL-compatible modes, multiple partitions cannot be deleted at a time. Therefore, when the DELETE statements are reversely parsed during distribution key update, multiple partitions are not involved. Data consistency is ensured by the condition **tableoid**.

- For row-store replication tables, UPDATE can be performed only in the following two scenarios:
  - Scenarios with PRIMARY KEY constraints.
  - Scenarios where the execution plan can be pushed down.
- For the UPDATE statement whose subquery is a STREAM plan, the same row cannot be concurrently updated.
- You cannot modify the database character encoding by updating a system catalog. Such operation will cause exceptions in existing data or other operations. If you need to change the character encoding of a database, follow the database switching process to migrate data.
- In high-concurrency scenarios, when a large number of UPDATE operations are performed on an ultra-wide table, if frequent DDL operations are performed in seconds, the UPDATE operation may take hours. Minute-level DDL operations do not significantly affect performance.

Syntax

```
[ WITH [ RECURSIVE ] with_query [, ...] ]
UPDATE [/*+ plan_hint */] [ ONLY ] {table_name [ partition_clause ] | subquery | view_name} [ * ] [ [ AS ] alias ]
SET {column_name = { expression | DEFAULT }
    |( column_name [, ...] ) = {( { expression | DEFAULT } [, ...] ) |sub_query }}[, ...]
```

```
   [ FROM from_list] [ WHERE condition | WHERE CURRENT OF cursor_name ]
   [ ORDER BY {expression [ [ ASC | DESC | USING operator ]
[ LIMIT { count } ]
   [ RETURNING {*
            | {output_expression [ [ AS ] output_name ]} [, ...] }];

where sub_query can be:
SELECT [ ALL | DISTINCT [ ON ( expression [, ...] ) ] ]
{ * | {expression [ [ AS ] output_name ]} [, ...] }
[ FROM from_item [, ...] ]
[ WHERE condition  ]
[ GROUP BY grouping_element [, ...] ]
[ HAVING condition [, ...] ]
[ ORDER BY {expression [ [ ASC | DESC | USING operator ] | nlssort_expression_clause ] [ NULLS { FIRST |
LAST } ]} [, ...] ]
[ LIMIT { [offset,] count | ALL } ]
```

- The subquery **with_query** is as follows:

  ```
  with_query_name [ ( column_name [, ...] ) ] AS [ [ NOT ] MATERIALIZED ] ( {select | values | insert |
  update | delete} )
  ```

- For subquery:

  ```
  [ WITH [ RECURSIVE ] with_query [, ...] ]
  SELECT [/*+ plan_hint */] [ ALL ]
  { * | {expression [ [ AS ] output_name ]} [, ...] }
  [ into_option ]
  [ FROM from_item [, ...] ]
  [ WHERE condition ]
  [ [ START WITH condition ] CONNECT BY [NOCYCLE] condition [ ORDER SIBLINGS BY expression ] ]
  [ ORDER BY {expression [ [ ASC | DESC | USING operator ] | nlssort_expression_clause ] [ NULLS
  { FIRST | LAST } ]} [, ...] ]
  [ FETCH { FIRST | NEXT } [ count ] { ROW | ROWS } ONLY ]
  [ into_option ];
  ```

- The specified subquery source **from_item** is as follows:

  ```
  {[ ONLY ] {table_name | view_name} [ * ] [ partition_clause ] [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
  |( select ) [ AS ] alias [ ( column_alias [, ...] ) ]
  |with_query_name [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
  |from_item [ NATURAL ] join_type from_item [ ON join_condition | USING ( join_column [, ...] ) ]}
  ```

- For partition_clause:

  ```
  PARTITION { ( partition_name [, ...] ) | FOR ( partition_value [, ...] ) }
  ```

Parameters

- **WITH [RECURSIVE] with_query [, ...]**

  Specifies one or more subqueries that can be referenced by name in the main query, which is equivalent to a temporary table. This subquery statement structure is called the common table expression (CTE) structure. When this structure is used, the execution plan contains the CTE SCAN content.

  If **RECURSIVE** is specified, it allows a SELECT subquery to reference itself by name.

  with_query:

  - **with_query_name** specifies the name of the result set generated by a subquery. Such names can be used to access the result sets of subqueries in a query.
  - **column_name** specifies the column name displayed in the subquery result set.
  - Each subquery can be a SELECT, VALUES, INSERT, UPDATE, or DELETE statement.
  - You can use MATERIALIZED or NOT MATERIALIZED to modify the CTE.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - If **MATERIALIZED** is specified, the WITH query will be materialized, and a copy of the subquery result set is generated. The copy is directly queried at the reference point. Therefore, the WITH subquery cannot be jointly optimized with the SELECT statement trunk (for example, predicate pushdown and equivalence class transfer). In this scenario, you can use **NOT MATERIALIZED** for modification. If the WITH query can be executed as a subquery inline, the preceding optimization can be performed.
  - If the user does not explicitly declare the materialized attribute, comply with the following rules: If the CTE is referenced only once in the SELECT statement trunk to which it belongs and semantically supports inline execution, it will be rewritten as subquery inline execution. Otherwise, the materialized execution will be performed in CTE Scan mode.

- **plan_hint**

  Follows the UPDATE keyword in the /*+ */ format. It is used to optimize the plan of a UPDATE statement block using hints. For details, see "Hint-based Optimization" in *SQL Optimization Guide*. In each statement, only the first **/*+ plan_hint */** comment block takes effect as a hint. Multiple hints can be written.

- **table_name**

  Specifies the name (optionally schema-qualified) of the table to be updated. If **ONLY** is specified before the table name, only matching rows in the specified table are updated. If it is not specified, any matching rows inherited from the table are also updated.

  Value range: existing table names.

  NOTE

  You can use database links to perform operations on remote tables. For details, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.

- **subquery**

  Specifies the subquery to be updated. When a subquery is updated, the subquery is regarded as a temporary view. The **CHECK OPTION** option can be added to the end of the subquery.

  If a subquery contains only one table, data is updated into the table. If a subquery contains multiple tables or has nested relationships, check whether a key-preserved table exists to determine whether data can be updated. For details about key-preserved tables and **WITH CHECK OPTION**, see **CREATE VIEW**.

- **view_name**

  Specifies the target view to be updated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> Pay attention to the following limitations when updating a view or subquery:
>
> - The UPDATE operation can be performed only on columns that directly reference user columns in the base table.
> - The view or subquery must contain at least one updatable column. For details about updatable columns, see **CREATE VIEW**.
> - The view or subquery must not contain the DISTINCT, GROUP BY, HAVING, LIMIT, or OFFSET clause at the top layer.
> - Views and subqueries that contain set operations (UNION, INTERSECT, EXCEPT, and MINUS) at the top layer are not supported.
> - The view or subquery must not contain aggregate functions, window functions, or return set functions (such as array_agg, json_agg, and generate_series) in the target list.
> - Views with BEFORE or AFTER triggers but without INSTEAD OF triggers or INSTEAD rules are not supported.
> - The view or subquery uses only the following types of tables: ordinary tables, temporary tables, global temporary tables, partitioned tables, subpartitioned tables, Ustore tables, and Astore tables.
> - Only one base table can be updated at a time in a multi-table join view or join subquery.
> - Join views or subqueries can update only key-preserved tables. If **CHECK OPTION** is specified, join columns cannot be updated. For details about key-preserved tables, see **CREATE VIEW**.
> - System views cannot be updated.

- **partition_clause**

  Updates a specified partition.

  > **NOTE**
  >
  > If PARTITION specifies multiple partition names, the partition names can be the same. The union set of the partition ranges is used.

- **alias**

  Specifies the alias of the target table.

  Value range: character strings complying with **Identifier Naming Conventions**.

- **column_name**

  Specifies the name of the column to be modified.

  You can reference this column by using the target table alias and column name. Example: **UPDATE foo AS f SET f.col_name = 'namecol'**

  Value range: existing column names.

- **expression**

  Specifies the expression or value to be assigned to the column.

- **DEFAULT**

  Specifies the default value of a column.

  The value is **NULL** if there is no default value.

- **sub_query**

  Specifies a subquery.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

This statement can be executed to update a table with information for other tables in the same database. For details about clauses in the SELECT statement, see **SELECT**.

When a single column is updated, the ORDER BY and LIMIT clauses can be used. When multiple columns are updated, the ORDER BY and LIMIT clauses cannot be used.

NOTE

- When multiple columns are updated using a subquery, if the GUC parameter **enable_shared_set_update** is disabled, the optimizer generates a subplan for each updated column. If the parameter is enabled, multiple columns can share one subplan to improve query efficiency and reduce query optimization and execution overhead. For details about the GUC parameter setting, contact an administrator.
- When a subquery is used to update multiple columns, you are advised not to use asterisks (*) to replace specific column names in the targetlist of the subquery. Instead, specify the names to be updated clearly.
- If the WHERE clause of a subquery in the SET clause uses the + operator, the configuration item **enable_ora_joinop_in_updatestmt** must be added to the compatibility parameter **behavior_compat_options** for the setting to take effect.

- **from_list**

  Specifies a list of table expressions. You can use columns of other tables in the WHERE condition. It is similar to specifying a table list in a FROM clause of a SELECT statement.

  NOTICE

  Note that the target table cannot appear in **from_list**, unless you intend a self-join (in which case it must appear with an alias in **from_list**).

- **condition**

  Specifies an expression that returns a value of the Boolean type. Only rows for which this expression returns **true** are updated. You are advised not to use numeric types such as int for **condition** because such types can be implicitly converted to Boolean values (non-zero values are implicitly converted to **true** and **0** is implicitly converted to **false**), which may cause unexpected results.

- **WHERE CURRENT OF cursor_name**

  When the cursor points to a row in a table, you can use this syntax to update the row.

  cursor_name: specifies the name of a cursor.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- This syntax is not supported by the database in MYSQL-compatible mode.
- This syntax supports only ordinary tables instead of partitioned tables or hash bucket tables.
- This syntax can be used only in stored procedures.
- This syntax cannot be used together with other WHERE conditions.
- This syntax cannot be used together with WITH, USING, ORDER BY, or FROM.
- The SELECT statement corresponding to the CURSOR must be declared as FOR UPDATE.
- The SELECT statement corresponding to the CURSOR supports only a single table. It does not support LIMIT/OFFSET, subqueries, or sublinks.
- The CURSOR declared as FOR UPDATE in a stored procedure cannot be used after being committed or rolled back.
- If the row to which the CURSOR points does not exist, an error is reported (only when UPDATE is used instead of DELETE) in the ORA-compatible mode, indicating that the specified row does not exist. In other compatibility modes, no error is reported.
- When this syntax is used in a replication table, pushdown is not supported. A primary key is required.

- **ORDER BY**

  For details about the keywords, see **SELECT**.

- **LIMIT**

  For details about the keywords, see **SELECT**.

- **RETURNING output_expression**

  Specifies an expression to be computed and returned by the UPDATE statement after each row is updated.

  Value range: any table and table columns listed in FROM. Write * to return all columns.

- **output_name**

  Specifies a name to use for a returned column.

Examples

- Modify all data in the table.

```
-- Create a table named tbl_test1 and insert data into the table.
gaussdb=# CREATE TABLE tbl_test1(id int, info varchar(10));
gaussdb=# INSERT INTO tbl_test1 VALUES (1, 'A'), (2, 'B');

-- Query.
gaussdb=# SELECT * FROM tbl_test1;
 id | info
----+------
  1 | A
  2 | B
(2 rows)

-- Modify the info column of all data in the tbl_test1 table.
gaussdb=# UPDATE tbl_test1 SET info = 'aa';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Query the tbl_test1 table.
gaussdb=# SELECT * FROM tbl_test1;
 id | info
----+------
  1 | aa
  2 | aa
(2 rows)
```

- Modify some data in the table.

```
-- Modify the data whose id is 2 in the tbl_test1 table.
gaussdb=# UPDATE tbl_test1 SET info = 'bb' WHERE id = 2;

-- Query the tbl_test1 table.
gaussdb=# SELECT * FROM tbl_test1;
 id | info
----+------
  1 | aa
  2 | bb
(2 rows)
```

- Modify the data and return the modified data.

```
-- Modify the data whose id is 1 in the tbl_test1 table and specify that the info column should be
returned.
gaussdb=# UPDATE tbl_test1 SET info = 'ABC' WHERE id = 1 RETURNING info;
 info
------
 ABC
(1 row)

UPDATE 1

-- Drop the tbl_test1 table.
gaussdb=# DROP TABLE tbl_test1;
```

- Use a subquery to insert new data based on the existing data when modifying data.

```
-- Create a table.
gaussdb=# CREATE TABLE test_grade (
    sid int,                -- Student ID
    name varchar(50),       -- Name
    score char,             -- Score
    examtime date,          -- Exam time
    last_exam boolean       -- The last exam or not
);

-- Insert data.
gaussdb=# INSERT INTO test_grade VALUES (1,'Scott','A','2008-07-08',1),(2,'Ben','D','2008-07-08',1),
(3,'Jack','D','2008-07-08',1);

-- Query.
gaussdb=# SELECT * FROM test_grade;
 sid | name  | score |  examtime  | last_exam
-----+-------+-------+------------+-----------
   3 | Jack  | D     | 2008-07-08 | t
   1 | Scott | A     | 2008-07-08 | t
   2 | Ben   | D     | 2008-07-08 | t
(3 rows)

-- On August 25, 2008, Ben took a make-up exam and the score is B. You need to change the value of
last_exam to No, and then insert the score of August 25, 2008.
gaussdb=# WITH old_exa AS ( UPDATE test_grade SET last_exam = 0 WHERE sid = 2 AND examtime
= '2008-07-08' RETURNING sid, name )
        INSERT INTO test_grade VALUES ( ( SELECT sid FROM old_exa ), (SELECT name FROM old_exa),
'B', '2008-08-25', 1 );

-- Query.
gaussdb=# SELECT * FROM test_grade;
 sid | name  | score |  examtime  | last_exam
-----+-------+-------+------------+-----------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  3 | Jack  | D     | 2008-07-08 | t
  1 | Scott | A     | 2008-07-08 | t
  2 | Ben   | D     | 2008-07-08 | f
  2 | Ben   | B     | 2008-08-25 | t
(4 rows)

-- Drop.
gaussdb=# DROP TABLE test_grade;
```

- **Updating a view or subquery**

  Example 1: Update a subquery.

```
-- Create a schema.
gaussdb=# CREATE SCHEMA upd_subqry;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = 'upd_subqry';
SET

-- Create tables and insert data into the tables.
gaussdb=# CREATE TABLE t1 (x1 int, y1 int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'x1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# CREATE TABLE t2 (x2 int PRIMARY KEY, y2 int);
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "t2_pkey" for table "t2"
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES (1, 1), (2, 2), (3, 3), (5, 5);
INSERT 0 4
gaussdb=# INSERT INTO t2 VALUES (1, 1), (2, 2), (3, 3), (5, 5);
INSERT 0 4

-- Update t1 through a subquery.
gaussdb=# UPDATE (SELECT * FROM t1) SET y1 = 13 where y1 = 3;
UPDATE 1
gaussdb=# UPDATE (SELECT * FROM t1 WHERE y1 < 2) SET y1 = 12 WHERE y1 = 2;
UPDATE 0

-- Insert a subquery with READ ONLY specified.
gaussdb=# UPDATE (SELECT * FROM t1 WITH READ ONLY) SET y1 = 1 WHERE y1 = 11;
ERROR:  DML operation cannot be performed on a read-only view.

-- Insert a multi-table join subquery.
gaussdb=# UPDATE (SELECT * FROM t1, t2 WHERE x1 = x2) SET y1 = 11 WHERE y2 = 1;
UPDATE 1

-- Insert a multi-table join subquery with CHECK OPTION specified. The join columns x1 and x2
cannot be updated.
gaussdb=# UPDATE (SELECT * FROM t1, t2 WHERE x1 = x2 WITH CHECK OPTION) SET y1 = 1 WHERE
y2 = 1;
UPDATE 1
gaussdb=# UPDATE (SELECT * FROM t1, t2 WHERE x1 = x2 WITH CHECK OPTION) SET x1 = 6 WHERE
y2 = 5;
ERROR:  This operation is not allowed on virtual column.

-- Drop a schema.
gaussdb=# RESET CURRENT_SCHEMA;
RESET
gaussdb=# DROP SCHEMA upd_subqry CASCADE;
NOTICE:  drop cascades to 2 other objects
DETAIL:  drop cascades to table upd_subqry.t1
drop cascades to table upd_subqry.t2
DROP SCHEMA
```

  Example 2: Update a view.

```
-- Create a schema.
gaussdb=# CREATE SCHEMA upd_view;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = 'upd_view';
SET
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create tables and insert data into the tables.
gaussdb=# CREATE TABLE t1 (x1 int, y1 int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'x1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# CREATE TABLE t2 (x2 int PRIMARY KEY, y2 int);
NOTICE:  CREATE TABLE / PRIMARY KEY will create implicit index "t2_pkey" for table "t2"
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES (1, 1), (2, 2), (3, 3), (5, 5);
INSERT 0 4
gaussdb=# INSERT INTO t2 VALUES (1, 1), (2, 2), (3, 3), (5, 5);
INSERT 0 4

-- Create a single table view.
gaussdb=# CREATE VIEW v_upd1 AS SELECT * FROM t1;
CREATE VIEW
gaussdb=# CREATE VIEW v_upd_read AS SELECT * FROM t1 WITH READ ONLY;
CREATE VIEW

-- Update t1 through a view.
gaussdb=# UPDATE v_upd1 SET y1 = 13 where y1 = 3;
UPDATE 1
gaussdb=# UPDATE v_upd_read SET y1 = 1 WHERE y1 = 11;
ERROR:  DML operation cannot be performed on a read-only view.

-- Create a multi-table view.
gaussdb=# CREATE VIEW vv_upd AS SELECT * FROM t1, t2 WHERE x1 = x2;
CREATE VIEW
gaussdb=# CREATE VIEW vv_upd_wco AS SELECT * FROM t1, t2 WHERE x1 = x2 WITH CHECK OPTION;
CREATE VIEW

-- Update t1 through the join view.
gaussdb=# UPDATE vv_upd SET y1 = 1 WHERE y2 = 1;
UPDATE 1
gaussdb=# UPDATE vv_upd_wco SET x1 = 6 WHERE y2 = 5;
ERROR:  This operation is not allowed on virtual column.

-- Drop a schema.
gaussdb=# RESET CURRENT_SCHEMA;
RESET
gaussdb=# DROP SCHEMA upd_view CASCADE;
NOTICE:  drop cascades to 6 other objects
DETAIL:  drop cascades to table upd_view.t1
drop cascades to table upd_view.t2
drop cascades to view upd_view.v_upd1
drop cascades to view upd_view.v_upd_read
drop cascades to view upd_view.vv_upd
drop cascades to view upd_view.vv_upd_wco
DROP SCHEMA
ERROR:  virtual column not allowed here
```

1.13.21.1.1 Pseudo-update of Distribution Keys

Description

When the UPDATE statement is executed to update a distribution key, GaussDB checks the equivalent distribution keys and eliminates the distribution key information that is not actually updated through the query rewriting mechanism. This prevents the SELECT FOR UPDATE syntax visibility error caused by the deletion or rebuild of distribution keys that are not actually updated. This function is controlled by the GUC parameter **enable_remove_redundant_update_col**. This function is enabled when the parameter is set to **on**.

After this function is enabled, the actual effect is as follows:

Original user statement:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
UPDATE TABLE SET distkey=value,amount=10000 WHERE distkey=value;
```

Statement after query rewriting:

```
UPDATE TABLE SET amount=10000 WHERE distkey=value;
```

NOTE

Before this function is enabled, the following error message is displayed for the pseudo-update of distribution keys (no additional error information is displayed):

```
ERROR:  Distributed key cannot be updated in the current version.
```

Limitations

- Only ORA compatibility is supported.
- Only hash distributed tables are supported.
- Syntax constraints on pseudo-update of distribution keys:
  - Only the UPDATE syntax is supported. The MERGE INTO, REPLACE INTO, or UPSERT syntax is not supported.
  - Only scenarios where actual values can be calculated during query rewriting are supported, including:
    - Common constants: data passed by values such as numbers and strings.
    - Parameter types: constant values (such as numbers and strings) passed in PBE packets.
    - Function types: immutable functions (stable and volatile functions are not supported).
    - Operator types: common operators (such as + - * /), with constant inputs.
    - Boolean expression types: OR, AND, and NOT, with constant inputs.
    - Explicit type conversion: explicit type conversion of constants ('1.23'::numeric).
    - Condition expression types: CASE WHEN, with constant parameters. NVL/COALESCE, with constant parameters.
    - NULL value.
  - Only equality expressions are supported. Ranges specified by greater than, less than, between, and in are not supported.
  - Subquery scenarios are not supported.
  - The combination of UPDATE and CTE is not supported.
  - Views cannot be updated.
- In scenarios where pseudo-update of distribution keys can be eliminated, the distribution keys do not need to be updated after elimination. In this case, constraints listed in **NOTE** do not need to be followed.
- Pseudo-update of distribution keys forcibly selects a Cplan. The priorities of plan selection are as follows:

  /*+ use_gplan*/ > Pseudo-update of distribution keys > **plan_cache_mode**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

- Example 1: Pseudo-update a distribution key of the nvarchar2 type.

```
-- Create a test database.
gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA';
gaussdb=# \c ora_db

-- Create a hash distributed table.
ora_db=# CREATE TABLE test_hash(
    id       nvarchar2(10),
    name     VARCHAR(20),
    province VARCHAR(60),                    -- Province
    country   VARCHAR(30) DEFAULT 'Country1'        -- Country
)DISTRIBUTE BY HASH(id);
ora_db=# INSERT INTO test_hash VALUES ('0000001', 'Bob',   'City1', 'Country1'),
    ('0000002', 'Jack',  'City2',  'Country1'),
    ('0000003', 'Scott', 'City2',  'Country1');

-- An error is reported in the original scenario.
ora_db=# UPDATE test_hash SET id = '0000001', name = 'bob1' WHERE id = '0000001';
ERROR:  Distributed key cannot be updated in the current version.

-- No error is reported after the function is enabled.
ora_db=# SET enable_remove_redundant_update_col=on;
ora_db=# UPDATE test_hash SET id = '0000001', name = 'bob1' WHERE id = '0000001';
UPDATE 1

-- Display the plan (the projected column id = '0000001' in the SET clause is successfully eliminated).
ora_db=# EXPLAIN VERBOSE UPDATE test_hash SET id = '0000001', name = 'bob1' WHERE id =
'0000001';
                                        QUERY PLAN
-----------------------------------------------------------------------------------------------------------------
 Data Node Scan on "__REMOTE_FQS_QUERY__"  (cost=0.00..0.00 rows=0 width=0)
   Node/s: datanode1
   Remote query: UPDATE public.test_hash SET name = 'bob1'::character varying WHERE id::text =
'0000001'::text
(3 rows)

-- Delete the test database. Replace postgres with the actual initial database name.
ora_db=# \c postgres
gaussdb=# DROP DATABASE ora_db;
```

- Example 2: Pseudo-update multiple distribution keys.

```
-- Create a test database.
gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA';
gaussdb=# \c ora_db

-- Enable the function.
ora_db=# SET enable_remove_redundant_update_col=on;

-- Create a hash distributed table.
ora_db=# CREATE TABLE test_hash_multi_distcol(
    id       int,
    name     VARCHAR(20),
    province VARCHAR(60),                    -- Province
    country   VARCHAR(30) DEFAULT 'Country1'        -- Country
)DISTRIBUTE BY HASH(id,name);
ora_db=# INSERT INTO test_hash_multi_distcol VALUES (1, 'Bob',   'City1', 'Country1'),
    (2, 'Jack',  'City1',  'Country1'),
    (3, 'Scott', 'City2',  'Country1');

-- No error is reported after the function is enabled.
ora_db=# UPDATE test_hash_multi_distcol SET id = 1,name='Bob', province = 'City1' WHERE id = 1
and name='Bob';
UPDATE 1

-- Display the plan.
ora_db=# EXPLAIN VERBOSE UPDATE test_hash_multi_distcol SET id = 1,name='Bob', province =
'City1' WHERE id = 1 AND name='Bob';
                                            QUERY PLAN
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
------------------------------------------------------------------------------------------------------------------------------------------------
------------------------
 Data Node Scan on "__REMOTE_FQS_QUERY__"  (cost=0.00..0.00 rows=0 width=0)
   Node/s: datanode3
   Remote query: UPDATE public.test_hash_multi_distcol SET province = 'City1'::character varying
WHERE id = 1 AND name::text = 'Bob'::text
(3 rows)

-- Delete the test database. Replace postgres with the actual initial database name.
ora_db=# \c postgres
gaussdb=# DROP DATABASE ora_db;
```

- Example 3: Pseudo-update a distribution key in the PBE scenario.

```
-- Create a test database.
gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA';
gaussdb=# \c ora_db

-- Enable the function.
ora_db=# SET enable_remove_redundant_update_col=on;

-- Create a hash distributed table.
ora_db=# CREATE TABLE test_hash_int(
    id       int,
    name     VARCHAR(20),
    province VARCHAR(60),                    -- Province
    country   VARCHAR(30) DEFAULT 'Country1'        -- Country
)DISTRIBUTE BY HASH(id);
ora_db=# INSERT INTO test_hash_int VALUES (1, 'Bob',   'City1', 'Country1'),
    (2, 'Jack',  'City2',  'Country1'),
    (3, 'Scott', 'City2',  'Country1');

ora_db=# SELECT a.count,b.node_name
    FROM (SELECT COUNT(*) AS count, xc_node_id FROM test_hash_int GROUP BY xc_node_id) a,
    pgxc_node b
WHERE a.xc_node_id=b.node_id ORDER BY a.count DESC;
 count | node_name
-------+-----------
     2 | datanode2
     1 | datanode1
(2 rows)

-- No error is reported after the function is enabled.
ora_db=# PREPARE p1 AS UPDATE test_hash_int SET id = $1, name = $2 WHERE id = $3;
PREPARE
ora_db=# EXECUTE p1(1,'bob1',1);
UPDATE 1

-- Delete the test database. Replace postgres with the actual initial database name.
ora_db=# \c postgres
gaussdb=# DROP DATABASE ora_db;
```

- Example 4: Implicit type conversion between timestamp and text is not supported.

```
-- Create a test database.
gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA';
gaussdb=# \c ora_db

-- Enable the function.
ora_db=# SET enable_remove_redundant_update_col = on;

-- Create a hash distributed table.
ora_db=# CREATE TABLE distcol_update_timestamp(c1 timestamp(6),c2 int,c3 text) DISTRIBUTE BY
hash(c1);
ora_db=# INSERT INTO distcol_update_timestamp VALUES ('2026-03-05
09:25:19.411077+08',1,'gaussdb'),
    ('2026-03-05 09:25:19.411077+08',2,'gaussdb'),
    ('2026-03-05 09:25:20.411077+08',3,'gaussdb'),
    ('2026-03-05 09:25:21.411077+08',4,'gaussdb'),
    ('2026-03-05 09:25:22.411077+08',5,'gaussdb'),
    ('2026-03-06 09:25:22.411077+08',6,'gaussdb'),
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    ('2026-03-07 09:25:22',7,'gaussdb');
ora_db=# CREATE INDEX ON distcol_update_timestamp(c1);

ora_db=# EXPLAIN VERBOSE UPDATE distcol_update_timestamp SET c1 = '2026-03-05
09:25:20.411077+08',c2 = -3
WHERE c1 = '2026-03-05 09:25:20.411077+08'::varchar2;
                                                                QUERY PLAN
------------------------------------------------------------------------------------------------------------------------------------------
 Data Node Scan on "__REMOTE_FQS_QUERY__"  (cost=0.00..0.00 rows=0 width=0)
   Node/s: datanode2
   Remote query: UPDATE public.distcol_update_timestamp SET c2 = (-3) WHERE c1 = '2026-03-05
09:25:20.411077+08'::character varying::timestamp without time zone
(3 rows)

ora_db=# EXPLAIN VERBOSE UPDATE distcol_update_timestamp SET c1 = '2026-03-05
09:25:20.411077+08',c2 = -3
WHERE c1 = '2026-03-05 09:25:20.411077+08';
                                                             QUERY PLAN
------------------------------------------------------------------------------------------------------------------------------------------
 Data Node Scan on "__REMOTE_FQS_QUERY__"  (cost=0.00..0.00 rows=0 width=0)
   Node/s: datanode2
   Remote query: UPDATE public.distcol_update_timestamp SET c2 = (-3) WHERE c1 = '2026-03-05
09:25:20.411077'::timestamp without time zone
(3 rows)

ora_db=# EXPLAIN VERBOSE UPDATE distcol_update_timestamp SET c1 = '2026-03-05
09:25:20.411077+08',c2 = -3
WHERE c1 = '2026-03-05 09:25:20.411077+08'::text;
ERROR:  Distributed key cannot be updated in the current version.

-- There are two implicit type conversion functions, one is stable and the other is immutable. You can
see from the provolatile column that the implicit type conversion from text to timestamp is marked
as s, which is not supported.
ora_db=# SELECT * FROM pg_proc WHERE proname IN ('varchar_timestamp','text_timestamp');
      proname      | pronamespace | proowner | prolang | procost | prorows | provariadic | protransform |
proisagg | proiswindow | prosecdef | proleakproof | proisstrict | proretset | provolatile |
pronargs | pronargdefaults | prorettype | proargtypes | proallargtypes | proargmodes | proargnames |
proargdefaults |      prosrc       | probin | proconfig | proacl | prodefaultargpos | fencedmode
| proshippable | propackage | prokind | proargsrc | propackageid | proisprivate | proargtypesext |
prodefaultargposext | allargtypes | allargtypesext
------------------+--------------+----------+---------+---------+---------+-------------+--------------+----------
+-------------+-----------+--------------+-------------+-----------+-------------+-
---------+-----------------+------------+-------------+----------------+-------------+-------------+----------------
+-------------------+--------+-----------+--------+------------------+------------
+--------------+------------+---------+-----------+--------------+--------------+----------------
+----------------------+-------------+----------------
 text_timestamp    |          11 |       10 |      12 |       1 |       0 |          0 | -          | f        | f          |
f         | f           | t          | f        | s          |
       1 |             0 |       1114 | 25         |             |           |           |             | text_timestamp
|       |          |       |               | f
|           | f        | f       |          |          |          0 | f          |             |               | 25         |
 varchar_timestamp |          11 |       10 |      12 |       1 |       0 |          0 | -          | f        | f          |
f         | f           | t          | f        | i          |
       1 |             0 |       1114 | 1043       |             |           |           |             |
varchar_timestamp |        |          |       |               | f
|           | f        | f       |          |          |          0 | f          |             |               | 1043       |
(2 rows)

-- Delete the test database. Replace postgres with the actual initial database name.
ora_db=# \c postgres
gaussdb=# DROP DATABASE ora_db;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

In the example, the forcible conversion of the varchar2 type is affected by the GUC parameter **modify_function_property_json**.

- Set the GUC parameter to an empty value, or set **property_value** from **s** to **i** for the object whose **property_name** is **provolatile** in the configuration item with OID **4186** in the GUC parameter, and restart the database. The result is returned normally.
- If the **modify_function_property_json** parameter is not modified, an error is reported, as expected. This happens because the function variability is stable, similar to the forcible conversion of the text type. The error message is as follows:

```
ora_db=# EXPLAIN VERBOSE UPDATE distcol_update_timestamp SET c1 = '2026-03-05
09:25:20.411077+08',c2 = -3 WHERE c1 = '2026-03-05 09:25:20.411077+08'::varchar2;
ERROR:  Distributed key cannot be updated in the current version.
```

- Example 5: System functions such as NVL and implicit type conversion are not supported.

```
-- Create a test database.
gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA';
gaussdb=# \c ora_db

-- Enable the function.
ora_db=# SET enable_remove_redundant_update_col=on;

-- Create a hash distributed table.
ora_db=# CREATE TABLE distcol_update_timestamp(c1 timestamp(6),c2 int,c3 text) DISTRIBUTE BY
HASH(c1);
ora_db=# INSERT INTO distcol_update_timestamp VALUES ('2026-03-05
09:25:19.411077+08',1,'gaussdb'),
    ('2026-03-05 09:25:19.411077+08',2,'gaussdb'),
    ('2026-03-05 09:25:20.411077+08',3,'gaussdb'),
    ('2026-03-05 09:25:21.411077+08',4,'gaussdb'),
    ('2026-03-05 09:25:22.411077+08',5,'gaussdb'),
    ('2026-03-06 09:25:22.411077+08',6,'gaussdb'),
    ('2026-03-07 09:25:22',7,'gaussdb');
ora_db=# CREATE INDEX ON distcol_update_timestamp(c1);

-- Run the UPDATE statement.
ora_db=# UPDATE distcol_update_timestamp SET c1 = '2026-03-05 09:25:20.411077+08',c2 = -3
WHERE c1 = nvl('2026-03-05 09:25:20.411077+08','2026-03-05 09:25:21.411077+08');
ERROR:  Distributed key cannot be updated in the current version.

-- Delete the test database. Replace postgres with the actual initial database name.
ora_db=# \c postgres
gaussdb=# DROP DATABASE ora_db;
```

- **Example 6**: An error is reported when a distribution key of the bigint or int type is pseudo-updated using the statement SET a = null,b = 1 WHERE a = null.

```
-- Create a test database.
gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA';
gaussdb=# \c ora_db

-- Create a hash distributed table.
ora_db=# CREATE TABLE distcol_update_bigint(c1 bigint,c2 int,c3 text) DISTRIBUTE BY hash(c1);
ora_db=# INSERT INTO distcol_update_bigint VALUES(null,9,'gaussdb');

-- Set parameters and run the UPDATE statement.
ora_db=# SET enable_remove_redundant_update_col=on;
ora_db=# SET a_format_version = '10c';
ora_db=# SET a_format_dev_version = 's6';
ora_db=# UPDATE distcol_update_bigint SET c1 = null,c3 = 'update4' WHERE c1 = null;
ERROR:  Distributed key cannot be updated in the current version.

-- Delete the test database. Replace postgres with the actual initial database name.
ora_db=# \c postgres
gaussdb=# DROP DATABASE ora_db;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1.13.22 V

1.13.22.1 VACUUM

Description

VACUUM recycles storage space occupied by deleted rows in tables or B-Tree indexes. In normal database operation, rows that have been deleted are not physically removed from their table; they remain present until a VACUUM is done. Therefore, it is necessary to execute VACUUM periodically, especially on frequently-updated tables.

Precautions

- If no table is specified, VACUUM processes the tables on which the user has the corresponding permission in the current database. If a table is specified, VACUUM processes only the specified table.
- To vacuum a table, you must be the table owner or a user with the VACUUM permission on the table. When separation of duties is disabled, system administrators have this authorization by default. However, database owners are allowed to perform VACUUM on all tables in their databases, except shared catalogs. (The restriction for shared catalogs means that a true database-wide VACUUM can only be executed by system administrators). VACUUM skips over any tables that the calling user does not have permission to vacuum.
- VACUUM cannot be executed inside a transaction block.
- It is recommended that production databases be vacuumed frequently (at least nightly) to ensure that dead rows are removed. It might be a good idea to always perform VACUUM ANALYZE on the affected table after adding or deleting a large number of rows. This will update the system catalogs with the results of all recent changes, and allow the query planner to make better choices in planning queries.
- FULL is recommended only in special scenarios. For example, you want to physically narrow the table to decrease the occupied disk space after deleting most rows of a table. VACUUM FULL usually shrinks a table more than VACUUM does. If the occupied physical space does not decrease after you run the statement, check whether there are other active transactions (that started before the data deleting transaction and has not finished before you run VACUUM FULL). If there are such transactions, wait for them to finish and run the statement again.
- VACUUM FULL returns the free space in the table to the tablespace by rebuilding the table. During the rebuild, extra storage space equivalent to the valid data size in the table is required. If the storage space is insufficient, the cluster may become read-only when you run VACUUM FULL. For a non-segment-page table, after VACUUM FULL is executed, the physical files occupied by the source table are deleted, and the physical file space occupied by the source table is returned to the OS. For a segment-page table, after VACUUM FULL is executed, the physical space occupied by the source table is returned to the segment-page data file instead of the OS.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- VACUUM causes a significant increase in I/O traffic, which may affect the performance of other active sessions. Therefore, it is sometimes advisable to use the cost-based VACUUM delay feature.
- When **VERBOSE** is specified, VACUUM prints progress messages to indicate which table is currently being processed. Various statistics about the tables are printed as well.
- When the option list is surrounded by parentheses, the options can be written in any order. Without parentheses, the options must be specified in the order given in the syntax.
- During VACUUM and VACUUM FULL operations, the cleanup of row-store table records is deferred based on the **vacuum_defer_cleanup_age** parameter, which means recently deleted tuples are not cleaned up immediately.
- VACUUM ANALYZE executes a VACUUM operation and then an ANALYZE operation for each selected table. This is a handy combination form for routine maintenance scripts.
- Plain VACUUM (without FULL) simply recycles space and makes it available for reuse. This form of statement can operate in parallel with normal reading and writing of the table, as an exclusive lock is not obtained. VACUUM FULL executes wider processing, including moving rows across blocks to compress tables so they occupy the minimum number of disk blocks. This form is much slower and requires an exclusive lock on each table while it is being processed.
- If the **xc_maintenance_mode** parameter is not enabled, VACUUM FULL skips all system catalogs.
- VACUUM HDFSDIRECTORY is a reserved syntax.
- If you run **VACUUM FULL** immediately after running **DELETE**, the space will not be recycled. After running **DELETE**, execute 1000 non-SELECT transactions, or wait for 1s and then execute one transaction. Then, run **VACUUM FULL** to recycle the space.
- During VACUUM FULL, an exclusive lock is added to the table. Therefore, you are advised not to run **VACUUM FULL** during peak hours. Otherwise, the lock waiting time is too long or a deadlock occurs.
- To ensure performance and statistics accuracy, do not run ANALYZE-related commands, such as VACUUM ANALYZE, AUTOANALYZE, and manual ANALYZE, at the same time or frequently.
- For Ustore, the behavior of manual VACUUM is the same as that in Astore. Locks are obtained to clear heap tables and indexes. In Ustore, AUTOVACUUM only clears GPIs of partitioned tables, updates FSMs of heap tables, and recycles index pages.
- When **VACUUM FULL** is executed, partitions are traversed for clearance and GPIs are rebuilt after partition clearance. Therefore, if there are a large number of partitions, you are advised to delete GPIs first and rebuild indexes after **VACUUM FULL** is executed. In this way, the execution time of **VACUUM FULL** is reduced. For online VACUUM FULL, if the object is a partitioned table, the GPI is rebuilt when the partition is cleared. The GPI is rebuilt after the partition is cleared only when the object is a single partition. Therefore, you do not need to delete the GPI before using VACUUM FULL.
- Online VACUUM FULL cannot be performed on database-level, index-level, segment-page, hash bucket, temporary, or unlogged tables. If an object

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

involves the preceding unsupported types, the ONLINE keyword does not take effect. By default, VACUUM FULL is executed offline, and a message is displayed indicating that the VACUUM FULL statement cannot be executed online.

- When performing online VACUUM FULL operations, reserve sufficient disk space, at least equal to the total size of the tables to be operated and the accessories such as indexes on the tables.
- If the online VACUUM FULL execution is canceled, the residual data clearance process starts first. If the execution is also canceled again, the clearance process ends and residual data remains.
- Resource contention exists between online VACUUM FULL operations and user services. Therefore, online VACUUM FULL operations are not suitable for heavy-traffic scenarios. In scenarios where resources are limited, fine-grained resource control is recommended. Parallel acceleration is not recommended.
- Online VACUUM FULL is not supported in M-compatible mode.
- Online VACUUM FULL execution does not support the FREEZE keyword. If both FREEZE and ONLINE keywords are specified, VACUUM FULL is executed offline by default, and a message is displayed indicating that the VACUUM FULL statement cannot be executed online.
- Online VACUUM FULL cannot be executed within a transaction or a stored procedure. If you use the JDBC driver to run multiple SQL statements, a transaction is implicitly started, and these SQL statements cannot be executed together.
- If online VACUUM FULL fails, significant table space bloat may occur. The bloat degree is proportional to the online VACUUM FULL execution duration and the number of concurrent write operations. Long transactions may block online VACUUM FULL. You are advised not to perform online VACUUM FULL when long transactions exist.
- DQL and DML operations will be blocked after online VACUUM FULL is executed.
- During online VACUUM FULL execution, do not execute DDL statements concurrently with other DDL statements. Otherwise, deadlocks or errors may occur. In this case, reconnect to the client and try again.
- A schema named **online$$ddl$$[hash value]** is generated during online VACUUM FULL execution.
  - By default, the owner of the schema is the initial user. You are advised not to perform the operations on the schema during online VACUUM FULL execution. Otherwise, the online VACUUM FULL execution may be terminated and an unknown error may occur. You are advised not to query tables in this schema. By default, this schema is not exported when gs_dump is used for data export.
  - You cannot create a schema whose prefix is **online$$**. Otherwise, an error notice is displayed. If schemas with the same name conflict with each other, online VACUUM FULL execution does not take effect. By default, VACUUM FULL is executed offline, and a message is displayed indicating that the VACUUM FULL statement cannot be executed online due to the schema name conflict.
- If a GSI exists in the object table, the ONLINE keyword does not take effect. By default, VACUUM FULL is executed offline, and a message is displayed indicating that VACUUM FULL cannot be executed online.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Avoid performing online VACUUM FULL operations and scaling concurrently. Otherwise, online VACUUM FULL operations or the scaling may fail.
- Except the VACUUM ANALYZE statement, other VACUUM operations do not report the **relpages**/**reltuples** information of the main table of a partitioned table in the statistics if the original behavior remains unchanged.
- In the **reloptions** column of the PG_PARTITION system catalog, **wait_clean_gpi** specifies whether to wait for the global partitioned index to be cleared. If a tuple whose **wait_clean_gpi** is **y** exists, the tuple cannot be directly cleared and can be cleared only after the global partitioned index is cleared.

Syntax

- Recycle space and update statistics. The keyword order does not matter.

```
VACUUM [ ( { FULL | FREEZE | VERBOSE | { ANALYZE | ANALYSE } } [, ... ] ) ]
    [ table_name [ (column_name [, ...] ) ] [ PARTITION ( partition_name ) | SUBPARTITION
( subpartition_name ) ] ]
    [ OFFLINE | [ ONLINE [ WITH ( { online_parameter = value } [, ... ] ) ] ] ];
```

- Recycle space, without updating statistics information.

```
VACUUM [ FULL ] [ FREEZE ] [ VERBOSE ] [ table_name [ PARTITION ( partition_name ) |
SUBPARTITION ( subpartition_name ) ] ]
    [ OFFLINE | [ ONLINE [ WITH ( { online_parameter = value } [, ... ] ) ] ] ];
```

- Recycle space and update statistics. The keywords need to be in order.

```
VACUUM [ FULL ] [ FREEZE ] [ VERBOSE ] { ANALYZE | ANALYSE } [ VERBOSE ]
    [ table_name [ ( column_name [, ...] ) ] ] [ PARTITION ( partition_name ) ]
    [ OFFLINE | [ ONLINE [ WITH ( { online_parameter = value } [, ... ] ) ] ] ];
```

- Recycle the space of a specified bucket and update the **bucketxid** column in the pg_hashbucket system catalog. Statistics are not updated. It can be called by administrators only during scale-out.

```
VACUUM FREEZE BUCKETS (bucketid [, ...]);
```

Parameters

- **FULL**

  Selects "FULL" vacuum, which can recycle more space, but takes much longer and exclusively locks the table.

  NOTE

  Using FULL will cause statistics information missing. To collect statistics information, add the keyword ANALYZE to the VACUUM FULL statement.

- **FREEZE**

  Specifying FREEZE is equivalent to running **VACUUM** with the **vacuum_freeze_min_age** parameter set to **0**.

- **VERBOSE**

  Prints a detailed vacuum activity report for each table.

- **ANALYZE | ANALYSE**

  Updates statistics used by the planner to determine the most efficient way to execute a query.

- **table_name**

  Specifies the name (optionally schema-qualified) of a specific table to vacuum.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: name of a specific table to vacuum. The default value indicates all tables in the current database.

- **column_name**

  Specifies the name of the column to be analyzed. This parameter must be used together with ANALYZE.

  Value range: name of a specific column to analyze. The default value indicates all columns.

  NOTE

  The mechanism of the VACUUM ANALYZE statement is to execute VACUUM and ANALYZE in sequence. Therefore, if **column_name** is incorrect, VACUUM may be successfully executed but ANALYZE may fail to be executed. For a partitioned table, ANALYZE may fail to be executed after VACUUM is successfully executed on a partition.

- **partition_name**

  Specifies the partition name of the table to be cleared. The default value indicates all partitions.

- **ONLINE | OFFLINE**

  Executes VACUUM online or offline. This parameter is used to execute VACUUM FULL online and therefore the read and write services are not blocked for a long time during VACUUM FULL.

  The keywords ONLINE and OFFLINE are used.

  – ONLINE: executes online. If the VACUUM FULL statement contains scenarios (such as the FREEZE keyword) that are not supported by the current feature, the statement is executed offline and a NOTICE is displayed, indicating that the ONLINE keyword is ignored.

  – OFFLINE: executes offline.

  Value range: **OFFLINE** or **ONLINE**, which indicates offline execution and online execution respectively. The value is **OFFLINE** by default.

  NOTE

  If VACUUM FULL fails to be executed online due to user instruction termination commands, lock timeout, or database faults, the system automatically clears residual objects. Automatic cleanup may fail due to reasons such as lock timeout. In this case, manual cleanup is required. In the case of severe faults, such as FATAL, PANIC, and database faults, temporary schema objects may remain, affecting the execution of subsequent online VACUUM FULL and online DDL operations. In this case, you can use the **online_ddl_cleanup()** function to manually clear the objects or choose to automatically clear them when the next online VACUUM FULL or online DDL operation is performed. If no residual data exists, you are advised not to use the online_ddl_cleanup() function. Otherwise, online VACUUM FULL may be terminated and an unknown error may occur.

- **online_parameter**

  Specifies the online VACUUM FULL option when the **ONLINE** keyword is specified. This parameter does not take effect if the **ONLINE** keyword is not specified or conditions for online VACUUM FULL execution are not met.

  Value range:

  – **parallel_threads**: controls multi-thread parallel operations to achieve higher performance when resources are sufficient. Value range: an integer ranging from 1 to 32. The default value is **1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- When a user enables resource control, for example, I/O resource control (for example, **io_limits** > **0** or **io_priority** = **high**/**medium**/**low**), **parallel_threads** ignores resource control under the user or session. In addition, a message is displayed, indicating that the current resource control is invalid and multi-thread acceleration is enabled. The online VACUUM FULL multi-thread execution complies with the resource control of the initial user.
- This parameter can take effect together with the index parallel parameter **parallel_workers**. **parallel_workers** controls the degree of parallelism for index building during VACUUM FULL, and **parallel_threads** controls the degree of parallelism for catchups during VACUUM FULL.

– **max_catchup_times**: controls the maximum number of catchups. This parameter is used to prevent the execution of VACUUM FULL from taking a long time. The value is an integer from 1 to *INT_MAX*, with **50** being the default.

- Online VACUUM FULL execution records the IUD incremental data generated during data reconstruction to a temporary table and reconstructs the incremental data in multiple rounds of catchups. To prevent VACUUM FULL catchup failures due to a large amount of concurrent IUD data, you can set this parameter to limit the number of catchups.

– **last_catchup_threshold**: specifies the upper limit of the time for triggering the last catchup. After a catchup round is complete, if the duration of this round is less than the threshold, the last round of catchup and switchover is triggered. Value range: [1000,1200000] (unit: ms). The default value is **5000**.

– **online_ddl_lockwait_timeout**: specifies the lock wait timeout interval for online DDL operations. If the thread waiting for the lock fails to obtain the lock within the specified time, a lock timeout error is reported. When the **CANCEL** mode is used to obtain the lock, the actual timeout interval is twice the value of this parameter. Value range: [1,1200000] (unit: ms). The default value is **10000**.

– **lockwait_interval**: indicates the lock wait increment, that is, the retry interval after the lock wait times out. If the lock wait times out in the current round, the system waits for the period specified by this parameter and then attempts to obtain the lock in the next round. Value range: [0,300] (unit: seconds). The default value is **5**.

– **trylock_threshold**: indicates the maximum number of lock wait attempts. It must be used together with **enable_cancel**. If the lock cannot be obtained due to lock wait timeout after the maximum number of attempts is reached, the **CANCEL** mode will be used for the next attempt. If **enable_cancel** is not enabled, the lock fails to be obtained and an error is reported after the maximum number of attempts is reached. Value range: [1,*INT_MAX*] or **-1** (indicating that there is no upper limit). The default value is **-1**.

– **enable_cancel**: specifies whether services can be canceled. That is, when the number of lock attempts reaches the upper limit or the level-8 lock of the user table is obtained during the table switching transaction, user

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

services are quickly terminated to ensure that the online DDL operation can successfully obtain the lock. The value can be **false** or **true**. The default value is **false**.

Examples

- VACUUM

```
-- Create the table tbl_test and insert data into the tables.
gaussdb=# CREATE TABLE tbl_test(c1 int);
gaussdb=# INSERT INTO tbl_test VALUES (1);

-- View the data and the CTID of the data.
gaussdb=# SELECT ctid,* FROM tbl_test;
 ctid  | c1
-------+----
 (0,1) |  1
(1 row)

-- Delete the data record.
gaussdb=# DELETE FROM tbl_test;

-- Insert a data record again. It is found that a new ctid is used.
gaussdb=# INSERT INTO tbl_test VALUES (2);
gaussdb=# SELECT ctid,* FROM tbl_test;
 ctid  | c1
-------+----
 (0,2) |  2
(1 row)

-- After the VACUUM statement is executed, the old space is reused when data is inserted.
gaussdb=# VACUUM ANALYZE tbl_test;
gaussdb=# INSERT INTO tbl_test VALUES (3);
gaussdb=# SELECT ctid,* FROM tbl_test;
 ctid  | c1
-------+----
 (0,1) |  3
 (0,2) |  2
(2 rows)

-- Drop the table.
gaussdb=# DROP TABLE tbl_test;
```

- VACUUM FULL

```
-- Create a table.
gaussdb=# CREATE TABLE tbl_test2(c1 int);

-- Insert 100,000 data records and check the table size.
gaussdb=# INSERT INTO tbl_test2 VALUES (generate_series(1,100000));
gaussdb=# SELECT 'tbl_test2' AS tablename, pg_size_pretty(pg_relation_size('tbl_test2')) AS size;
 tablename |  size
-----------+---------
 tbl_test2 | 3048 kB
(1 row)

-- Delete data and check the table size.
gaussdb=# DELETE FROM tbl_test2;
gaussdb=# SELECT 'tbl_test2' AS tablename, pg_size_pretty(pg_relation_size('tbl_test2')) AS size;
 tablename |  size
-----------+---------
 tbl_test2 | 3048 kB
(1 row)

-- Run the SELECT statement to obtain the ID of the current transaction.
gaussdb=# SELECT TXID_CURRENT();
txid_current
-------------
       55196
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Use VACUUM FULL to recycle space, update statistics, and check the table size.
gaussdb=# VACUUM FULL ANALYZE tbl_test2;
gaussdb=# SELECT 'tbl_test2' AS tablename, pg_size_pretty(pg_relation_size('tbl_test2')) AS size;
 tablename |  size
-----------+---------
 tbl_test2 | 0 bytes
(1 row)

-- Use online VACUUM FULL to recycle space and update statistics.
gaussdb=# VACUUM FULL ANALYZE tbl_test2 ONLINE;
VACUUM

-- Use online VACUUM FULL to recycle space, update statistics, and set online_parameter.
gaussdb=# VACUUM FULL ANALYZE tbl_test2 ONLINE with(parallel_threads=10,
max_catchup_times=20, last_catchup_threshold=3000, online_ddl_lockwait_timeout=10,
lockwait_interval=10, trylock_threshold=1, enable_cancel=true);
VACUUM

-- Drop.
gaussdb=# DROP TABLE tbl_test2;
```

Suggestions

- vacuum
  - VACUUM cannot be executed inside a transaction block.
  - It is recommended that production databases be vacuumed frequently (at least nightly) to ensure that dead rows are removed. It is strongly recommended that you perform VACUUM ANALYZE on corresponding tables after adding or deleting a large number of records.
  - FULL is recommended only in special scenarios. For example, you wish to physically narrow the table to decrease the occupied disk space after deleting most rows of a table.

1.13.22.2 VALUES

Description

Computes a row or a set of rows based on given values. It is most commonly used to generate a constant table within a large statement.

Precautions

- VALUES lists with large numbers of rows should be avoided, as you might encounter out-of-memory failures or poor performance. VALUES appearing within INSERT is a special case because the desired column types are known from the INSERT's target table, and need not be inferred by scanning the VALUES list. In this case, a large number of result rows can be processed.
- If more than one row is specified, all the rows must have the same number of elements.
- When the IN expression is used for multi-column query, IN LIST and IN VALUES LIST have similar functions but different internal processing mechanisms. To ensure query performance and data accuracy, pay special attention to the following key differences during replacement:
  - Logical structure difference: IN LIST parses multiple columns of conditions into a combination of multiple OR conditions. For example, (a, b) IN ((1, 2), (2, 3)) is converted into (a = 1 AND b = 2) OR (a = 2 AND b = 3).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

When the list after IN is too long, the generated OR condition chain is extremely long, the query optimizer needs to traverse the entire condition chain, which may cause the query performance to deteriorate significantly.

- Query optimization mechanism: IN VALUES LIST considers the VALUES clause as a derived table, parses the IN expression to an ANY subquery, and converts the IN expression to a JOIN operation in the query rewriting phase. This mechanism effectively avoids the performance overhead of processing long OR condition chains.
- Difference in type processing: IN LIST performs type check and implicit conversion on each group of data separately, while IN VALUES LIST deduces the common data type for each column in the VALUES clause, and then performs type check and conversion. This unified type processing method is stricter, requiring that the data types of the same column in the VALUES clause be consistent to ensure the accuracy of the query result.

Syntax

```
VALUES {( expression [, ...] )} [, ...]
    [ ORDER BY { sort_expression [ ASC | DESC | USING operator ] } [, ...] ]
    [ LIMIT { count | ALL } ]
    [ OFFSET start [ ROW | ROWS ] ]
    [ FETCH { FIRST | NEXT } [ count ] { ROW | ROWS } ONLY ];
```

Parameters

- **expression**

  Specifies a constant or expression to compute and insert at the indicated place in the resulting table or set of rows.

  In a **VALUES** list appearing at the top level of an **INSERT**, an expression can be replaced by **DEFAULT** to indicate that the destination column's default value should be inserted. **DEFAULT** cannot be used when **VALUES** appears in other contexts.

- **sort_expression**

  Specifies an expression or integer constant indicating how to sort the result rows.

- **ASC**

  Specifies an ascending sort order.

- **DESC**

  Specifies a descending sort order.

- **operator**

  Specifies a sorting operator.

- **count**

  Specifies the maximum number of rows to return.

- **OFFSET start [ROW | ROWS]**

  Specifies the maximum number of returned rows, whereas **start** specifies the number of rows to skip before starting to return rows.

- **FETCH { FIRST | NEXT } [count] { ROW | ROWS } ONLY**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The **FETCH** clause restricts the total number of rows starting from the first row of the return query result, and the default value of **count** is **1**.

Examples

```
-- Create a table.
gaussdb=# CREATE TABLE test_t2(col1 INT,col2 VARCHAR);
gaussdb=# CREATE TABLE test_t3(col1 INT,col2 VARCHAR);

-- Insert multiple data records.
gaussdb=# INSERT INTO test_t2 (col1, col2) VALUES (10,'AA'),(20,'BB'),(30,'CC');

-- Perform a pagination query and return to the first page. Display one row of data on each page.
gaussdb=# SELECT * FROM test_t2 ORDER BY col1 ASC LIMIT 1 OFFSET 0;
col1 | col2
------+------
   10 | AA
(1 row)

-- Insert data in test_t2 into test_t3.
gaussdb=# INSERT INTO test_t3 SELECT * FROM test_t2;

-- Perform a query using FETCH and return to the first page. Display one row of data on each page.
gaussdb=# SELECT * FROM test_t3 ORDER BY col1 ASC OFFSET 0 FETCH FIRST 1 ROWS ONLY;
 col1 | col2
------+------
   10 | AA
(1 row)

-- Delete tables test_t2 and test_t3.
gaussdb=# DROP TABLE test_t2;
gaussdb=# DROP TABLE test_t3;
```

1.14 Appendix

1.14.1 Extended Functions

The following table lists the extended functions supported by GaussDB and they are for reference only.

Type	Name	Description
Trigger function	pg_get_triggerdef(trigger_oid)	Gets **CREATE** [**CONSTRAINT**] **TRIGGER** command for triggers.
	pg_get_triggerdef(trigger_oid, pretty_bool)	Gets **CREATE** [**CONSTRAINT**] **TRIGGER** command for triggers.

1.14.2 Dollar-Quoted String Constants

If a string sequence contains a single quotation mark ('), double the single quotation mark (') to two single quotation marks (''). Otherwise, the SQL statement may fail to be executed.

If a string contains many single quotation marks (') or backslashes (\), the string may be difficult to understand and error-prone because all the single quotation marks (') are doubled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

To make the query more readable in such situations, dollar quoting is provided to write string constants. A dollar-quoted string constant consists of a dollar sign ($), a "tag" of zero or more characters, another dollar sign, an arbitrary sequence of characters that make up the string content, a dollar sign, another tag the same as the previous one, and a dollar sign.

```
gaussdb=# SELECT $$it's an example$$;
   ?column?
-----------------
 it's an example
(1 row)
```

1.14.3 Row Expression Function Whitelist

Table 1-407 Expression function whitelist for data objects to add or modify ILM policies

func_oid_value	func_name
38	int2in
39	int2out
42	int4in
43	int4out
46	textin
47	textout
56	boollt
57	boolgt
60	booleq
61	chareq
63	int2eq
64	int2lt
65	int4eq
66	int4lt
67	texteq
70	charne
72	charle
73	chargt
74	charge
77	int4
78	char

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
84	boolne
111	numeric_fac
141	int4mul
144	int4ne
145	int2ne
146	int2gt
147	int4gt
148	int2le
149	int4le
150	int4ge
151	int2ge
152	int2mul
153	int2div
154	int4div
155	int2mod
156	int4mod
157	textne
158	int24eq
159	int42eq
160	int24lt
161	int42lt
162	int24gt
163	int42gt
164	int24ne
165	int42ne
166	int24le
167	int42le
168	int24ge
169	int42ge
170	int24mul

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
171	int42mul
172	int24div
173	int42div
176	int2pl
177	int4pl
178	int24pl
179	int42pl
180	int2mi
181	int4mi
182	int24mi
183	int42mi
200	float4in
201	float4out
202	float4mul
203	float4div
204	float4pl
205	float4mi
206	float4um
207	float4abs
209	float4larger
211	float4smaller
212	int4um
213	int2um
214	float8in
215	float8out
216	float8mul
217	float8div
218	float8pl
219	float8mi
220	float8um

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
221	float8abs
223	float8larger
224	float8smaller
228	dround
229	dtrunc
235	float8
236	float4
237	int2
238	int2
244	timepl
245	timemi
248	intinterval
249	tintervalrel
251	abstimeeq
252	abstimene
253	abstimelt
254	abstimegt
255	abstimele
256	abstimege
257	reltimeeq
258	reltimene
259	reltimelt
260	reltimegt
261	reltimele
262	reltimege
263	tintervalsame
264	tintervalct
265	tintervalov
266	tintervalleneq
267	tintervallenne

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
268	tintervallenlt
269	tintervallengt
270	tintervallenle
271	tintervallenge
273	tintervalend
275	isfinite
279	float48mul
280	float48div
281	float48pl
282	float48mi
283	float84mul
284	float84div
285	float84pl
286	float84mi
287	float4eq
288	float4ne
289	float4lt
290	float4le
291	float4gt
292	float4ge
293	float8eq
294	float8ne
295	float8lt
296	float8le
297	float8gt
298	float8ge
299	float48eq
300	float48ne
301	float48lt
302	float48le

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
303	float48gt
304	float48ge
305	float84eq
306	float84ne
307	float84lt
308	float84le
309	float84gt
310	float84ge
311	float8
312	float4
313	int4
314	int2
316	float8
317	int4
318	float4
319	int4
350	btint2cmp
351	btint4cmp
352	year_int1_mysql
354	btfloat4cmp
355	btfloat8cmp
357	btabstimecmp
358	btcharcmp
360	bttextcmp
377	cash_cmp
380	btreltimecmp
381	bttintervalcmp
385	regexp_count
386	regexp_count
387	regexp_count

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
400	hashtext
432	hash_numeric
449	hashint2
450	hashint4
451	hashfloat4
452	hashfloat8
454	hashchar
458	text_larger
459	text_smaller
460	int8in
461	int8out
462	int8um
463	int8pl
464	int8mi
465	int8mul
466	int8div
467	int8eq
468	int8ne
469	int8lt
470	int8gt
471	int8le
472	int8ge
474	int84eq
475	int84ne
476	int84lt
477	int84gt
478	int84le
479	int84ge
480	int4
481	int8

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
482	float8
483	int8
630	regexp_instr
631	regexp_instr
632	regexp_instr
633	regexp_instr
634	regexp_instr
652	float4
653	ftoi8
654	hashint1_numeric
665	hashint2_numeric
667	hashint16
668	bpchar
669	varchar
676	mktinterval
682	hashint4_numeric
714	int2
720	octet_length
721	get_byte
722	set_byte
723	get_bit
724	set_bit
740	text_lt
741	text_le
742	text_gt
743	text_ge
754	int8
755	hashint8_numeric
766	int4inc
768	int4larger

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
769	int4smaller
770	int2larger
771	int2smaller
784	tintervaleq
785	tintervalne
786	tintervallt
787	tintervalgt
788	tintervalle
789	tintervalge
792	btint12cmp
793	btint14cmp
794	btint18cmp
795	btint116cmp
796	btint1numericcmp
797	btint21cmp
798	btint216cmp
799	btint2numericcmp
800	btint41cmp
801	btint416cmp
802	btint4numericcmp
803	btint81cmp
804	btint816cmp
805	btint8numericcmp
837	int82pl
838	int82mi
839	int82mul
840	int82div
841	int28pl
842	btint8cmp
846	cash_mul_flt4

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
847	cash_div_flt4
848	flt4_mul_cash
849	position
852	int48eq
853	int48ne
854	int48lt
855	int48gt
856	int48le
857	int48ge
860	bpchar
862	int4_mul_cash
863	int2_mul_cash
864	cash_mul_int4
865	cash_div_int4
866	cash_mul_int2
867	cash_div_int2
868	strpos
870	lower
871	upper
877	substr
883	substr
888	cash_eq
889	cash_ne
890	cash_lt
891	cash_le
892	cash_gt
893	cash_ge
894	cash_pl
895	cash_mi
896	cash_mul_flt8

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
897	cash_div_flt8
898	cashlarger
899	cashsmaller
919	flt8_mul_cash
935	cash_words
936	substring
937	substring
940	mod
941	mod
942	int28mi
943	int28mul
944	char
945	int8mod
947	mod
948	int28div
949	hashint8
1026	timezone
1044	bpcharin
1045	bpcharout
1046	varcharin
1047	varcharout
1048	bpchareq
1049	bpcharlt
1050	bpcharle
1051	bpchargt
1052	bpcharge
1053	bpcharne
1063	bpchar_larger
1064	bpchar_smaller
1078	bpcharcmp

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
1080	hashbpchar
1102	time_lt
1103	time_le
1104	time_gt
1105	time_ge
1106	time_ne
1107	time_cmp
1116	regexp_replace
1117	regexp_replace
1144	time_out
1145	time_eq
1152	timestamptz_eq
1153	timestamptz_ne
1154	timestamptz_lt
1155	timestamptz_le
1156	timestamptz_ge
1157	timestamptz_gt
1158	to_timestamp
1159	timezone
1162	interval_eq
1163	interval_ne
1164	interval_lt
1165	interval_le
1166	interval_ge
1167	interval_gt
1168	interval_um
1169	interval_pl
1170	interval_mi
1172	date_part
1173	timestamptz

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
1177	interval
1180	abstime
1188	timestamptz_mi
1194	reltime
1195	timestamptz_smaller
1196	timestamptz_larger
1197	interval_smaller
1198	interval_larger
1199	age
1200	interval
1218	date_trunc
1219	int8inc
1230	int8abs
1236	int8larger
1237	int8smaller
1238	texticregexeq
1239	texticregexne
1246	charlt
1251	int4abs
1253	int2abs
1254	textregexeq
1256	textregexne
1271	overlaps
1273	date_part
1274	int84pl
1275	int84mi
1276	int84mul
1277	int84div
1278	int48pl
1279	int48mi

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
1280	int48mul
1281	int48div
1282	quote_ident
1283	quote_literal
1289	quote_nullable
1299	now
1304	overlaps
1308	overlaps
1309	overlaps
1310	overlaps
1311	overlaps
1314	timestamptz_cmp
1315	interval_cmp
1316	time
1326	interval_div
1337	interval
1342	round
1343	trunc
1352	timetz_eq
1353	timetz_ne
1354	timetz_lt
1355	timetz_le
1356	timetz_ge
1357	timetz_gt
1358	timetz_cmp
1359	timestamptz
1360	space
1370	interval
1373	isfinite
1374	octet_length

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
1375	octet_length
1377	time_larger
1378	time_smaller
1379	timetz_larger
1380	timetz_smaller
1384	date_part
1385	date_part
1389	isfinite
1390	isfinite
1394	abs
1395	abs
1396	abs
1397	abs
1398	abs
1419	time
1481	tinterval
1564	bit_in
1565	bit_out
1581	biteq
1582	bitne
1592	bitge
1593	bitgt
1594	bitle
1595	bitlt
1596	bitcmp
1608	degrees
1618	interval_mul
1620	ascii
1621	chr
1622	repeat

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
1623	similar_escape
1624	mul_d_interval
1633	texticlike
1634	texticnlike
1656	bpcharicregexeq
1657	bpcharicregexne
1658	bpcharregexeq
1659	bpcharregexne
1660	bpchariclike
1661	bpcharicnlike
1666	varbiteq
1667	varbitne
1668	varbitge
1669	varbitgt
1670	varbitle
1671	varbitlt
1672	varbitcmp
1673	bitand
1674	bitor
1675	bitxor
1676	bitnot
1677	bitshiftleft
1678	bitshiftright
1679	bitcat
1682	octet_length
1683	bit
1684	int4
1685	bit
1687	varbit
1688	time_hash

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
1690	time_mi_time
1691	boolle
1692	boolge
1693	btboolcmp
1696	timetz_hash
1697	interval_hash
1698	position
1701	numeric_in
1702	numeric_out
1703	numeric
1704	numeric_abs
1705	abs
1706	sign
1707	round
1709	trunc
1710	trunc
1711	ceil
1712	floor
1718	numeric_eq
1719	numeric_ne
1720	numeric_gt
1721	numeric_ge
1722	numeric_lt
1723	numeric_le
1724	numeric_add
1725	numeric_sub
1726	numeric_mul
1727	numeric_div
1728	mod
1729	numeric_mod

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
1740	numeric
1742	numeric
1743	numeric
1744	int4
1745	float4
1746	float8
1747	time_pl_interval
1748	time_mi_interval
1749	timetz_pl_interval
1750	timetz_mi_interval
1752	trunc
1753	trunc
1764	numeric_inc
1766	numeric_smaller
1767	numeric_larger
1769	numeric_cmp
1771	numeric_uminus
1779	numeric_int8
1781	numeric
1782	numeric
1783	int2
1810	bit_length
1811	bit_length
1812	bit_length
1840	int2_sum
1841	int4_sum
1842	int8_sum
1845	to_ascii
1846	to_ascii
1848	interval_pl_time

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
1850	int28eq
1851	int28ne
1852	int28lt
1853	int28gt
1854	int28le
1855	int28ge
1856	int82eq
1857	int82ne
1858	int82lt
1859	int82gt
1860	int82le
1861	int82ge
1874	btint161cmp
1875	btint162cmp
1876	btint164cmp
1877	btint168cmp
1878	btnumericint1cmp
1879	btnumericint2cmp
1880	btnumericint4cmp
1881	btnumericint8cmp
1882	btint16cmp
1892	int2and
1893	int2or
1894	int2xor
1895	int2not
1896	int2shl
1897	int2shr
1898	int4and
1899	int4or
1900	int4xor

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
1901	int4not
1902	int4shl
1903	int4shr
1904	int8and
1905	int8or
1906	int8xor
1907	int8not
1908	int8shl
1909	int8shr
1910	int8up
1911	int2up
1912	int4up
1913	float4up
1914	float8up
1915	numeric_uplus
1946	encode
1961	timestamp
1967	timestamptz
1968	time
1969	timetz
1973	div
1980	numeric_div_trunc
2012	substring
2013	substring
2014	position
2020	date_trunc
2021	date_part
2024	timestamp
2025	timestamp
2029	date

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
2031	timestamp_mi
2032	timestamp_pl_interval
2033	timestamp_mi_interval
2035	timestamp_smaller
2036	timestamp_larger
2038	timezone
2039	timestamp_hash
2041	overlaps
2042	overlaps
2043	overlaps
2044	overlaps
2045	timestamp_cmp
2046	time
2048	isfinite
2052	timestamp_eq
2053	timestamp_ne
2054	timestamp_lt
2055	timestamp_le
2056	timestamp_ge
2057	timestamp_gt
2058	age
2069	timezone
2070	timezone
2073	substring
2075	bit
2076	int8
2089	to_hex
2090	to_hex
2160	text_pattern_lt
2161	text_pattern_le

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
2163	text_pattern_ge
2164	text_pattern_gt
2166	bttext_pattern_cmp
2167	ceiling
2174	bpchar_pattern_lt
2175	bpchar_pattern_le
2177	bpchar_pattern_ge
2178	bpchar_pattern_gt
2180	btbpchar_pattern_cmp
2188	btint48cmp
2189	btint84cmp
2190	btint24cmp
2191	btint42cmp
2192	btint28cmp
2193	btint82cmp
2194	btfloat48cmp
2195	btfloat84cmp
2308	ceil
2309	floor
2310	sign
2320	ceiling
2338	date_timestamp_lt_mysql
2339	date_le_timestamp
2340	date_eq_timestamp
2341	date_gt_timestamp
2342	date_ge_timestamp
2343	date_ne_timestamp
2364	timestamp_date_lt_mysql
2365	timestamp_le_date
2366	timestamp_eq_date

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
2367	timestamp_gt_date
2368	timestamp_ge_date
2369	timestamp_ne_date
2424	float4recv
2425	float4send
2426	float8recv
2456	bit_recv
2457	bit_send
2460	numeric_recv
2461	numeric_send
2470	time_recv
2515	booland_statefunc
2516	boolor_statefunc
2522	timestamp_eq_timestamptz
2529	timestamptz_eq_timestamp
2547	interval_pl_timetz
2548	interval_pl_timestamp
2557	bool
2558	int4
2765	regexp_split_to_table
2766	regexp_split_to_table
2805	int8inc_float8_float8
2906	timestamptypmodout
2908	timestamptztypmodout
2910	timetypmodout
2912	timetztypmodout
2918	numerictypmodout
2919	bittypmodin
2920	bittypmodout
2971	text

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
2996	int8_sum_to_int8
3032	get_bit
3033	set_bit
3062	reverse
3167	instr
3168	instr
3169	instr
3170	multiply
3171	multiply
3175	lengthb
3176	lengthb
3177	int8_bool
3178	bool_int8
3180	int2_bool
3181	bool_int2
3182	substring_inner
3183	substring_inner
3192	int4_bpchar
3226	timestamp_diff
3227	timestamp_diff
3343	int8_mul_cash
3344	cash_mul_int8
3345	cash_div_int8
3360	addtime
3361	subtime
3616	tsvector_lt
3617	tsvector_le
3618	tsvector_eq
3619	tsvector_ne
3620	tsvector_ge

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
3621	tsvector_gt
3625	tsvector_concat
3634	ts_match_vq
3635	ts_match_qv
3662	tsquery_lt
3663	tsquery_le
3664	tsquery_eq
3665	tsquery_ne
3666	tsquery_ge
3667	tsquery_gt
3669	tsquery_and
3670	tsquery_or
3671	tsquery_not
3691	tsq_mcontains
3692	tsq_mcontained
3822	cash_div_cash
3922	int4range_subdiff
3923	int8range_subdiff
3924	numrange_subdiff
3925	daterange_subdiff
3929	tsrange_subdiff
3930	tstzrange_subdiff
3951	pg_systimestamp
4035	rawin
4036	rawout
4065	int1_varchar
4067	int1_bpchar
4068	int2_bpchar
4069	int8_bpchar
4070	float4_bpchar

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
4071	float8_bpchar
4072	numeric_bpchar
4073	text_timestamp
4098	group_concat_finalfn_mysql
4099	group_concat_transfn_mysql
4159	date_text
4160	date_bpchar
4161	date_varchar
4162	varchar_date
4163	bpchar_date
4164	text_date
4165	int1_text
4166	int2_text
4167	int4_text
4168	int8_text
4169	float4_text
4170	float8_text
4171	numeric_text
4172	bpchar_numeric
4173	varchar_numeric
4178	timestamp_text
4179	timestamp_varchar
4180	int2_varchar
4181	int4_varchar
4182	int8_varchar
4184	float4_varchar
4185	float8_varchar
4186	varchar_timestamp
4187	bpchar_timestamp
4194	text_numeric

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
4471	strcmp
4490	texticregexeq_mysql
4491	texticregexne_mysql
4609	btequalimage
4835	datea_cmp
4915	timestamptz_datea
4916	timestamp_datea
4918	datea_timestamp
5046	int2_int8_equal_mysql
5047	int8_int4_equal_mysql
5048	int4_int8_equal_mysql
5049	int2_int2_equal_mysql
5050	int4_int4_equal_mysql
5051	int2_int4_equal_mysql
5052	int4_int2_equal_mysql
5053	float4equal_mysql
5054	float48equal_mysql
5055	float84equal_mysql
5056	bpchar_equal_mysql
5080	float4_bit_mysql
5081	float8_bit_mysql
5082	numeric_bit_mysql
5083	cast
5084	cast
5107	double_bool_m
5108	datetime_bool_m
5109	timestamp_bool_m
5110	date_bool_m
5111	year_bool_m
5112	time_bool_m

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
5113	mediumint_bool_m
5114	uint1_bool_m
5115	uint2_bool_m
5116	uint3_bool_m
5117	uint4_bool_m
5118	uint8_bool_m
5119	text_bool_m
5120	blob_bool_m
5121	binary_bool_m
5519	int1cmp
5520	hashint1
5521	int1_numeric
5522	numeric_int1
5523	i1toi2
5524	i2toi1
5525	i1toi4
5526	i4toi1
5527	i1toi8
5528	i8toi1
5529	i1tof4
5530	f4toi1
5531	i1tof8
5532	f8toi1
5541	int1in
5542	int1out
5543	int1recv
5544	int1send
5561	real_bool_m
5577	bit_in_for_copy
5580	smalldatetime_eq

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
5581	smalldatetime_ne
5582	smalldatetime_lt
5583	smalldatetime_le
5584	smalldatetime_ge
5585	smalldatetime_gt
5586	smalldatetime_cmp
5587	smalldatetime_hash
5588	bit_bool_m
5809	b_db_last_day
5810	b_db_last_day
5811	b_db_last_day
5816	b_db_last_day
5844	bool_bit_mysql
5940	datea_eq
5945	datea_ge
5950	datea_gt
5955	datea_le
5960	datea_lt
5965	datea_neq
5970	datea_pl_interval
5971	interval_pl_datea
5972	datea_mi_interval
5973	datea_mi
5974	datea_mi_timestamp
5975	timestamp_mi_datea
6041	year_in_mysql
6042	year_out_mysql
6043	year_recv_mysql
6044	year_send_mysql
6045	yeartypmodin_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
6046	yeartypmodout_mysql
6069	date_year_mysql
6070	time_year_mysql
6071	timestamp_year_mysql
6118	mediumtext_equal_mysql
6135	now_b
6230	float4_row_text_mysql
6231	float4_row_blob_mysql
6282	datea_larger
6283	datea_smaller
6294	datea_hash
6296	trunc
6308	date_part
6407	int16
6408	int2
6409	int16
6410	int4
6411	int16
6412	int8
6413	int16
6414	float8
6415	int16
6416	float4
6419	int16
6420	int16_bool
6421	int16
6422	numeric
6423	int16eq
6424	int16ne
6425	int16lt

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
6426	int16le
6427	int16gt
6428	int16ge
6429	int16pl
6430	int16mi
6431	int16mul
6432	int16div
6433	numeric
6434	numeric_bool
6435	uint_int3_gt_mysql
6436	uint_int3_le_mysql
6437	int3_uint_eq_mysql
6438	int21gt
6439	int21le
6440	int21ge
6441	int216eq
6442	int216ne
6443	int216lt
6444	int216gt
6445	int216le
6446	int216ge
6447	int2numericeq
6448	int2numericne
6449	int2numericlt
6450	int2numericgt
6451	int2numericle
6452	int2numericge
6453	int41eq
6454	int41ne
6455	int41lt

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
6456	int41gt
6457	int41le
6458	int41ge
6459	int416eq
6460	int416ne
6461	int416lt
6462	int416gt
6463	int416le
6464	int416ge
6465	int4numericeq
6466	int4numericne
6467	int4numericlt
6468	int4numericgt
6469	int4numericle
6470	int4numericge
6471	int81eq
6472	int81ne
6473	int81lt
6474	int81gt
6475	int81le
6476	int81ge
6477	int816eq
6478	int816ne
6479	int816lt
6480	int816gt
6481	int816le
6482	int816ge
6483	int8numericeq
6484	int8numericne
6485	int8numericlt

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
6486	int8numericgt
6487	int8numericle
6488	int8numericge
6539	int21eq
6540	int21ne
6551	tinytext_send_mysql
6577	varbinary_equal_mysql
6578	b_timestampdiff
6579	b_timestampdiff
6582	b_timestampdiff
6583	b_timestampdiff
6584	b_timestampdiff
6585	b_timestampdiff
6586	b_timestampdiff
6587	b_timestampdiff
6588	b_timestampdiff
6589	b_timestampdiff
6590	b_timestampdiff
6591	b_timestampdiff
6592	b_timestampdiff
6593	b_timestampdiff
6594	b_timestampdiff
6595	b_timestampdiff
6599	blobtypmodout
6601	texttypmodout
6603	i1typmodout_mysql
6605	i2typmodout_mysql
6607	i4typmodout_mysql
6609	i8typmodout_mysql
6610	plus_any

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
6611	minus_any
6612	mul_any
6613	div_any
6614	mod_any
6615	neg_any
6616	logical_oper_bool
6617	xor_bool
6618	bit_or_any
6619	bit_and_any
6620	bit_xor_any
6621	bit_neg_any
6622	bit_shift_left_any
6623	bit_shift_right_any
6624	like_any
6625	nlike_any
6626	int_div_any
6627	tinytext_equal_mysql
6628	binary_equal_mysql
6629	tinyblob_equal_mysql
6630	cast_to_int8
6631	cast_to_float8
6632	cast_to_numeric
6633	cast_to_cstring
6634	cast_to_datetime
6635	int21lt
6636	cast_to_binary
6637	coalesce
6640	makedate
6641	maketime
6643	float4_typmod_out_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
6644	float4_mysql
6645	float4_int3_mysql
6646	float4_uint1_mysql
6647	float4_uint2_mysql
6648	float4_uint3_mysql
6649	float4_uint4_mysql
6650	float4_uint8_mysql
6652	float8_typmod_out_mysql
6653	float8_mysql
6654	float8_int3_mysql
6655	float8_uint1_mysql
6656	float8_uint2_mysql
6657	float8_uint3_mysql
6658	float8_uint4_mysql
6659	float8_uint8_mysql
6661	float8_timestamp_mysql
6662	float8_datetime_mysql
6663	float8_date_mysql
6664	float8_time_mysql
6665	text_binary_mysql
6671	text_datetime_mysql
6672	text_int8_mysql
6673	text_int4_mysql
6674	text_int2_mysql
6675	text_int1_mysql
6676	text_uint8_mysql
6677	text_uint4_mysql
6678	text_uint2_mysql
6679	text_uint1_mysql
6680	text_varchar_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
6681	text_text_mysql
6682	text_tinytext_mysql
6683	text_mediumtext_mysql
6684	text_longtext_mysql
6685	text_tinyblob_mysql
6686	text_blob_mysql
6687	text_mediumblob_mysql
6688	text_longblob_mysql
6689	text_varbinary_mysql
6690	text_year_mysql
6691	text_time_mysql
6692	text_float4_mysql
6693	text_float8_mysql
6694	text_mediumint_mysql
6699	numeric_blob_mysql
6704	numeric_uint1_mysql
6705	numeric_uint2_mysql
6706	numeric_uint4_mysql
6707	numeric_uint8_mysql
6708	numeric_mediumint_mysql
6709	numeric_year_mysql
6710	numeric_timestamp_mysql
6711	numeric_datetime_mysql
6712	numeric_time_mysql
6713	numeric_date_mysql
6714	i8toi3_mysql
6715	i8toui1_mysql
6716	i8toui2_mysql
6717	i8toui4_mysql
6718	i8toui8_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
6719	int4_year_mysql
6720	i8tobinary_mysql
6721	i8tovarbinary_mysql
6725	timestamptz_datetime_mysql
6727	i8toblob_mysql
6728	convert_tz
6729	int8_time_mysql
6730	int8_date_mysql
6731	int8_datetime_mysql
6732	int8_timestamp_mysql
6733	bit_tinyint_mysql
6734	bit_smallint_mysql
6735	bit_mediumint_mysql
6736	bit_uint1_mysql
6737	bit_uint2_mysql
6738	bit_uint4_mysql
6739	bit_uint8_mysql
6740	bit_year_mysql
6741	bit_float_mysql
6742	bit_double_mysql
6743	bit_numeric_mysql
6746	bit_text_mysql
6748	bit_char_mysql
6749	bit_varchar_mysql
6750	bit_binary_mysql
6751	bit_varbinary_mysql
6754	bit_blob_mysql
6756	bit_date_mysql
6757	bit_datetime_mysql
6758	bit_time_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
6759	bit_timestamp_mysql
6760	timestamp_int8_mysql
6761	timestamp_float4_mysql
6762	timestamp_float8_mysql
6763	timestamp_numeric_mysql
6764	timestamp_datetime_mysql
6765	timestamp_uint8_mysql
6766	timestamp_bpchar_mysql
6767	datetime_float4_mysql
6768	datetime_float8_mysql
6769	datetime_numeric_mysql
6770	datetime_time_mysql
6771	datetime_timestamp_mysql
6772	datetime_date_mysql
6773	date_float4_mysql
6774	date_float8_mysql
6775	date_numeric_mysql
6776	date_datetime_mysql
6777	date_date_mysql
6778	date_time_mysql
6779	time_float4_mysql
6780	time_float8_mysql
6781	time_numeric_mysql
6782	time_date_mysql
6783	time_datetime_mysql
6784	time_timestamp_mysql
6785	datetime_int8_mysql
6786	date_int8_mysql
6787	time_int8_mysql
6788	time_uint4_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
6789	time_uint8_mysql
6790	time_int4_mysql
6791	datetime_uint8_mysql
6792	date_uint8_mysql
6793	timestamp_binary_mysql
6794	timestamp_varbinary_mysql
6795	datetime_bpchar_mysql
6796	datetime_varchar_mysql
6797	datetime_binary_mysql
6798	datetime_varbinary_mysql
6799	datetime_text_mysql
6800	date_binary_mysql
6801	date_varbinary_mysql
6802	time_bpchar_mysql
6803	time_varchar_mysql
6804	time_binary_mysql
6805	time_varbinary_mysql
6806	time_text_mysql
6810	binary_send_mysql
6811	binary_recv_mysql
6812	binary_typmodin_mysql
6813	binary_typmodout_mysql
6814	int12eq
6815	numericint1eq
6816	binary_float_mysql
6817	binary_int4_mysql
6818	mediumint_equal_mysql
6822	binary_datetime_mysql
6823	binary_char_mysql
6824	binary_text_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
6825	binary_numeric_mysql
6826	binary_mysql
6827	blob_recv_mysql
6828	blob_send_mysql
6845	longblob_recv_mysql
6846	longblob_send_mysql
6853	int168ge
6859	longtext_recv_mysql
6860	longtext_send_mysql
6871	mediumblob_recv_mysql
6872	mediumblob_send_mysql
6939	rowid_in
6940	rowid_out
6942	rowid_send
6945	mediumtext_recv_mysql
6946	mediumtext_send_mysql
6954	rowid_recv
6956	chartorowid
6959	tinyblob_recv_mysql
6960	tinyblob_send_mysql
6967	chartorowid
6969	datetime_timestamp_equal_mysql
6973	datetime_date_equal_mysql
6983	mediumblob_equal_mysql
6985	year_equal_mysql
6987	int1_equal_mysql
6993	bit_equal_mysql
6995	timestamp_datetime_equal_mysql
6999	timestamp_equal_mysql
7030	tinytext_recv_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7039	year_bit_mysql
7044	uint1_send_mysql
7045	uint1_recv_mysql
7050	uint2_send_mysql
7051	uint2_recv_mysql
7056	uint4_send_mysql
7057	uint4_recv_mysql
7062	uint8_send_mysql
7063	uint8_recv_mysql
7076	date_timestamp_equal_mysql
7093	date_datetime_equal_mysql
7095	str_to_date
7098	date_equal_mysql
7104	timestamp_date_equal_mysql
7121	longblob_equal_mysql
7136	blob_equal_mysql
7161	bool_char_mysql
7162	bool_binary_mysql
7163	year_timestamp_mysql
7164	year_time_mysql
7165	int8_int3_cmp_mysql
7168	int3_int8_cmp_mysql
7176	int4_int3_cmp_mysql
7181	varbinary_send_mysql
7182	varbinary_recv_mysql
7183	varbinary_typmodin_mysql
7184	varbinary_typmodout_mysql
7185	chartorowid
7186	varbinary_float_mysql
7187	varbinary_int_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7188	varbinary_datetime_mysql
7189	varbinary_char_mysql
7190	varbinary_text_mysql
7193	datetime_recv_mysql
7194	datetime_send_mysql
7195	datetimetypmodin_mysql
7196	datetimetypmodout_mysql
7198	datetime_tinyblob_mysql
7199	datetime_mediumblob_mysql
7200	datetime_blob_mysql
7201	datetime_longblob_mysql
7202	datetime_tinytext_mysql
7203	datetime_mediumtext_mysql
7204	datetime_longtext_mysql
7206	datetime_pl_interval
7207	interval_pl_datetime
7208	datetime_mi
7209	datetime_mi_interval
7212	time_tinyblob_mysql
7213	time_mediumblob_mysql
7214	time_blob_mysql
7215	time_longblob_mysql
7216	time_tinytext_mysql
7217	time_mediumtext_mysql
7218	time_longtext_mysql
7219	varchar_time_mysql
7220	bpchar_time_mysql
7222	timestamp_tinyblob_mysql
7223	timestamp_mediumblob_mysql
7224	timestamp_blob_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7225	timestamp_longblob_mysql
7226	timestamp_tinytext_mysql
7227	timestamp_mediumtext_mysql
7228	timestamp_longtext_mysql
7229	date_tinyblob_mysql
7230	date_mediumblob_mysql
7231	date_blob_mysql
7232	int3_int4_cmp_mysql
7235	date_longblob_mysql
7236	date_tinytext_mysql
7237	date_mediumtext_mysql
7238	date_longtext_mysql
7245	year_uint2_mysql
7246	year_uint4_mysql
7247	year_uint8_mysql
7248	year_int2_mysql
7249	year_int4_mysql
7250	year_int8_mysql
7251	year_float4_mysql
7252	year_float8_mysql
7253	year_numeric_mysql
7254	year_binary_mysql
7255	year_varbinary_mysql
7256	chartorowid
7257	rowidtobpchar
7258	year_blob_mysql
7259	rowidtochar
7260	rowidtonchar
7261	rowidtotext
7262	year_text_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7264	year_varchar_mysql
7265	year_bpchar_mysql
7266	float4_year_mysql
7267	float8_year_mysql
7268	varchar_year_mysql
7269	bpchar_year_mysql
7270	rowid_nodeid
7275	varbinary_uint8_mysql
7276	binary_uint8_mysql
7277	day
7278	month
7279	year
7280	dayofyear
7281	weekofyear
7282	quarter
7283	hour
7284	minute
7285	second
7286	microsecond
7287	to_days
7288	datediff
7289	lpad
7290	rpad
7291	left
7292	right
7293	acos
7294	asin
7295	length
7296	bit_length
7297	char_length

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7298	character_length
7299	is_true
7300	is_false
7301	is_not_true
7302	is_not_false
7303	cos
7304	week
7305	week
7306	yearweek
7307	yearweek
7308	rowid_hashbucketid
7309	degrees
7310	exp
7311	floor
7312	greatest
7313	least
7314	dayofmonth
7315	last_day
7316	weekday
7317	from_unixtime
7318	from_unixtime
7319	timediff
7320	ceiling
7321	sec_to_time
7322	rowid_tableoid
7323	repeat
7324	sin
7325	rowid_sequence
7327	dayname
7328	monthname

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7329	date_format
7330	ascii
7331	sqrt
7332	sign
7333	hex
7334	unhex
7335	substr
7336	substr
7337	truncate
7338	lower
7339	lcase
7340	upper
7341	ucase
7342	atan
7343	atan2
7344	tan
7348	replace
7349	sha
7350	sha1
7351	sha2
7352	pow
7353	power
7354	ltrim
7355	rtrim
7356	trim
7357	trim
7358	period_add
7359	period_diff
7360	get_format
7364	dayofweek

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7365	find_in_set
7366	time_format
7367	substring_index
7368	varbinary_mysql
7369	bit_or
7370	bit_and
7371	bit_xor
7372	year_uint1_mysql
7373	year_int3_mysql
7374	ui1toi3_mysql
7375	ui2toi3_mysql
7376	ui4toi3_mysql
7377	ui8toi3_mysql
7378	bit_func_char_mysql
7379	bit_func_varchar_mysql
7380	bit_func_binary_mysql
7381	bit_func_varbinary_mysql
7390	uint8_bit_mysql
7392	to_seconds
7393	time_to_sec
7394	extract
7395	abs
7396	datetime_scale_mysql
7397	timestampdiff
7398	concat_ws
7400	reverse
7401	eq_datetime
7402	eq_binary
7403	ne_datetime
7404	ne_binary

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7405	equal_int
7406	equal_numeric
7407	equal_float
7408	equal_string
7409	equal_datetime
7410	equal_time
7411	equal_binary
7412	ge_datetime
7413	ge_binary
7414	gt_datetime
7415	gt_binary
7416	le_datetime
7417	le_binary
7418	lt_datetime
7419	lt_binary
7420	text_bit_mysql
7421	blob_bit_mysql
7422	bool_int1_mysql
7423	bool_int2_mysql
7424	bool_mediumint_mysql
7425	bool_int4_mysql
7426	bool_int8_mysql
7427	bool_uint1_mysql
7428	int2_int3_cmp_mysql
7430	bool_uint2_mysql
7431	bool_uint4_mysql
7432	bool_uint8_mysql
7433	bool_real_mysql
7434	bool_double_mysql
7435	bool_numeric_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7436	booltext_mysql
7437	uint_cmp_mysql
7438	uint_sortsupport_mysql
7439	hash_uint8_mysql
7440	bool_bool_equal_mysql
7441	int8_int2_equal_mysql
7442	int3_int2_cmp_mysql
7443	year_cmp_mysql
7444	year_sortsupport
7445	datetime_cmp_mysql
7446	datetime_sortsupport
7447	datetime_cmp_date
7448	datetime_cmp_timestamp
7449	datetime_cmp_time
7450	date_cmp_datetime
7451	timestamp_cmp_datetime
7452	time_cmp_datetime
7453	datetime_cmp_timestamp
7454	datetime_cmp_date
7455	datetime_cmp_time
7456	date_cmp_datetime
7457	time_cmp_datetime
7458	timestamp_cmp_datetime
7459	sum
7460	sum_any
7461	avg
7462	avg_any
7463	avg_final
7464	max
7465	max_any

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7466	min
7467	min_any
7468	sum_final
7469	avg_collect
7470	int1_int16_mysql
7471	int3_int16_mysql
7472	uint1_int16_mysql
7473	uint2_int16_mysql
7474	uint3_int16_mysql
7475	uint4_int16_mysql
7476	uint8_int16_mysql
7477	int16_int1_mysql
7478	int16_int3_mysql
7479	int16_uint1_mysql
7480	int16_uint2_mysql
7481	int16_uint3_mysql
7482	int16_uint4_mysql
7483	int16_uint8_mysql
7484	mediumint_eq_mysql
7485	mediumint_ne_mysql
7486	mediumint_lt_mysql
7487	mediumint_le_mysql
7488	mediumint_gt_mysql
7489	mediumint_ge_mysql
7501	date_int1_mysql
7502	date_int2_mysql
7503	date_int3_mysql
7504	date_int4_mysql
7505	datetime_int1_mysql
7506	datetime_int2_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7507	datetime_int3_mysql
7508	datetime_int4_mysql
7509	timestamp_int1_mysql
7510	timestamp_int2_mysql
7511	timestamp_int3_mysql
7512	timestamp_int4_mysql
7514	binary_eq_mysql
7515	varbinary_eq_mysql
7516	tinyblob_eq_mysql
7517	mediumblob_eq_mysql
7518	longblob_eq_mysql
7519	binary_ne_mysql
7520	varbinary_ne_mysql
7521	tinyblob_ne_mysql
7522	mediumblob_ne_mysql
7523	longblob_ne_mysql
7524	binary_lt_mysql
7525	varbinary_lt_mysql
7526	tinyblob_lt_mysql
7527	mediumblob_lt_mysql
7528	longblob_lt_mysql
7529	binary_le_mysql
7530	varbinary_le_mysql
7531	tinyblob_le_mysql
7532	mediumblob_le_mysql
7533	longblob_le_mysql
7534	binary_gt_mysql
7535	varbinary_gt_mysql
7536	tinyblob_gt_mysql
7537	mediumblob_gt_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7538	longblob_gt_mysql
7539	binary_ge_mysql
7540	varbinary_ge_mysql
7541	tinyblob_ge_mysql
7542	mediumblob_ge_mysql
7543	longblob_ge_mysql
7544	tinytext_eq_mysql
7545	mediumtext_eq_mysql
7546	longtext_eq_mysql
7547	tinytext_ne_mysql
7548	mediumtext_ne_mysql
7549	longtext_ne_mysql
7550	tinytext_lt_mysql
7551	mediumtext_lt_mysql
7552	longtext_lt_mysql
7553	tinytext_le_mysql
7554	mediumtext_le_mysql
7555	longtext_le_mysql
7556	tinytext_gt_mysql
7557	mediumtext_gt_mysql
7558	longtext_gt_mysql
7559	tinytext_ge_mysql
7560	mediumtext_ge_mysql
7561	longtext_ge_mysql
7562	date_literal_mysql
7563	time_literal_mysql
7564	datetime_literal_mysql
7568	year_eq_mysql
7569	year_ne_mysql
7570	year_lt_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7571	year_le_mysql
7572	year_gt_mysql
7573	year_ge_mysql
7574	time
7575	date
7576	timestamp
7577	timestamp
7582	uint3_send_mysql
7583	uint3_recv_mysql
7600	i8toui3_mysql
7605	numeric_uint3_mysql
7606	bit_uint3_mysql
7607	bool_uint3_mysql
7608	year_uint3_mysql
7609	text_uint3_mysql
7610	time_bit_mysql
7611	date_bit_mysql
7612	datetime_bit_mysql
7613	timestamp_bit_mysql
7614	date_uint1_mysql
7615	date_uint2_mysql
7616	date_uint3_mysql
7617	date_uint4_mysql
7618	datetime_uint1_mysql
7619	datetime_uint2_mysql
7620	datetime_uint3_mysql
7621	datetime_uint4_mysql
7622	timestamp_uint1_mysql
7623	timestamp_uint2_mysql
7624	timestamp_uint3_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7625	timestamp_uint4_mysql
7635	ln
7636	log2
7637	log10
7638	log
7639	log
7640	mysql_compare_int
7641	mysql_compare_numeric
7642	mysql_compare_float
7643	mysql_compare_string
7644	mysql_compare_datetime
7645	mysql_compare_time
7646	mysql_compare_binary
7653	int2_int3_eq_mysql
7654	int2_int3_ne_mysql
7655	int2_int3_equal_mysql
7656	int2_int3_gt_mysql
7657	int2_int3_lt_mysql
7658	int2_int3_ge_mysql
7659	int2_int3_le_mysql
7660	int3_int2_eq_mysql
7661	int3_int2_ne_mysql
7662	int3_int2_equal_mysql
7663	int3_int2_gt_mysql
7664	int3_int2_lt_mysql
7665	int3_int2_ge_mysql
7666	int3_int2_le_mysql
7667	int3_int4_eq_mysql
7668	int3_int4_ne_mysql
7669	int3_int4_equal_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7670	int3_int4_gt_mysql
7671	int3_int4_lt_mysql
7672	int3_int4_ge_mysql
7673	int3_int4_le_mysql
7674	int3_int8_eq_mysql
7675	int3_int8_ne_mysql
7676	int3_int8_equal_mysql
7677	int3_int8_gt_mysql
7678	int3_int8_lt_mysql
7679	int3_int8_ge_mysql
7680	int3_int8_le_mysql
7681	int4_int3_eq_mysql
7682	int4_int3_ne_mysql
7683	int4_int3_equal_mysql
7684	int4_int3_gt_mysql
7685	int4_int3_lt_mysql
7686	int4_int3_ge_mysql
7687	int4_int3_le_mysql
7688	int8_int3_eq_mysql
7689	int8_int3_ne_mysql
7690	int8_int3_equal_mysql
7691	int8_int3_gt_mysql
7692	int8_int3_lt_mysql
7693	int8_int3_ge_mysql
7694	int8_int3_le_mysql
7695	binary_int8_mysql
7696	blob_ne_mysql
7697	blob_lt_mysql
7698	blob_le_mysql
7699	blob_gt_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7700	blob_ge_mysql
7701	blob_eq_mysql
7717	rowid_eq
7718	rowid_ne
7719	rowid_lt
7720	rowid_le
7721	rowid_gt
7722	rowid_ge
7723	rowid_cmp
7724	rowid_hash
7726	if
7727	ifnull
7728	nullif
7729	rowid_larger
7730	rowid_smaller
7737	aes_encrypt
7738	aes_encrypt
7739	aes_encrypt
7740	aes_encrypt
7741	aes_encrypt
7742	aes_decrypt
7743	aes_decrypt
7744	aes_decrypt
7745	aes_decrypt
7746	aes_decrypt
7747	numericint2le
7748	numericint2ge
7749	numericint4eq
7750	numericint4ne
7751	numericint4lt

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7752	numericint4gt
7753	numericint4le
7754	numericint4ge
7755	numericint8eq
7756	numericint8ne
7757	numericint8lt
7758	numericint8gt
7759	numericint8le
7760	numericint8ge
7761	int161eq
7762	int161ne
7763	int161lt
7766	rowid_list_agg_noarg2_transfn
7767	rowid_list_agg_transfn
7768	pivot_
7779	year_datetime_mysql
7788	year_date_mysql
7790	uint_int3_eq_mysql
7791	uint_int3_ne_mysql
7792	uint_int3_equal_mysql
7793	uint_int3_ge_mysql
7797	int3_uint_ne_mysql
7818	datetime_year_mysql
7821	attname_eq_mysql
7822	date_func_int2_mysql
7823	date_func_int3_mysql
7824	date_func_int4_mysql
7825	date_func_int8_mysql
7826	date_func_uint1_mysql
7827	date_func_uint2_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7828	date_func_uint3_mysql
7829	date_func_uint4_mysql
7830	date_func_uint8_mysql
7831	date_func_bit_mysql
7832	date_row_int1_mysql
7833	date_row_int2_mysql
7834	date_row_int3_mysql
7835	date_row_int4_mysql
7836	date_row_int8_mysql
7837	date_row_uint1_mysql
7838	date_row_uint2_mysql
7839	date_row_uint3_mysql
7840	date_row_uint4_mysql
7841	date_row_uint8_mysql
7842	date_row_float4_mysql
7843	date_row_float8_mysql
7844	date_row_numeric_mysql
7845	date_row_bit_mysql
7846	time_func_bit_mysql
7847	time_row_int1_mysql
7848	time_row_int2_mysql
7849	time_row_int3_mysql
7850	time_row_int4_mysql
7851	time_row_int8_mysql
7852	time_row_uint1_mysql
7853	time_row_uint2_mysql
7854	time_row_uint3_mysql
7855	time_row_uint4_mysql
7856	time_row_uint8_mysql
7857	time_row_float4_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7858	time_row_float8_mysql
7859	time_row_numeric_mysql
7860	time_row_bit_mysql
7861	datetime_func_bit_mysql
7862	datetime_row_int1_mysql
7863	datetime_row_int2_mysql
7864	datetime_row_int3_mysql
7865	datetime_row_int4_mysql
7866	datetime_row_int8_mysql
7867	datetime_row_uint1_mysql
7868	datetime_row_uint2_mysql
7869	datetime_row_uint3_mysql
7870	datetime_row_uint4_mysql
7871	datetime_row_uint8_mysql
7872	datetime_row_float4_mysql
7873	datetime_row_float8_mysql
7874	datetime_row_numeric_mysql
7875	datetime_row_bit_mysql
7876	timestamp_row_int1_mysql
7877	timestamp_row_int2_mysql
7878	timestamp_row_int3_mysql
7879	timestamp_row_int4_mysql
7880	timestamp_row_int8_mysql
7897	timestamp_row_uint1_mysql
7898	timestamp_row_uint2_mysql
7899	timestamp_row_uint3_mysql
7900	timestamp_row_uint4_mysql
7901	timestamp_row_uint8_mysql
7902	timestamp_row_float4_mysql
7903	timestamp_row_float8_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
7904	timestamp_row_numeric_mysql
7905	timestamp_row_bit_mysql
7906	date_func_year_mysql
7907	date_row_year_mysql
7908	date_row_time_mysql
7909	time_func_year_mysql
7910	time_row_year_mysql
7911	time_row_date_mysql
7912	time_row_datetime_mysql
7913	time_row_timestamp_mysql
7914	datetime_func_year_mysql
7915	datetime_row_year_mysql
7916	datetime_row_date_mysql
7917	datetime_row_time_mysql
7918	timestamp_row_year_mysql
7919	timestamp_row_date_mysql
7920	timestamp_row_time_mysql
7921	date_func_int1_mysql
8117	uint_int3_lt_mysql
8234	binarycmp_mysql
8300	bpchar_bit_mysql
8523	uint_int8_eq_mysql
8524	uint_int8_ne_mysql
8525	uint_int8_equal_mysql
8526	uint_int8_ge_mysql
8527	uint_int8_gt_mysql
8528	uint_int8_le_mysql
8529	uint_int8_lt_mysql
8530	int8_uint_eq_mysql
8531	int8_uint_ne_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
8532	int8_uint_equal_mysql
8533	int8_uint_ge_mysql
8534	int8_uint_gt_mysql
8535	int8_uint_le_mysql
8536	int8_uint_lt_mysql
8537	uint_int4_eq_mysql
8538	uint_int4_ne_mysql
8539	uint_int4_equal_mysql
8540	uint_int4_ge_mysql
8541	uint_int4_gt_mysql
8542	uint_int4_le_mysql
8543	uint_int4_lt_mysql
8544	int4_uint_eq_mysql
8545	int4_uint_ne_mysql
8546	int4_uint_equal_mysql
8547	int4_uint_ge_mysql
8548	int4_uint_gt_mysql
8549	int4_uint_le_mysql
8550	int4_uint_lt_mysql
8627	uint_int2_eq_mysql
8628	uint_int2_ne_mysql
8629	uint_int2_equal_mysql
8630	uint_int2_ge_mysql
8631	uint_int2_gt_mysql
8632	uint_int2_le_mysql
8633	uint_int2_lt_mysql
8634	int2_uint_eq_mysql
8635	int2_uint_ne_mysql
8636	int2_uint_equal_mysql
8637	int2_uint_ge_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
8638	int2_uint_gt_mysql
8639	int2_uint_le_mysql
8640	int2_uint_lt_mysql
8705	uint_uint_eq_mysql
8706	uint_uint_ne_mysql
8707	uint_uint_equal_mysql
8708	uint_uint_ge_mysql
8709	uint_uint_gt_mysql
8710	uint_uint_le_mysql
8711	uint_uint_lt_mysql
8746	int3_uint_equal_mysql
8747	int3_uint_ge_mysql
8748	int3_uint_gt_mysql
8749	int3_uint_le_mysql
8750	int3_uint_lt_mysql
8751	int161gt
8752	int161le
8753	int161ge
8754	int162eq
8755	int162ne
8756	int162lt
8757	int162gt
8758	int162le
8759	int162ge
8760	int164eq
8761	int164ne
8762	int164lt
8763	int164gt
8764	int164le
8765	int164ge

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
8766	int168eq
8767	int168ne
8768	int168lt
8769	int168gt
8770	int168le
8815	concat
8983	uint_int2_cmp_mysql
8984	uint_int3_cmp_mysql
8985	uint_int4_cmp_mysql
8986	uint_int8_cmp_mysql
8987	int2_uint_cmp_mysql
8988	int3_uint_cmp_mysql
8989	int4_uint_cmp_mysql
8990	int8_uint_cmp_mysql
9000	date_add
9011	smalldatetime_smaller
9012	smalldatetime_larger
9362	hash_meet_partition
9548	mediumblobtypmodout
9555	mediumtexttypmodout
9557	tinytexttypmodout
9558	int12ne
9559	int12lt
9560	int12gt
9561	int12le
9562	int12ge
9563	int14eq
9564	int14ne
9566	int14lt
9567	int14gt

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
9568	int14le
9569	int14ge
9573	int18eq
9574	int18ne
9575	int18lt
9576	int18gt
9584	int18le
9585	int18ge
9586	int116eq
9587	int116ne
9588	int116lt
9589	int116gt
9590	int116le
9591	int116ge
9592	int1numericeq
9593	int1numericne
9594	int1numericlt
9595	int1numericgt
9596	int1numericle
9597	int1numericge
9624	numericint1ne
9625	numericint1lt
9626	numericint1gt
9627	numericint1le
9628	numericint1ge
9629	numericint2eq
9630	numericint2ne
9631	numericint2lt
9632	numericint2gt
9749	time_uint1_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
9750	time_int1_mysql
9751	time_uint2_mysql
9752	time_int2_mysql
9753	time_uint3_mysql
9754	time_int3_mysql
9807	datea_pl_text
9808	datea_mi_text
9809	text_pl_datea
9810	datea_pl_int8
9811	datea_mi_int8
9812	int8_pl_datea
9813	datea_pl_float4
9814	datea_mi_float4
9815	float4_pl_datea
9816	datea_pl_float8
9817	datea_mi_float8
9818	float8_pl_datea
9819	datea_pl_numeric
9820	datea_mi_numeric
9821	numeric_pl_datea
9887	float4_blob_mysql
9888	float4_varbinary_mysql
9889	float4_binary_mysql
9890	float8_blob_mysql
9891	float8_varbinary_mysql
9892	float8_binary_mysql
9910	substring_index
9934	datetime_date_eq_mysql
9935	date_datetime_eq_mysql
9936	datetime_timestamp_eq_mysql

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

func_oid_value	func_name
9937	timestamp_datetime_eq_mysql
9938	datetime_date_gt_mysql
9939	date_datetime_gt_mysql
9940	datetime_timestamp_gt_mysql
9941	timestamp_datetime_gt_mysql
9942	datetime_date_lt_mysql
9943	date_datetime_lt_mysql
9944	datetime_timestamp_lt_mysql
9945	timestamp_datetime_lt_mysql
9946	datetime_date_le_mysql
9947	date_datetime_le_mysql
9948	datetime_timestamp_le_mysql
9949	timestamp_datetime_le_mysql
9950	datetime_date_ge_mysql
9951	date_datetime_ge_mysql
9952	datetime_timestamp_ge_mysql
9953	timestamp_datetime_ge_mysql
9954	datetime_date_ne_mysql
9955	date_datetime_ne_mysql
9956	datetime_timestamp_ne_mysql
9957	timestamp_datetime_ne_mysql
9964	tinyblobtypmodout

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2 Stored Procedures

2.1 Overview

In GaussDB, business rules and logics are saved as stored procedures.

A stored procedure is a combination of SQL, PL/SQL, and Java statements. With stored procedures, the code for executing business rules can be migrated from the application layer to the database layer, thereby reusing the code. That is, the code, once stored, can be called by multiple programs.

For details about how to create and call a stored procedure, see **CREATE PROCEDURE**.

The application methods for PL/SQL functions mentioned in "User-defined Functions > PL/SQL Functions" in *Developer Guide* are the same as those for stored procedures. Unless otherwise specified, the following sections apply to stored procedures and PL/SQL functions.

2.2 Data Types

A data type refers to a value set and an operation set defined on the value set. GaussDB consists of tables, each of which is defined by its own columns. Each column corresponds to a data type. GaussDB provides corresponding functions to perform operations on data based on data types. For example, GaussDB can perform operations such as addition, subtraction, multiplication, and division on numeric data.

XML data can be used as input parameters, output parameters, user-defined variables, and return values of stored procedures. Stored procedures in autonomous transactions are supported.

2.3 Data Type Conversion

In the database system, some data types support implicit type conversion (such as assignment and function call parameters), while some data types (such as INT and composite types) do not support implicit type conversion. You can use the type conversion function provided by GaussDB to forcibly convert the data type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-1 lists common implicit data type conversions in GaussDB.

NOTICE

The valid value range of **DATE** supported by GaussDB is from 4713 B.C. to 294276 A.D.

Table 2-1 Implicit data type conversions

Raw Data Type	Target Data Type	Remarks
CHAR	VARCHAR2	-
CHAR	NUMBER	Raw data must consist of digits.
CHAR	DATE	Raw data cannot exceed the valid date range.
CHAR	RAW	-
CHAR	CLOB	-
VARCHAR2	CHAR	-
VARCHAR2	NUMBER	Raw data must consist of digits.
VARCHAR2	DATE	Raw data cannot exceed the valid date range.
VARCHAR2	CLOB	-
NUMBER	CHAR	-
NUMBER	VARCHAR2	-
DATE	CHAR	-
DATE	VARCHAR2	-
RAW	CHAR	-
RAW	VARCHAR2	-
CLOB	CHAR	-
CLOB	VARCHAR2	-
CLOB	NUMBER	Raw data must consist of digits.
INT4	CHAR	-
DATEA	TEXT	-
DATEA	VARCHAR	-
DATEA	CHAR	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Raw Data Type	Target Data Type	Remarks
DATEA	CLOB	-
DATEA	NVARCHAR2	-
DATEA	TIMESTAMP WITH TIME ZONE	-
DATEA	TIMESTAMP WITHOUT TIME ZONE	-
TEXT	DATEA	The original data cannot exceed the valid date range and must comply with the format specified by **nls_date_format**.
VARCHAR	DATEA	The original data cannot exceed the valid date range and must comply with the format specified by **nls_date_format**.
CHAR	DATEA	The original data cannot exceed the valid date range and must comply with the format specified by **nls_date_format**.
CLOB	DATEA	The original data cannot exceed the valid date range and must comply with the format specified by **nls_date_format**.
NVARCHAR2	DATEA	The original data cannot exceed the valid date range and must comply with the format specified by **nls_date_format**.

2.4 Arrays, Collections, and Records

2.4.1 Arrays

2.4.1.1 Use of Array Types

Before the use of arrays, an array type needs to be defined:

Syntax

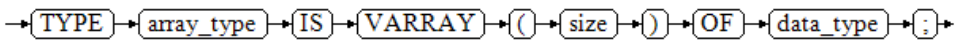
Define an array type immediately after the AS keyword in a stored procedure. The definition method is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
TYPE array_type IS VARRAY(size) OF data_type;
```

Figure 2-1 shows the syntax diagram.

Figure 2-1 Syntax for defining an array type



In the preceding information:

- **array_type**: name of the array type to be defined.
- **size**: maximum number of members in the array type to be defined. The value is a positive integer.
- **data_type**: type of members in the array type to be created.

Precautions

- GaussDB supports access to array elements using index syntax (enclosed by parentheses) and provides various dedicated functions (such as EXTEND, COUNT, FIRST, LAST, PRIOR, EXISTS, TRIM, NEXT, and DELETE) for flexible array operations and management.
- In GaussDB, an array is auto-incremental. If out-of-bounds access occurs, a **NULL** value will be returned, and no error message will be reported.
- The scope of an array type defined in a stored procedure is only limited to this stored procedure.
- **size** is used only for syntax compatibility and has no actual meaning.
- **data_type** can also be the record type defined in a stored procedure (anonymous blocks are not supported) or collection type, but cannot be the array or cursor type defined in a stored procedure.
- When **data_type** is defined as a collection, multi-dimensional arrays are not supported.
- Constructors of the array type can be used only in ORA-compatible mode.
- A constructor of the array type cannot be used as the default value of a function or stored procedure parameter.
- If the elements of an array are of the collection type and **data_type** of the array is set to **VARCHAR** or **NUMERIC** that can define the length and precision, you need to enable the **tableof_elem_constraints** option for GUC parameter **behavior_compat_options** to verify the element length of the array or convert the element to the corresponding precision.
- After **enable_recordtype_check_strict** is set to **on**, if the elements are of the record type and a column of the record type has the NOT NULL or DEFAULT attribute, an error will be reported during stored procedure or package compilation.
- If the stored procedure contains DML statements (SELECT, UPDATE, INSERT, or DELETE), DML statements can access array elements only by using brackets, so as to differentiate it from a function expression.
- The NOT NULL syntax is not supported.
- Elements of varray variables cannot be accessed by DML statements.
- Custom array types cannot be pushed down. In pushdown scenarios, variables are used to carry array elements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- When you access an array member using an index, the parser preferentially interprets the array variable as a table column name. If the table column name is ambiguous, for example, two tables have columns with the same name, an error will occur.

Examples

```
-- Perform array operations in a stored procedure.
gaussdb=# CREATE OR REPLACE PROCEDURE array_proc AS
DECLARE
    TYPE ARRAY_INTEGER IS VARRAY(1024) OF INTEGER;-- Define an array type.
    ARRINT ARRAY_INTEGER: = ARRAY_INTEGER(); -- Declare the variable of the array type.
BEGIN
    ARRINT.EXTEND(10);
    FOR I IN 1..10 LOOP
        ARRINT(I) := I;
    END LOOP;
    DBE_OUTPUT.PRINT_LINE(ARRINT.COUNT);
    DBE_OUTPUT.PRINT_LINE(ARRINT(1));
    DBE_OUTPUT.PRINT_LINE(ARRINT(10));
    DBE_OUTPUT.PRINT_LINE(ARRINT(ARRINT.FIRST));
    DBE_OUTPUT.PRINT_LINE(ARRINT(ARRINT.LAST));
    DBE_OUTPUT.PRINT_LINE(ARRINT(ARRINT.NEXT(ARRINT.FIRST)));
    DBE_OUTPUT.PRINT_LINE(ARRINT(ARRINT.PRIOR(ARRINT.LAST)));
    ARRINT.TRIM();
    IF ARRINT.EXISTS(10) THEN
       DBE_OUTPUT.PRINT_LINE('Exist 10th element');
    ELSE
       DBE_OUTPUT.PRINT_LINE('Not exist 10th element');
    END IF;
    DBE_OUTPUT.PRINT_LINE(ARRINT.COUNT);
    DBE_OUTPUT.PRINT_LINE(ARRINT(ARRINT.FIRST));
    DBE_OUTPUT.PRINT_LINE(ARRINT(ARRINT.LAST));
    ARRINT.DELETE();
END;
/

-- Call a stored procedure.
gaussdb=# CALL array_proc();
10
1
10
1
10
2
9
Not exist 10th element
9
1
9
 array_proc
------------

(1 row)


-- Drop a stored procedure.
gaussdb=# DROP PROCEDURE array_proc;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2.4.1.2 Array Functions

NOTE

- The following examples are applicable in ORA-compatible mode. In the function definition, the content in [] is optional. For example, count[()] can be written as count or count().
- Currently, the following errors are not supported: "An uninitialized collection is referenced" (array uninitialized), "Subscript beyond count" (array index out-of-bounds), and "Subscript outside of limit" (array index out-of-limit).
- In an inner expression, functions of the array type cannot be called in nesting mode.
- The extend(count), delete(idx), and trim(n) functions support only a single constant or variable and do not support parameters in other forms, such as function calls and expressions.
- When types are nested and the array type is used as the internal type, the extend, delete, and trim functions without parameters can only be called as extend, delete, and trim, but not as extend(), delete(), or trim().

- extend[(count)]

  Parameter: *count* of the int4 type.

  Return value: none.

  Description: Extends one or *count* elements at the end of a varray variable. The default value is **NULL**.

  Example 1: Extend one element. The default value is **NULL**.

```
gaussdb=# DECLARE
gaussdb-#     TYPE array_integer IS VARRAY(10) OF INTEGER;
gaussdb-#     arrint array_integer := array_integer();
gaussdb-# BEGIN
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$#     arrint.EXTEND;
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$# END;
gaussdb$# /
0
1
ANONYMOUS BLOCK EXECUTE
```

  Example 2: Extend *count* elements. The default value is **NULL**.

```
gaussdb=# DECLARE
gaussdb-#     TYPE array_integer IS VARRAY(10) OF INTEGER;
gaussdb-#     arrint array_integer := array_integer();
gaussdb-# BEGIN
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$#     arrint.EXTEND(3);
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$# END;
gaussdb$# /
0
3
ANONYMOUS BLOCK EXECUTE
```

- extend(count, idx)

  Parameters: *idx* and *count* are of the INT4 type.

  Return value: none.

  Description: Extends *count* elements at the end of a varray variable, with the values of the extended elements equal to the value of the element at the *idx* position.

  Example: Extend *count* elements at the *idx* position.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SET a_format_version='10c';
SET
gaussdb=# SET a_format_dev_version='s1';
SET
gaussdb=# DECLARE
gaussdb-#     TYPE array_integer IS VARRAY(10) OF INTEGER;
gaussdb-#     arrint array_integer := array_integer(1,2);
gaussdb-# BEGIN
gaussdb$#     arrint.EXTEND(2, 1);
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$#     FOR i IN 1..arrint.COUNT LOOP
gaussdb$#         DBE_OUTPUT.PRINT_LINE(arrint(i));
gaussdb$#     END LOOP;
gaussdb$# END;
gaussdb$# /
4
1
2
1
1
ANONYMOUS BLOCK EXECUTE
```

- count[()]

  Parameter: none.

  Return value: INT4 type.

  Description: Returns the number of elements in an array.

  Example: View the number of initialized array elements.

```
gaussdb=# DECLARE
gaussdb-#     TYPE array_integer IS VARRAY(10) OF INTEGER;
gaussdb-#     arrint array_integer;
gaussdb-#     len INT;
gaussdb-# BEGIN
gaussdb$#     arrint := array_integer(1, 2, 3);
gaussdb$#     len := arrint.COUNT();
gaussdb$#     DBE_OUTPUT.PRINT_LINE(len);
gaussdb$# END;
gaussdb$# /
3
ANONYMOUS BLOCK EXECUTE
```

- trim[(n)]

  Parameter: *n* is of the INT4 type.

  Return value: none.

  Description: Deletes an element at the end of the array if there is no parameter. Alternatively, deletes a specified number of elements at the end of the array if the input parameter *n* is specified.

  Example: Delete *n* elements from an array.

```
gaussdb=# DECLARE
gaussdb-#     TYPE array_integer IS VARRAY(10) OF INTEGER;
gaussdb-#     arrint array_integer := array_integer(1,2,3);
gaussdb-# BEGIN
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$#     arrint.TRIM(1);
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$# END;
gaussdb$# /
3
2
ANONYMOUS BLOCK EXECUTE
-- If n is greater than the number of array elements, clear all array elements.
gaussdb=# DECLARE
gaussdb-#     TYPE array_integer IS VARRAY(10) OF INTEGER;
gaussdb-#     arrint array_integer := array_integer(1,2,3);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb-# BEGIN
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$#     arrint.TRIM(4);
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$# END;
gaussdb$# /
3
0
ANONYMOUS BLOCK EXECUTE
```

- delete[()]

  Parameter: none.

  Return value: No value is returned.

  Description: Clears elements in an array.

  Example: Clear elements in an array.

```
gaussdb=# DECLARE
gaussdb-#     TYPE array_integer IS VARRAY(10) OF INTEGER;
gaussdb-#     arrint array_integer := array_integer(1,2,3);
gaussdb-# BEGIN
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$#     arrint.DELETE();
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$# END;
gaussdb$# /
3
0
ANONYMOUS BLOCK EXECUTE. All array elements are NULL.
-- The array is not initialized.
gaussdb=# DECLARE
gaussdb-#     TYPE array_integer IS VARRAY(10) OF INTEGER;
gaussdb-#     arrint array_integer;
gaussdb-# BEGIN
gaussdb$#     arrint.DELETE();
gaussdb$# END;
gaussdb$# /
ANONYMOUS BLOCK EXECUTE
```

- delete(idx)

  Parameter: INT4 type.

  Return value: none.

  Description: Deletes the element with the specified index *idx* from an array if the specified index *idx* is within the array range.

  Example 1: The given index *idx* is within the array range.

```
gaussdb=# DECLARE
gaussdb-#     TYPE array_integer IS VARRAY(10) OF INTEGER;
gaussdb-#     arrint array_integer := array_integer(1, 2, 3);
gaussdb-# BEGIN
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$#     arrint.DELETE(2);
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$#     FOR i IN 1..arrint.COUNT LOOP
gaussdb$#         DBE_OUTPUT.PRINT_LINE(arrint(i));
gaussdb$#     END LOOP;
gaussdb$# END;
gaussdb$# /
3
2
1
3
ANONYMOUS BLOCK EXECUTE
```

  Example 2: The given index *idx* is out of the array range.

```
gaussdb=# DECLARE
gaussdb-#     TYPE array_integer IS VARRAY(10) OF INTEGER;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb-#     arrint array_integer := array_integer(1, 2, 3);
gaussdb-# BEGIN
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$#     arrint.DELETE(2);
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$#     FOR i IN 1..arrint.COUNT LOOP
gaussdb$#         DBE_OUTPUT.PRINT_LINE(arrint(i));
gaussdb$#     END LOOP;
gaussdb$# END;
gaussdb$# /
3
INFO:  {1,2,3}
ANONYMOUS BLOCK EXECUTE

gaussdb=# DECLARE
gaussdb-#     TYPE array_integer IS VARRAY(10) OF INTEGER;
gaussdb-#     arrint array_integer:= array_integer(1, 2, 3);
gaussdb-#BEGIN
gaussdb$#     DBE_OUTPUT.PRINT_LINE(arrint.COUNT);
gaussdb$#     arrint.DELETE(11);
gaussdb$#     RAISE INFO '%', arrint;
gaussdb$# END;
gaussdb$# /
3
INFO:  {1,2,3}
ANONYMOUS BLOCK EXECUTE
```

- first[()]

  Parameter: none.

  Return value: INT4 type.

  Description: Returns the index of the first element in an array. If there is no first element, **NULL** is returned.

  Example:

```
gaussdb=# DECLARE
gaussdb-#     TYPE varr IS VARRAY(10) OF VARCHAR(3);
gaussdb-#     v varr := varr('asd','zxc');
gaussdb-# BEGIN
gaussdb$#     RAISE INFO 'first is %',v.FIRST();
gaussdb$# END;
gaussdb$# /
INFO:  first is 1
ANONYMOUS BLOCK EXECUTE
-- If the array is not initialized, NULL is returned.
gaussdb=# DECLARE
gaussdb-#     TYPE varr IS VARRAY(10) OF VARCHAR(3);
gaussdb-#     v varr;
gaussdb-# BEGIN
gaussdb$#     RAISE INFO 'first is %',v.FIRST;
gaussdb$# END;
gaussdb$# /
INFO:  first is <NULL>
ANONYMOUS BLOCK EXECUTE
```

- last[()]

  Parameter: none.

  Return value: INT4 type.

  Description: Returns the index of the last element in an array. If there is no last element, **NULL** is returned.

  Example:

```
gaussdb=# DECLARE
gaussdb-#     TYPE varr IS VARRAY(10) OF VARCHAR(3);
gaussdb-#     v varr := varr('asd','zxc');
gaussdb-# BEGIN
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb$#     RAISE INFO 'last is %',v.LAST();
gaussdb$# END;
gaussdb$# /
INFO:  last is 2
ANONYMOUS BLOCK EXECUTE
-- If the array is not initialized, NULL is returned.
gaussdb=# DECLARE
gaussdb-#     TYPE varr IS VARRAY(10) OF VARCHAR(3);
gaussdb-#     v varr;
gaussdb-# BEGIN
gaussdb$#     RAISE INFO 'last is %',v.LAST;
gaussdb$# END;
gaussdb$# /
INFO:  last is <NULL>
ANONYMOUS BLOCK EXECUTE
```

- prior(idx)

  Parameter: INT4 type.

  Return value: INT4 type.

  Description: Returns the index of the element before the specified index in an array. If the index of the element cannot be found, **NULL** is returned.

  Example:

```
gaussdb=# DECLARE
gaussdb-#     TYPE varr IS VARRAY(10) OF VARCHAR(3);
gaussdb-#     v varr := varr('asd','zxc');
gaussdb-# BEGIN
gaussdb$#     RAISE INFO 'prior is %',v.PRIOR(2);
gaussdb$# END;
gaussdb$# /
INFO:  prior is 1
ANONYMOUS BLOCK EXECUTE
-- If the array is not initialized, NULL is returned.
gaussdb=# DECLARE
gaussdb-#     TYPE varr IS VARRAY(10) OF VARCHAR(3);
gaussdb-#     v varr;
gaussdb-# BEGIN
gaussdb$#     RAISE INFO 'prior is %',v.PRIOR(2);
gaussdb$# END;
gaussdb$# /
INFO:  prior is <NULL>
ANONYMOUS BLOCK EXECUTE
-- The index is out of range and is greater than the array range.
gaussdb=# DECLARE
gaussdb-#     TYPE varr IS VARRAY(10) OF VARCHAR(3);
gaussdb-#     v varr := varr('asd','zxc','qwe');
gaussdb-# BEGIN
gaussdb$#     RAISE INFO 'prior is %',v.PRIOR(10);
gaussdb$# END;
gaussdb$# /
INFO:  prior is 3
ANONYMOUS BLOCK EXECUTE
```

- next(idx)

  Parameter: int4 type.

  Return value: INT4 type.

  Description: Returns the index of the element following the specified index in an array. If the index of an element cannot be found, **NULL** is returned.

  Example:

```
gaussdb=# DECLARE
gaussdb-#     TYPE varr IS VARRAY(10) OF VARCHAR(3);
gaussdb-#     v varr := varr('asd','zxc');
gaussdb-# BEGIN
gaussdb$#     RAISE INFO 'next is %',v.NEXT(1);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb$# END;
gaussdb$# /
INFO:  next is 2
ANONYMOUS BLOCK EXECUTE
-- If the array is not initialized, NULL is returned.
gaussdb=# DECLARE
gaussdb-#     TYPE varr IS VARRAY(10) OF VARCHAR(3);
gaussdb-#     v varr;
gaussdb-# BEGIN
gaussdb$#     RAISE INFO 'next  is %',v.NEXT(1);
gaussdb$# END;
gaussdb$# /
INFO:  next  is <NULL>
ANONYMOUS BLOCK EXECUTE
-- The index is out of range and is greater than the array range.
gaussdb=# DECLARE
gaussdb-#     TYPE varr IS VARRAY(10) OF VARCHAR(3);
gaussdb-#     v varr := varr('asd','zxc','qwe');
gaussdb-# BEGIN
gaussdb$#     RAISE INFO 'next  is %',v.NEXT(10);
gaussdb$# END;
gaussdb$# /
INFO:  next  is <NULL>
ANONYMOUS BLOCK EXECUTE
```

- exists(idx)

  Parameter: INT4 type.

  Return value: **TRUE** or **FALSE**, of the Boolean type.

  Function: Checks whether an element exists in a specified position.

  Example:

```
gaussdb=# DECLARE
gaussdb-#     TYPE varr IS VARRAY(10) OF VARCHAR(3);
gaussdb-#     v varr := varr('asd','zxc');
gaussdb-#     flag BOOL;
gaussdb-# BEGIN
gaussdb$#     flag := v.EXISTS(1);
gaussdb$#     RAISE INFO '%',flag;
gaussdb$#     flag := v.EXISTS(3);
gaussdb$#     RAISE INFO '%',flag;
gaussdb$#     flag := v.EXISTS(7);
gaussdb$#     RAISE INFO '%',flag;
gaussdb$# END;
gaussdb$# /
INFO:  t
INFO:  f
INFO:  f
ANONYMOUS BLOCK EXECUTE
-- If the array is not initialized, false is returned.
gaussdb=# DECLARE
gaussdb-#     TYPE varr IS VARRAY(10) OF VARCHAR(3);
gaussdb-#     v varr;
gaussdb-#     flag BOOL;
gaussdb-# BEGIN
gaussdb$#     flag := v.EXISTS(1);
gaussdb$#     RAISE INFO '%',flag;
gaussdb$#     flag := v.EXISTS(3);
gaussdb$#     RAISE INFO '%',flag;
gaussdb$#     flag := v.EXISTS(7);
gaussdb$#     RAISE INFO '%',flag;
gaussdb$# END;
gaussdb$# /
INFO:  f
INFO:  f
INFO:  f
ANONYMOUS BLOCK EXECUTE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2.4.2 Collections

2.4.2.1 Use of Collection Types

Before the use of collections, a collection type must be defined.

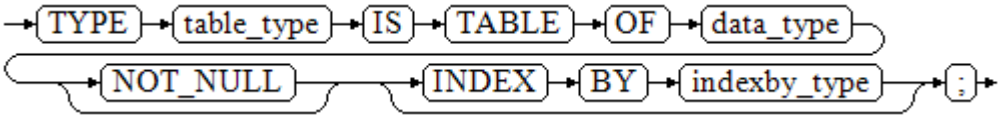
Syntax

Define a collection type immediately after the AS keyword in a stored procedure. The definition method is as follows:

```
TYPE table_type IS TABLE OF data_type [ [NOT_NULL] [INDEX BY indexby_type]];
```

Figure 2-2 shows the syntax diagram.

Figure 2-2 Syntax for defining a collection type



In the preceding information:

- **table_type**: name of the collection type to be defined.
- **data_type**: type of members in the collection to be created.
- **indexby_type**: type of the index set to be created.

Nest-Table Collection Type

- Type definition

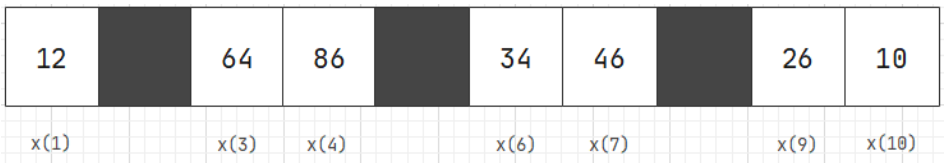
  After a collection type is defined, use **table_type** as the type name to declare the variable.

- Variable declaration and initialization

  You can use a type constructor to initialize the variable during or after declaration. If the variable is not initialized, the value of **var_name** is **NULL**.

  This collection type stores indexes and member values of a specified data type in a variable-length array. You can use the extend function to expand the storage space and use the trim function to release the storage space. For example, if a collection variable *x* has 10 storage units and the member type is int, members are stored as shown in Figure 2-3.

  Figure 2-3 Storage model

  Members x(2), x(5), and x(8) are invalid, but their storage space is still reserved. You can assign values to them later without re-allocating space.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

When a nest-table collection is used as the return value of a function or an input parameter of an autonomous transaction, invalid members will be deleted from the storage space, and the index sets of all members change accordingly.

- Variable assignment

  After the variable is declared and initialized, you can access collection members through an index set or assign values to members. The index set range is [1,*upper*]. The value of *upper* is the size of the current space. If you attempt to access a deleted member, the error message "no data found" will be returned.

- Variable value

  After the variable is declared and initialized, you can access collection members through an index set to obtain the values of member variables. The index set range is [1,*upper*]. The value of *upper* is the size of the current space. If you attempt to access a deleted member, the error message "no data found" will be returned.

Limitations

- GaussDB supports access to collection elements by using parentheses, and it also supports the extend, count, first, last, prior, next, and delete functions. Collection functions multiset union, intersect, except all, and distinct are supported.
- Collection types can be created only in ORA-compatible mode (**sql_compatibility** is set to **ORA**).
- In GaussDB, a nest-table collection does not automatically increment, and an error is reported when you attempt to access the index set out of the specified range.
- Collections can be defined in schemas, anonymous blocks, stored procedures, packages, and user-defined functions, among which the scopes of the collection types vary.
- **NOT NULL** has no function but only takes effect in the syntax.
- When **data_type** is set to a type that can define the length and precision, such as varchar and numeric, you need to enable the **tableof_elem_constraints** parameter (**behavior_compat_options** must be set to **tableof_elem_constraints**) to validate the length of elements in the collection or convert elements to the corresponding precision.
- If **data_type** is of the array type, the element length validation or precision conversion of the array type is also affected by whether the **tableof_elem_constraints** parameter is enabled.
- The value of the collection type converted from the array type does not support element length validation or precision conversion.
- The value of **data_type** can be a basic data type or a record type, collection type, or array type defined in a stored procedure. The ref cursor type is not supported.
- Variables of a collection type cannot be assigned a value of another collection type, even if their member types are the same. For example, *t1* and *t2* are of different collection types defined by **TYPE** *t1* **IS TABLE OF** *int* and **TYPE** *t2* **IS TABLE OF** *int*, respectively. A value of the collection type defined by **TYPE** *t1* **IS TABLE OF** *int* cannot be assigned to *t2*, and vice versa. (This limitation may

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

not take effect when the variables are of the member types because the value assignment logic of variables is affected by the parent type.)

- Only the equal (=) and non-equal operations (<> or !=) between collections are supported. Other relational and arithmetic operations are not supported.
- Use IS [NOT] NULL to compare a collection type with **NULL**. The result of comparison with **NULL** using the equal operator (=) is inaccurate.
- Variables of the collection type can be used as parameters and return values of functions. In this case, the type of the parameters or return values must have been defined in a schema.
- When a nest-table collection is used as the input parameter of a function, an array with elements of the same type can be specified as the input parameter. Multi-dimensional arrays are not supported, and the array index set must start from 1. (This function is outdated and not recommended.) You can run the **SET behavior_compat_options = 'disable_rewrite_nesttable';** command to disable the function.
- Operations on XML data are not supported.
- When creating a table, do not use the collection type or any type containing a collection as a column in the table.
- Constructors of the collection type do not support floating-point numbers or expressions as indexes.
- For a collection type defined in an anonymous block, after **ROLLBACK** is executed or "EXCEPTION" is thrown in an anonymous block, the collection type cannot be used.
- If the GUC parameter **enable_recordtype_check_strict** is set to **on** and the member is of the record type and a column of the record type has the NOT NULL or DEFAULT attribute, an error will be reported during stored procedure or package compilation.
- DML statements cannot be used to access elements of the collection type if the elements are not defined in schemas.
- User-defined collections cannot be pushed down. In pushdown scenarios, variables are used to carry collection elements.
- When you access a collection member using an index, the parser preferentially interprets the collection variable as a table column name. If the table column name is ambiguous, for example, two tables have columns with the same name, an error will occur.

Index-By Table Collection Type

This collection type stores the index set and corresponding member values in a hash table as key-value pairs. All operations on variables of this type are actually operations on the hash table. Users do not need to expand or release storage space, but only need to assign values or delete members. The operations related to the collection type are described as follows:

1. Type definition

   When defining the index-by table collection type, specify both the member type **data_type** and index set type **indexby_type**. The index set type can only be INTEGER or VARCHAR.

2. Variable declaration and initialization

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

After the index-by table collection type is declared, it can be initialized. There are three initialization scenarios: uninitialized, initialized to null, and initialized to specified index and member values. The effect of uninitialized variables is the same as that of initializing variables to null. If a variable is not initialized or is initialized to null, the variable is not **NULL**. You can assign a value to the variable later. Initializing a variable to specified index set and member values will save the specified index set and member values to the variable as key-value pairs.

3. Variable assignment

   Assignment to variables of the index-by table collection type is classified into member assignment and group assignment. Member assignment can assign a value to a member by specifying the index set. If the member does not exist, the value is assigned after the member is inserted. If the member has a value, the member value is updated. Group assignment will clear the original members in the variable and save the new member values. In the group assignment scenario, **NULL** cannot be assigned to variables. Otherwise, an error is reported.

4. Variable value

   You can specify an index set to obtain the member value of the corresponding index in the variable. If the member cannot be found based on the index set, the error message "no data found" is returned.

Limitations

- Index-by table collection types can be created only in ORA-compatible mode (**sql_compatibility** is set to **ORA**).
- Index-by table collection types can be defined in anonymous blocks, stored procedures, user-defined functions, and packages, among which the scopes of the collection types vary. Index-by table collection types cannot be defined in schemas.
- **NOT NULL** has no function but only takes effect in the syntax.
- When **data_type** is set to a type that can define the length and precision, such as varchar and numeric, you need to enable the **tableof_elem_constraints** parameter (**behavior_compat_options** must be set to **tableof_elem_constraints**) to validate the length of elements in the collection or convert elements to the corresponding precision.
- If **data_type** is of the array type, the element length validation or precision conversion of the array type is also affected by whether the **tableof_elem_constraints** parameter is enabled.
- The value of the collection type converted from the array type does not support element length validation or precision conversion.
- The value of **data_type** can be a basic data type or a record type, collection type, or array type defined in a stored procedure. The ref cursor type is not supported.
- The value of **indexby_type** can only be **INTEGER** or **VARCHAR**.
- When **indexby_type** is set to **VARCHAR** and **tableof_elem_constraints** is enabled, the length of the index value is validated when a value is assigned to an index-by table collection type. The validation behavior is not affected by whether **char_coerce_compat** is enabled. If the index length is greater than the defined length, an error is reported. If the **tableof_elem_constraints** parameter is disabled, the index value length is not validated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- An uninitialized variable of the index-by table collection type is not **NULL**.
- **NULL** cannot be assigned to a variable of the index-by table collection type. Otherwise, an error is reported.
- **NULL** and single quotation marks ('') cannot be assigned to a variable of the index-by table collection type.
- Variables of the index-by table collection type cannot be assigned values of another index-by table collection type, even if their member type and index set type are the same. For example, **t1** and **t2** are different collection types defined by running **TYPE t1 IS TABLE OF int index by int;** and **TYPE t2 IS TABLE OF int index by int;**. Variables defined by them cannot be assigned values to each other. (This constraint may not take effect when the variables are used as member types because the assignment logic is affected by their parent type.)
- An index-by table collection type does not support relational and arithmetic operations.
- When a variable of the index-by table collection type is assigned by running **SELECT ... bulk collect INTO**, the index set must be of the integer type. The index set type cannot be varchar.
- Variables of the index-by table collection type can be used as the parameters and return values of functions. In this case, the type of the parameters or return values must be a collection type defined in the package.
- When an index-by table collection is used as the input parameter of a function, an array with elements of the same type can be passed as the input parameter. Multi-dimensional arrays are not supported, and the index type cannot be VARCHAR. (This function is outdated and not recommended. You can run the **SET behavior_compat_options = 'disable_rewrite_nesttable';** command to disable the function.)
- Currently, the type constructor supports only the collection type and the maximum number of parameters is the same as that of user-defined function parameters. For the index-by table collection type, the index value can only be a constant when the constructor is used.
- Operations on XML data are not supported.
- When creating a table, do not use the collection type or any type containing a collection as a column in the table.
- Constructors of the collection type do not support floating-point numbers or expressions as indexes.
- For a collection type defined in an anonymous block, after **ROLLBACK** is executed or "EXCEPTION" is thrown in an anonymous block, the collection type cannot be used.
- If the GUC parameter **enable_recordtype_check_strict** is set to **on** and the member is of the record type and a column of the record type has the NOT NULL or DEFAULT attribute, an error is reported during stored procedure or package compilation.
- DML statements cannot be used to access elements of the collection type if the elements are not defined in schemas.
- User-defined collections cannot be pushed down. In pushdown scenarios, variables are used to carry collection elements.
- When you access a collection member using an index, the parser preferentially interprets the collection variable as a table column name. If the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

table column name is ambiguous, for example, two tables have columns with the same name, an error will occur.

Examples

Example 1: nest-table collection type

```
-- Perform operations on a collection in a stored procedure.
gaussdb=# CREATE OR REPLACE PROCEDURE table_proc AS
DECLARE
    TYPE TABLE_INTEGER IS TABLE OF INTEGER;-- Define a collection type.
    TABLEINT TABLE_INTEGER := TABLE_INTEGER();  -- Declare the variable of the collection type.
BEGIN
    TABLEINT.extend(10);
    FOR I IN 1..10 LOOP
        TABLEINT(I) := I;
    END LOOP;
    DBE_OUTPUT.PRINT_LINE(TABLEINT.COUNT);
    DBE_OUTPUT.PRINT_LINE(TABLEINT(1));
    DBE_OUTPUT.PRINT_LINE(TABLEINT(10));
END;
/
CREATE PROCEDURE
-- Call the stored procedure.
gaussdb=# CALL table_proc();
10
1
10
 table_proc
------------


(1 row)

-- Drop the stored procedure.
gaussdb=# DROP PROCEDURE table_proc;
DROP PROCEDURE
-- Perform operations on a nest-table collection in the stored procedure.
gaussdb=# CREATE OR REPLACE PROCEDURE nest_table_proc AS
DECLARE
    TYPE TABLE_INTEGER IS TABLE OF INTEGER;-- Define a collection type.
    TYPE NEST_TABLE_INTEGER IS TABLE OF TABLE_INTEGER;-- Define a collection type.
    NEST_TABLE_VAR NEST_TABLE_INTEGER := NEST_TABLE_INTEGER(); -- Declare a variable of the nest-table collection type.
BEGIN
    NEST_TABLE_VAR.extend(10);
    FOR I IN 1..10 LOOP
        NEST_TABLE_VAR(I) := TABLE_INTEGER();
        NEST_TABLE_VAR(I).extend(10);
        NEST_TABLE_VAR(I)(I) := I;
    END LOOP;
    DBE_OUTPUT.PRINT_LINE(NEST_TABLE_VAR.COUNT);
    DBE_OUTPUT.PRINT_LINE(NEST_TABLE_VAR(1)(1));
    DBE_OUTPUT.PRINT_LINE(NEST_TABLE_VAR(10)(10));
END;
/
CREATE PROCEDURE
-- Call the stored procedure.
gaussdb=# CALL nest_table_proc();
10
1
10
 nest_table_proc
-----------------


(1 row)

-- Drop the stored procedure.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# DROP PROCEDURE nest_table_proc;
DROP PROCEDURE
```

Example 2: index-by table collection type

```
-- Perform operations on an index-by table collection in a stored procedure.
gaussdb=# CREATE OR REPLACE PROCEDURE index_table_proc AS
DECLARE
    TYPE TABLE_INTEGER IS TABLE OF INTEGER INDEX BY INTEGER; -- Define a collection type.
    TYPE TABLE_VARCHAR IS TABLE OF INTEGER INDEX BY VARCHAR; -- Define a collection type.
    TABLEINT_01 TABLE_INTEGER;                              -- Declare a variable of the collection type, which is
not initialized.
    TABLEINT_02 TABLE_INTEGER := TABLE_INTEGER();           -- Declare a variable of the collection type.
The initial value is null.
    TABLEINT_03 TABLE_INTEGER := TABLE_INTEGER(2=>3,3=>4);   -- Declare a variable of the collection
type and initialize it to the specified value.
    RES INTEGER;
BEGIN
    FOR I IN 1..10 LOOP
        TABLEINT_01(I) := I;     -- Assign values to members.
        TABLEINT_02(I) := I + 1; -- Assign values to members.
    END LOOP;
    TABLEINT_01 := TABLEINT_02;  -- Group assignment
    RES := TABLEINT_03(2);       -- Return values.
    DBE_OUTPUT.PRINT_LINE(RES);
    DBE_OUTPUT.PRINT_LINE(TABLEINT_01(1));
    DBE_OUTPUT.PRINT_LINE(TABLEINT_01(10));
END;
/
CREATE PROCEDURE
-- Call the stored procedure.
gaussdb=# CALL index_table_proc();
3
2
11
 index_table_proc
------------------

(1 row)

-- Drop the stored procedure.
gaussdb=# DROP PROCEDURE index_table_proc;
DROP PROCEDURE
-- Perform operations on a nest-table collection in the stored procedure.
gaussdb=# CREATE OR REPLACE PROCEDURE nest_table_proc AS
DECLARE
    TYPE TABLE_INTEGER IS TABLE OF INTEGER INDEX BY INTEGER; -- Define a collection type.
    TYPE NEST_TABLE_INTEGER IS TABLE OF TABLE_INTEGER INDEX BY INTEGER;-- Define a collection
type.
    NEST_TABLE_VAR NEST_TABLE_INTEGER;                      -- Declare variables of the nest-table
collection type.
BEGIN
    FOR I IN 1..10 LOOP
        NEST_TABLE_VAR(I)(I) := I;
    END LOOP;
    DBE_OUTPUT.PRINT_LINE(NEST_TABLE_VAR.COUNT);
    DBE_OUTPUT.PRINT_LINE(NEST_TABLE_VAR(1)(1));
    DBE_OUTPUT.PRINT_LINE(NEST_TABLE_VAR(10)(10));
END;
/
CREATE PROCEDURE
-- Call the stored procedure.
gaussdb=# CALL nest_table_proc();
10
1
10
 nest_table_proc
-----------------

(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Drop the stored procedure.
gaussdb=# DROP PROCEDURE nest_table_proc;
DROP PROCEDURE
```

2.4.2.2 Collection Functions

Collection Operators

- =

  Parameter: nesttable type.

  Return value: **TRUE** or **FALSE**, of the Boolean type.

  Description: Checks whether two collections are equal.

  Example:

```
gaussdb=# DECLARE
 TYPE nest IS TABLE OF INT
 a nest := nest(1,2);
 b nest := nest(1,2);
 flag BOOL;
 BEGIN
    flag := a = b;
    RAISE INFO '%', flag;
END;
 /
INFO:  t
ANONYMOUS BLOCK EXECUTE
```

- <>

  Parameter: nesttable type.

  Return value: **TRUE** or **FALSE**, of the Boolean type.

  Description: Checks whether two collections are not equal.

  Example:

```
gaussdb=# DECLARE
  TYPE nest IS TABLE OF INT;
  a nest := nest(1,2);
  b nest := nest(1,2);
  flag BOOL;
 BEGIN
   flag := a <> b;
   RAISE INFO '%', flag;
END;
/
INFO:  f
ANONYMOUS BLOCK EXECUTE
```

- IS [NOT] NULL

  Parameter: nesttable type.

  Return value: **TRUE** or **FALSE**, of the Boolean type.

  Description: IS NULL determines whether a collection is NULL, and IS NOT NULL determines whether a collection is not NULL.

  Example:

```
gaussdb=# DECLARE
   TYPE nest IS TABLE OF INT;
   a nest;
   b nest := NULL;
   c nest := nest(1,2);
   flag BOOL;
BEGIN
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
   flag := a IS NULL;
   RAISE INFO '%', flag;
   flag := b IS NULL;
   RAISE INFO '%', flag;
   flag := c IS NULL;
   RAISE INFO '%', flag;
 END;
 /
INFO:  t
INFO:  t
INFO:  f
ANONYMOUS BLOCK EXECUTE

gaussdb=# DECLARE
   TYPE nest IS TABLE OF INT;
   a nest;
   b nest := NULL;
   c nest := nest(1,2);
   flag BOOL;
BEGIN
   flag := a IS NOT NULL;
   RAISE INFO '%', flag;
   flag := b IS NOT NULL;
   RAISE INFO '%', flag;
   flag := c IS NOT NULL;
   RAISE INFO '%', flag;
 END;
 /
INFO:  f
INFO:  f
INFO:  t
ANONYMOUS BLOCK EXECUTE
```

- ^=

  Parameter: nesttable type.

  Return value: **TRUE** or **FALSE**, of the Boolean type.

  Description: Checks whether two collections are not equal.

  Example:

```
gaussdb=# DECLARE
   TYPE nest IS TABLE OF INT;
   a nest := nest(1,2);
   b nest := nest(1,2);
   flag BOOL;
 BEGIN
   flag := a ^= b;
   RAISE INFO '%', flag;
 END;
 /
INFO:  f
ANONYMOUS BLOCK EXECUTE
```

- [NOT] IN

  Parameter: nesttable type.

  Return value: **TRUE** or **FALSE**, of the Boolean type.

  Description: Determines whether a collection is in the collection list.

  Example:

```
gaussdb=# DECLARE
   TYPE tab IS TABLE OF VARCHAR(10);
   tmp1 tab := tab('a', 'b');
   tmp2 tab := tab('a', 'b');
   flag BOOL;
 BEGIN
   flag := tmp2 in (tmp1);
   RAISE INFO '%', flag;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 END;
 /
INFO:  t
ANONYMOUS BLOCK EXECUTE
```

MULTISET

- MULTISET UNION [ALL | DISTINCT]

  Parameter: nesttable type.

  Return value: nesttable type.

  Description: Union of two collection variables. The default value **ALL** indicates that duplicate elements are not removed, and **DISTINCT** indicates that duplicate elements are removed.

  Example 1: Calculate the union of two collection variables and do not remove duplicate elements. That is, MULTISET UNION ALL.

  ```
  gaussdb=# DECLARE
    TYPE nest IS TABLE OF INT;
    a nest := nest(1,2);
    b nest := nest(2,3);
  BEGIN
    a := a MULTISET UNION ALL b;
    RAISE INFO '%', a;
  end;
  /
  INFO:  {1,2,2,3}
  ANONYMOUS BLOCK EXECUTE
  ```

  Example 2: Calculate the union of two collection variables and remove duplicate elements. That is, MULTISET UNION DISTINCT.

  ```
  gaussdb=# DECLARE
     TYPE nest IS TABLE OF INT;
     a nest := nest(1,2);
     b nest := nest(2,3);
  BEGIN
     a := a MULTISET UNION DISTINCT b;
     RAISE INFO '%', a;
  END;
  /
  INFO:  {1,2,3}
  ANONYMOUS BLOCK EXECUTE
  ```

- MULTISET EXCEPT [ALL | DISTINCT]

  Parameter: nesttable type.

  Return value: nesttable type.

  Description: Difference of two collection variables. Take A MULTISET EXCEPT B as an example. **ALL** indicates that elements that both A and B have in A are removed and returned. The number of elements to be removed is calculated. For a specific element, if A appears *m* times and B appears *n* times (m > n), the number of elements removed from A is *m* minus *n*. If m ≤ n, *m* elements are removed from A. **DISTINCT** indicates that elements that both A and B have in A are removed and returned. The number of elements to be removed is not calculated. If specific elements appear in A and B, all such elements in A are removed. **ALL** is the default value.

  Example 1: Calculate a difference set of two collection variables, remove elements that both A and B have in A, and return the result. The number of elements to be removed is calculated. That is, MULTISET EXCEPT ALL.

  ```
  gaussdb=# DECLARE
    TYPE nest IS TABLE OF INT;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  a nest := nest(1,2,2);
  b nest := nest(2,3);
 BEGIN
  a := a MULTISET EXCEPT ALL b;
  RAISE INFO '%', a;
 END;
 /
INFO:  {1,2}
ANONYMOUS BLOCK EXECUTE
```

Example 2: Calculate a difference set of two collection variables, remove elements that both A and B have in A, and return the result. The number of elements to be removed is not calculated. That is, MULTISET EXCEPT DISTINCT.

```
gaussdb=# DECLARE
  TYPE nest IS TABLE OF INT;
  a nest := nest(1,2,2);
  b nest := nest(2,3);
 BEGIN
  a := a MULTISET EXCEPT DISTINCT b;
  RAISE INFO '%', a;
 END;
 /
INFO:  {1}
ANONYMOUS BLOCK EXECUTE
```

- MULTISET INTERSECT [ALL | DISTINCT]

  Parameter: nesttable type.

  Return value: nesttable type.

  Description: Intersection of two collection variables. Take A MULTISET INTERSECT B as an example. **ALL** indicates that all duplicate elements (including elements that repeat for multiple times) are retained. **DISTINCT** indicates that the intersection of distinct values is returned and duplicate elements are removed. **ALL** is the default value.

  Example 1: Calculate the intersection of two collection variables and do not remove duplicate elements. That is, MULTISET INTERSECT ALL.

```
gaussdb=# DECLARE
  TYPE nest IS TABLE OF INT;
  a nest := nest(1,2,2);
  b nest := nest(2,2,3);
 BEGIN
  a := a MULTISET INTERSECT ALL b;
  RAISE INFO '%', a;
 END;
 /
INFO:  {2,2}
ANONYMOUS BLOCK EXECUTE
```

  Example 2: Calculate the intersection of two collection variables and remove duplicate elements. That is, MULTISET INTERSECT DISTINCT.

```
gaussdb=# DECLARE
  TYPE nest IS TABLE OF INT;
  a nest := nest(1,2,2);
  b nest := nest(2,2,3);
 BEGIN
  a := a MULTISET INTERSECT DISTINCT b;
  RAISE INFO '%', a;
 END;
 /
INFO:  {2}
ANONYMOUS BLOCK EXECUTE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Functions of the Collection Type

NOTE

- In the following function definition description, the content in [] is optional. For example, count[()] can be written as count or count().
- In an inner expression, functions of the collection type cannot be called in nesting mode.
- When types are nested and the collection type is used as the internal type, the extend, delete, and trim functions without parameters can be called extend, delete, and trim, but not extend(), delete(), or trim().

- exists(idx)

  Parameter: *idx* is of the INT4 or VARCHAR type.

  Return value: **TRUE** or **FALSE**, of the Boolean type.

  Description: Checks whether a valid element exists in a specified position.

  Example:

```
gaussdb=# DECLARE
  TYPE nest IS TABLE OF VARCHAR2;
  a nest := nest('happy','?');
  flag BOOL;
 BEGIN
  flag := a.EXISTS(1);
  RAISE INFO '%', flag;
  flag := a.EXISTS(10);
  RAISE INFO '%', flag;
 END;
 /
INFO:  t
INFO:  f
ANONYMOUS BLOCK EXECUTE
gaussdb=# DECLARE
  TYPE nest IS TABLE OF VARCHAR2 INDEX BY VARCHAR2;
  a nest;
  flag BOOL;
 BEGIN
  a('1') := 'Be';
  a('2') := 'happy';
  a('3') := '.';
  flag := a.EXISTS('1');
  RAISE INFO '%', flag;
  flag := a.EXISTS('ddd');
  RAISE INFO '%', flag;
 END;
 /
INFO:  t
INFO:  f
ANONYMOUS BLOCK EXECUTE
```

- extend[(count[, idx])]

  Parameter: *idx* and *count* are of the INT4 type. If no parameter is passed, one element is extended and the value is **NULL**.

  Return value: No value is returned.

  Description: Supports only variables of the nest-table type. One or *count* elements are extended at the end of the nest-table variable. If the element indexed by *idx* exists, *count* elements are copied to the end of the variable.

  Example 1: A nest-table variable extends one element and the extended element value is **NULL**.

```
gaussdb=# DECLARE
  TYPE nest IS TABLE OF INT;
  a nest := nest(1);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 BEGIN
   RAISE INFO '%', a;
   a.EXTEND;
   RAISE INFO '%', a;
   a.EXTEND;
   RAISE INFO '%', a;
 END;
 /
INFO:  {1}
INFO:  {1,NULL}
INFO:  {1,NULL,NULL}
ANONYMOUS BLOCK EXECUTE
```

Example 2: A nest-table variable extends *n* elements and the extended element value is **NULL**.

```
gaussdb=# DECLARE
   TYPE nest IS TABLE OF INT;
   a nest := nest(1);
 BEGIN
   RAISE INFO '%', a;
   a.EXTEND(2);
   RAISE INFO '%', a;
 END;
 /
INFO:  {1}
INFO:  {1,NULL,NULL}
ANONYMOUS BLOCK EXECUTE
```

Example 3: A nest-table variable extends *n* elements and the extended element value is the value of the element with the specified index.

```
gaussdb=# DECLARE
   TYPE nest IS TABLE OF INT;
   a nest := nest(9);
 BEGIN
   RAISE INFO '%', a;
   a.EXTEND(2,1);
   RAISE INFO '%', a;
 END;
 /
INFO:  {9}
INFO:  {9,9,9}
ANONYMOUS BLOCK EXECUTE
```

- delete[(idx1[, idx2])]

  Parameter: *idx1* and *idx2* are of the INT4 or VARCHAR2 type.

  Return value: No value is returned.

  Description: If there is no parameter, the variable of the nest-table type deletes all elements of the collection type and the space. If the space is used later, the space needs to be expanded. If there is no parameter, the variable of the index-by table type deletes all elements. If there is one parameter, the element at the specified position is deleted (the space is not deleted). If there are two parameters, all elements in the index range are deleted (the space is not deleted).

  Limitation: For the nest-table collection type, if an element is deleted but it is not removed from the storage space, the space is invalid and can be assigned a value. Note: When a collection is used as the return value of a function or an input parameter of an autonomous transaction, invalid members will be deleted from the storage space, and the index set of all members changes accordingly.

  Example 1: Delete all elements and spaces of the collection type from the variable of the nest-table collection type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# DECLARE
  TYPE nest IS TABLE OF INT;
  a nest := nest(1,2,3,4,5);
 BEGIN
  RAISE INFO '%', a;
  a.DELETE();
  RAISE INFO '%', a;
 END;
 /
INFO:  {1,2,3,4,5}
INFO:  {}
ANONYMOUS BLOCK EXECUTE
```

Example 2: Delete an element at a specified position from the variable of the nest-table collection type.

```
gaussdb=# DECLARE
  TYPE nest IS TABLE OF INT;
  a nest := nest(1,2,3,4,5);
 BEGIN
  RAISE INFO '%', a;
  a.DELETE(3);
  RAISE INFO '%', a;
  a(3) := 3;
  RAISE INFO '%', a;
 END;
 /
INFO:  {1,2,3,4,5}
INFO:  {1,2,4,5}
INFO:  {1,2,3,4,5}
ANONYMOUS BLOCK EXECUTE
```

Example 3: Delete elements in a specified interval from the variable of the nest-table collection type.

```
gaussdb=# DECLARE
  TYPE nest IS TABLE OF INT;
  a nest := nest(1,2,3,4,5);
 BEGIN
  RAISE INFO '%', a;
  a.DELETE(2,4);
  RAISE INFO '%', a(1);
  RAISE INFO '%', a(5);
  RAISE INFO '%', a;
 END;
 /
INFO:  {1,2,3,4,5}
INFO:  1
INFO:  5
INFO:  {1,5}
ANONYMOUS BLOCK EXECUTE
```

Example 4: Delete all elements and spaces of the collection type from the variable of the index-by table collection type.

```
gaussdb=# DECLARE
  TYPE t1 IS TABLE OF INT INDEX BY VARCHAR;
  v t1 := t1('a' => 1, 'b' => 2, 'c' => 3, 'd' => 4);
 BEGIN
  v.DELETE();
  RAISE INFO '%', v.COUNT();
 END;
 /
INFO:  0
ANONYMOUS BLOCK EXECUTE
```

Example 5: Delete an element at a specified position from the variable of the index-by table collection type.

```
gaussdb=# declare
  type t1 is table of int index by varchar;
  v t1 := t1('a' => 1, 'b' => 2, 'c' => 3, 'd' => 4);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 begin
   raise info '%', v('a');
   v.delete('a');
   raise info '%', v('a');
 end;
 /
INFO:  1
ERROR:  no data found
CONTEXT:  PL/pgSQL function inline_code_block line 6 at RAISE
```

Example 6: Delete elements in a specified interval from the variable of the index-by table collection type.

```
gaussdb=# declare
   type t1 is table of int index by varchar;
   v t1 := t1('a' => 1, 'b' => 2, 'c' => 3, 'd' => 4);
 begin
   raise info '%', v('b');
   v.delete('a', 'c');
   raise info '%', v('b');
 end;
 /
INFO:  2
ERROR:  no data found
CONTEXT:  PL/pgSQL function inline_code_block line 6 at RAISE
```

- trim[(n)]

  Parameter: *n* is of the INT4 type.

  Return value: No value is returned.

  Description: Supports only variables of the nest-table type. If no parameter is specified, the last element space is deleted. If the input parameter is valid, the specified number of element spaces are deleted.

  Example:

```
gaussdb=# DECLARE
   TYPE nest IS TABLE OF INT;
   aa nest := nest(11,22,33,44,55);
 BEGIN
   RAISE INFO 'aa:%' ,aa;
   aa.TRIM;  -- No parameter
   RAISE INFO 'aa:%' ,aa;
   aa.TRIM();  -- No parameter
   RAISE INFO 'aa:%' ,aa;
   aa.TRIM(2);  -- Valid parameter
   RAISE INFO 'aa:%' ,aa;
   aa.TRIM(2);  -- The collection element space is less than 2. An error is reported when the parameter
is invalid.
 END;
 /
INFO:  aa:{11,22,33,44,55}
INFO:  aa:{11,22,33,44}
INFO:  aa:{11,22,33}
INFO:  aa:{11}
ERROR:  Subscript beyond count
CONTEXT:  PL/pgSQL function inline_code_block line 11 at assignment
```

- count[()]

  Parameter: none.

  Return value: INT type.

  Description: Returns the number of valid elements in a collection.

  Example 1: Use the count function for the variable of the nest-table collection type.

```
gaussdb=# DECLARE
 TYPE nest IS TABLE OF INT;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 aa nest:=nest(11,22,33,44,55);
 BEGIN
 RAISE INFO 'count:%' ,aa.COUNT;
 aa.DELETE(3);  -- Delete an element. The element whose index is 3 is invalid.
 RAISE INFO 'count:%' ,aa.COUNT();
 END;
 /
INFO:  count:5
INFO:  count:4
ANONYMOUS BLOCK EXECUTE
```

Example 2: Use the count function for the variable of the index-by table collection type.

```
gaussdb=# DECLARE
 TYPE t1 IS TABLE OF INT INDEX BY INT;
 aa t1;
 BEGIN
 aa(1) := 111;
 aa(2) := 222;
 aa(3) := 333;
 RAISE INFO 'count:%' ,aa.COUNT();
 END;
 /
INFO:  count:3
ANONYMOUS BLOCK EXECUTE

gaussdb=# DECLARE
 TYPE t1 IS TABLE OF INT INDEX BY INT;
 aa t1;
 BEGIN
 aa(1) := 111;
 aa(2) := 222;
 aa(3) := 333;
 RAISE INFO 'count:%' ,aa.COUNT();
 END;
 /
INFO:  count:3
ANONYMOUS BLOCK EXECUTE
```

- first[()]

  Parameter: none.

  Return value: INT or VARCHAR type.

  Description: Returns the index of the first valid element in a collection.

  Example 1: Use the first function for the variable of the nest-table collection type.

```
gaussdb=# DECLARE
 TYPE nest IS TABLE OF INT;
 aa nest:=nest(11,22,33,44,55);
 BEGIN
 RAISE INFO 'first:%' ,aa.FIRST();
 aa.DELETE(1);
 RAISE INFO 'first:%' ,aa.FIRST;  -- The element whose index is 1 is invalid. The index of the first valid
element is 2.
 END;
 /
INFO:  first:1
INFO:  first:2
ANONYMOUS BLOCK EXECUTE
```

  Example 2: Use the first function for the variable of the index-by table collection type.

```
gaussdb=# DECLARE
 TYPE t1 IS TABLE OF INT INDEX BY INT;
 aa t1;
 BEGIN
 aa(3) := 111;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 aa(2) := 222;
 aa(1) := 333;
 RAISE INFO 'first:%' ,aa.FIRST;
 END;
 /
INFO:  first:1
ANONYMOUS BLOCK EXECUTE

gaussdb=# DECLARE
 TYPE t1 IS TABLE OF INT INDEX BY VARCHAR;
 aa t1;
 BEGIN
 aa('aaa') := 111;
 aa('bbb') := 222;
 aa('ccc') := 333;
 RAISE INFO 'first:%' ,aa.FIRST;
 END;
 /
INFO:  first:aaa
ANONYMOUS BLOCK EXECUTE
```

- last[()]

  Parameter: none.

  Return value: INT or VARCHAR type.

  Description: Returns the index of the last valid element in a collection.

  Example 1: Use the last function for the variable of the nest-table collection type.

```
gaussdb=# DECLARE
 TYPE nest IS TABLE OF INT;
 aa nest:=nest(11,22,33,44,55);
 BEGIN
 RAISE INFO 'last:%' ,aa.LAST;
 aa.DELETE(5);
 RAISE INFO 'last:%' ,aa.LAST();  -- The element whose index is 5 is invalid. The index of the last valid
element is 4.
 END;
 /
INFO:  last:5
INFO:  last:4
ANONYMOUS BLOCK EXECUTE
```

  Example 2: Use the last function for the variable of the index-by table collection type.

```
gaussdb=# DECLARE
 TYPE t1 IS TABLE OF INT INDEX BY VARCHAR;
 aa t1;
 BEGIN
 aa(3) := 111;
 aa(2) := 222;
 aa(1) := 333;
 RAISE INFO 'last:%' ,aa.LAST();
 END;
 /
INFO:  last:3
ANONYMOUS BLOCK EXECUTE

gaussdb=# DECLARE
 TYPE t1 IS TABLE OF INT INDEX BY VARCHAR;
 aa t1;
 BEGIN
 aa(3) := 111;
 aa(2) := 222;
 aa(1) := 333;
 RAISE INFO 'last:%' ,aa.LAST();
 END;
 /
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
INFO:  last:ccc
ANONYMOUS BLOCK EXECUTE
```

- prior(idx)

  Parameter: *idx* is of the INT or VARCHAR type.

  Return value: INT or VARCHAR type.

  Description: Returns the index of the valid element before the given index in a collection. If the index does not exist, the last element in the collection is returned.

  Example 1: Use the prior function for the variable of the nest-table collection type.

```
gaussdb=# DECLARE
 TYPE nest IS TABLE OF INT;
 aa nest:=nest(11,22,33,44,55);
 BEGIN
 RAISE INFO 'prior:%' ,aa.PRIOR(3);
 END;
 /
INFO:  prior:2
ANONYMOUS BLOCK EXECUTE
```

  Example 2: Use the prior function for the variable of the index-by table collection type.

```
gaussdb=# DECLARE
 TYPE t1 IS TABLE OF INT INDEX BY VARCHAR;
 aa t1;
 BEGIN
 aa('ccc') := 111;
 aa('bbb') := 222;
 aa('aaa') := 333;
 RAISE INFO 'prior:%' ,aa.PRIOR('bbb');
 RAISE INFO 'prior:%' ,aa.PRIOR('ddd');
 END;
 /
INFO:  prior:aaa
INFO:  prior:ccc
ANONYMOUS BLOCK EXECUTE
```

- next(idx)

  Parameter: *idx* is of the INT or VARCHAR type. If the index does not exist, the first element in the collection is returned.

  Return value: INT or VARCHAR type.

  Description: Returns the index of a valid element following the given index in a collection.

  Example 1: Use the next function for the variable of the nest-table collection type.

```
gaussdb=# DECLARE
 TYPE nest IS TABLE OF INT;
 aa nest:=nest(11,22,33,44,55);
 BEGIN
 RAISE INFO 'next:%' ,aa.NEXT(3);
 END;
 /
INFO:  next:4
ANONYMOUS BLOCK EXECUTE
```

  Example 2: Use the next function for the variable of the index-by table collection type.

```
gaussdb=# DECLARE
 TYPE t1 IS TABLE OF INT INDEX BY INT;
 aa t1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
BEGIN
 aa(3) := 111;
 aa(2) := 222;
 aa(1) := 333;
 RAISE INFO 'next:%' ,aa.NEXT(2);
 RAISE INFO 'next:%' ,aa.NEXT(-999);
 END;
 /
INFO:  next:3
INFO:  next:1
ANONYMOUS BLOCK EXECUTE
```

- limit

  Parameter: none.

  Return value: **NULL**.

  Description: Applies only to variables of the nest-table type. The return value is **NULL**.

  Example:

```
gaussdb=# DECLARE
 TYPE nest IS TABLE OF INT;
 aa nest:=nest(11,22,33,44,55);
 BEGIN
 RAISE INFO 'limit:%' ,aa.LIMIT;
 END;
 /
INFO:  limit:<NULL>
ANONYMOUS BLOCK EXECUTE
```

Collection-related Functions

- unnest_table (anynesttable) or unnest (anynesttable)

  Parameter: any nest-table collection type.

  Description: Returns the result set of all valid elements in a given nest-table. Multiple rows of data are returned.

  Return type: setof anyelement.

  Example:

```
If the element type of the collection is record and any element is NULL, the element is not returned. Instead, a non-null record value is returned. All columns of the record are NULL.
gaussdb=# CREATE OR REPLACE p1()
gaussdb-# AS
gaussdb$#     TYPE t1 IS TABLE OF INT;
gaussdb$#     v2 t1 := t1(NULL, 2, 3, 4, NULL);
gaussdb$#     tmp INT;
gaussdb$#     CURSOR c1 IS SELECT * FROM UNNEST_TABLE(v2);
gaussdb$# BEGIN
gaussdb$# OPEN c1;
gaussdb$# FOR i IN 1 .. v2.COUNT LOOP
gaussdb$#     FETCH c1 INTO tmp;
gaussdb$#     IF tmp IS NULL THEN
gaussdb$#         DBE_OUTPUT.PRINT_LINE(i || ': is null');
gaussdb$#     ELSE
gaussdb$#         DBE_OUTPUT.PRINT_LINE(i || ': ' || tmp);
gaussdb$#     END IF;
gaussdb$# END LOOP;
gaussdb$# CLOSE c1;
gaussdb$# END;
gaussdb$# /
CREATE PROCEDURE
gaussdb=# CALL p1();
1: is null
2: 2
3: 3
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
4: 4
5: is null
 p1
----

(1 row)
gaussdb=# DROP PROCEDURE IS EXISTS p1();
DROP PROCEDURE

-- Example: nested record types in a nest-table
gaussdb=# CREATE OR REPLACE PROCEDURE p1() IS
gaussdb$#   TYPE rec IS RECORD(c1 INT, c2 INT);
gaussdb$#   TYPE t1 IS TABLE OF rec;
gaussdb$#   v t1 := t1(rec(1, 1), rec(2, NULL), rec(NULL, NULL), NULL);
gaussdb$#   v2 t1 := t1();
gaussdb$#   CURSOR cur IS SELECT * FROM UNNEST(v);
gaussdb$# BEGIN
gaussdb$#   v2.EXTEND(v.COUNT);
gaussdb$#   OPEN cur;
gaussdb$#   FOR i IN 1 .. v.COUNT LOOP
gaussdb$#     FETCH cur INTO v2(i);
gaussdb$#     RAISE INFO '%', v2(i);
gaussdb$#   END LOOP;
gaussdb$#   CLOSE cur;
gaussdb$# END
gaussdb$# /
CREATE PROCEDURE
gaussdb=# CALL p1();
INFO:  (1,1)
INFO:  (2,)
INFO:  (,)
INFO:  (,)
 p1
----

(1 row)
gaussdb=# DROP PROCEDURE IF EXISTS p1();
DROP PROCEDURE
```

- unnest_table (anyindexbytable) or unnest (anyindexbytable)

  Parameter: any index-by table collection type.

  Description: Returns the result set of all elements sorted by index in a given index-by table. Multiple rows of data are returned.

  Return type: setof anyelement.

  Limitation: Only the INDEX BY INT type is supported.

  Example:

```
-- If the element type of the collection is record and any element is NULL, the element is not returned. Instead, a non-null record value is returned. All columns of the record are NULL.
gaussdb=# CREATE OR REPLACE PROCEDURE p1()
gaussdb-# AS
gaussdb$#     TYPE t1 IS TABLE OF INT INDEX BY INT;
gaussdb$#     v2 t1 := t1(1=>1, -10=>(-10), 6=>6, 4=>NULL);
gaussdb$#     tmp INT;
gaussdb$#     CURSOR c1 IS SELECT * FROM UNNEST_TABLE(v2);
gaussdb$# BEGIN
gaussdb$# OPEN c1;
gaussdb$# FOR i IN 1 .. v2.COUNT LOOP
gaussdb$#     FETCH c1 INTO tmp;
gaussdb$#     IF tmp IS NULL THEN
gaussdb$#         DBE_OUTPUT.PRINT_LINE(i || ': is null');
gaussdb$#     ELSE
gaussdb$#         DBE_OUTPUT.PRINT_LINE(i || ': ' || tmp);
gaussdb$#     END IF;
gaussdb$# END LOOP;
gaussdb$# CLOSE c1;
gaussdb$# END;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb$# /
CREATE PROCEDURE
gaussdb=# CALL p1();
1: -10
2: 1
3: is null
4: 6
 p1
----

(1 row)
gaussdb=# DROP PROCEDURE IF EXISTS p1();
DROP PROCEDURE

-- Example: nested record types in an index-by table
gaussdb=# CREATE OR REPLACE PROCEDURE p1() IS
gaussdb$#   TYPE rec IS RECORD(c1 INT, c2 INT);
gaussdb$#   TYPE t1 IS TABLE OF rec INDEX BY INT;
gaussdb$#   v t1 := t1(1 => rec(1, 1), 2 => rec(2, NULL), 3 => rec(NULL, NULL), 4 => NULL);
gaussdb$#   v2 t1 := t1();
gaussdb$#   CURSOR cur IS SELECT * FROM UNNEST(v);
gaussdb$# BEGIN
gaussdb$#   OPEN cur;
gaussdb$#   FOR i IN 1 .. v.COUNT LOOP
gaussdb$#     FETCH cur INTO v2(i);
gaussdb$#     RAISE INFO '%', v2(i);
gaussdb$#   END LOOP;
gaussdb$#   CLOSE cur;
gaussdb$# END;
gaussdb$# /
CREATE PROCEDURE
gaussdb=# CALL p1();
INFO:  (1,1)
INFO:  (2,)
INFO:  (,)
INFO:  (,)
 p1
----

(1 row)
gaussdb=# drop procedure if exists p1();
DROP PROCEDURE
```

2.4.3 Records

Record Variables

Define a record type and use this type to declare a variable, which is a record variable.

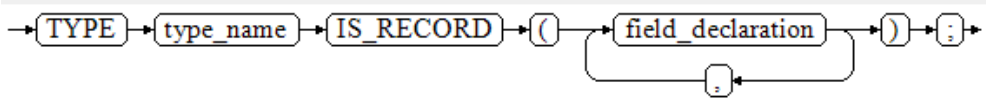
Syntax

The syntax for defining the record type is as follows:

```
TYPE type_name IS_RECORD (field_declaration [,...] );
```

Figure 2-4 shows the syntax diagram.

Figure 2-4 Syntax for defining a record type



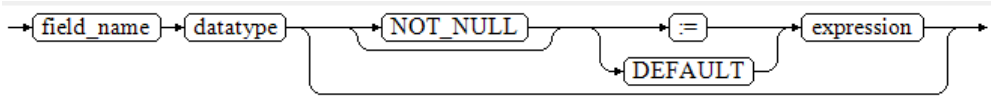
 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

field_declaration is as follows:

```
field_name datatype [ [NOT_NULL] { := | DEFAULT } expression]
```

Figure 2-5 shows the syntax diagram of field_declaration.

Figure 2-5 field_declaration



The syntax is described as follows:

- *type_name*: record type.
- *field_name*: member variables of this type.
- *datatype*: data type of the member variables.
- *expression*: expression for setting a default value.

Limitations

In GaussDB:

- When assigning values to record variables, you can:
  - Declare a record type and define member variables of this type when you declare a function or stored procedure.
  - Assign the value of a record variable to another record variable.
  - Use SELECT INTO or FETCH to assign values to a record variable.
  - Assign the **NULL** value to a record variable.
- The constructor of the record type can use => to assign values. The limitations are as follows:
  - The constructor of the record type can assign values using => only when an ORA-compatible database is used (**sql_compatibility** is set to **'ORA'**).
  - The record type can be defined in a package.
  - The constructor of the record type uses => to assign values only when => is consecutively used to assign values to input parameters, for example, **record_name(elename1 => val1, elename2 => val2, elename3 => val3)** or **record_name(val1, elename2 => val2, elename3 => val3)**. Discontinuous use of => is not supported. For example, **record_name(elename1 => val1, elename2 => val2, val3)** or **record_name(val1, elename2 => val2, val3)**, where the last input parameter is not assigned a value using =>.
- The INSERT and UPDATE statements cannot use a record variable to insert or update data.
- The constructor of the record type cannot be used as the default value of a function or stored procedure parameter.
- Just like a variable declared, a member of the composite type does not have a default value when it is declared.
- For members of the record type, the default value of the inner record type cannot be passed to the outer record type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If the GUC parameter **enable_recordtype_check_strict** is set to **on** and members are of the record type and a column of the record type has the NOT NULL or DEFAULT attribute, an error will be reported during stored procedure or package compilation.
- If the GUC parameter **enable_recordtype_check_strict** is set to **on**, a package contains the record type, and a column of the record type has the NOT NULL or DEFAULT attribute, an error will be reported during creation or compilation.
- If the GUC parameter **enable_recordtype_check_strict** is set to **on**, the NOT NULL attribute of the record type in the stored procedure takes effect.
- After the **proc_outparam_override** option is enabled for the GUC parameter **behavior_compat_options**:
  - Functions that contain the **out** parameter and returns data of the record type can be called using SELECT and CALL in external SQL statements, and can be called using PERFORM and expressions in stored procedures.
  - If the function returns data of the undefined record type, at least one **out** parameter is required. If the function returns data of the defined record type, the **out** parameter is optional. For details, see **Examples**.
  - Functions that contain the **out** output parameter and return data of the composite or record type, ensure that the expected return type of the function is the same as the actual return type.
- In the INSERT INTO *table_name* VALUES *record_var* syntax, *record_var* supports only record variables and record variables nested in a single-layer array, but does not support record variables nested in a multi-layer array.
- Static DDL statements in PL/SQL do not support variables or members of the record type.
- When a record variable is assigned as a whole, values are assigned to columns in the declaration order. If an exception occurs during the assignment of a column, the columns that have been assigned values will not be rolled back.

Examples

```
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE emp_rec(
gaussdb(#     empno NUMERIC(4,0) NOT NULL,
gaussdb(#     ename VARCHAR(10)
gaussdb(# );
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'empno' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# INSERT INTO emp_rec VALUES(111, 'aaa'), (222, 'bbb'), (333, 'ccc');
INSERT 0 3
-- The table is defined as follows:
gaussdb=# \d emp_rec
          Table "public.emp_rec"
 Column |         Type          | Modifiers
--------+-----------------------+-----------
 empno  | numeric(4,0)          | not null
 ename  | character varying(10) |

-- Perform record operations in the function.
gaussdb=# CREATE OR REPLACE FUNCTION regress_record(p_w VARCHAR2) RETURNS VARCHAR2 AS $$
gaussdb$# DECLARE
gaussdb$#     -- Declare a record type.
gaussdb$#     TYPE rec_type IS RECORD (name  VARCHAR2(100), epno INT);
gaussdb$#     employer rec_type;
gaussdb$#     -- Use %type to declare the record type.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb$#     TYPE rec_type1 IS RECORD (name  emp_rec.ename%TYPE, epno INT :=10);
gaussdb$#     employer1 rec_type1;
gaussdb$#     -- Declare a record type with the default value.
gaussdb$#     TYPE rec_type2 IS RECORD (
gaussdb$#         name VARCHAR2 NOT NULL := 'SCOTT',
gaussdb$#         epno INT NOT NULL :=10
gaussdb$#     );
gaussdb$#     employer2 rec_type2;
gaussdb$#     CURSOR C1 IS  SELECT ename,empno FROM emp_rec ORDER BY 1 LIMIT 1;
gaussdb$# BEGIN
gaussdb$#     -- Assign a value to a member record variable.
gaussdb$#     employer.name := 'WARD';
gaussdb$#     employer.epno = 18;
gaussdb$#     RAISE INFO 'employer name: % , epno:%', employer.name, employer.epno;
gaussdb$#
gaussdb$#     -- Assign the value of a record variable to another variable.
gaussdb$#     employer1 := employer;
gaussdb$#     RAISE INFO 'employer1 name: % , epno: %',employer1.name, employer1.epno;
gaussdb$#
gaussdb$#     -- Assign the NULL value to a record variable.
gaussdb$#     employer1 := NULL;
gaussdb$#     RAISE INFO 'employer1 name: % , epno: %',employer1.name, employer1.epno;
gaussdb$#
gaussdb$#     -- Obtain the default value of a record variable.
gaussdb$#     RAISE INFO 'employer2 name: % ,epno: %', employer2.name, employer2.epno;
gaussdb$#
gaussdb$#     -- Use a record variable in the FOR loop.
gaussdb$#     FOR employer IN SELECT ename,empno FROM emp_rec ORDER BY 1 LIMIT 1 LOOP
gaussdb$#         RAISE INFO 'employer name: % , epno: %', employer.name, employer.epno;
gaussdb$#     END LOOP;
gaussdb$#
gaussdb$#     -- Use a record variable in the SELECT INTO statement.
gaussdb$#     SELECT ename,empno  INTO employer2 FROM emp_rec ORDER BY 1 LIMIT 1;
gaussdb$#     RAISE INFO 'employer name: % , epno: %', employer2.name, employer2.epno;
gaussdb$#
gaussdb$#     -- Use a record variable in a cursor.
gaussdb$#     OPEN C1;
gaussdb$#     FETCH C1 INTO employer2;
gaussdb$#     RAISE INFO 'employer name: % , epno: %', employer2.name, employer2.epno;
gaussdb$#     CLOSE C1;
gaussdb$#     RETURN employer.name;
gaussdb$# END; $$
gaussdb-# LANGUAGE plpgsql;
CREATE FUNCTION

-- Call this function.
gaussdb=# CALL regress_record('abc');
INFO:  employer name: WARD , epno:18
INFO:  employer1 name: WARD , epno: 18
INFO:  employer1 name: <NULL> , epno: <NULL>
INFO:  employer2 name: SCOTT ,epno: 10
INFO:  employer name: aaa , epno: 111
INFO:  employer name: aaa , epno: 111
INFO:  employer name: aaa , epno: 111
 regress_record
----------------
 aaa
(1 row)
-- Drop the function.
gaussdb=# DROP FUNCTION regress_record;
DROP FUNCTION
-- Drop the table.
gaussdb=# DROP TABLE emp_rec;
DROP TABLE

-- If the compatibility parameter proc_outparam_override is enabled, the data of the defined record type is returned. The out parameter is not required for the function.
gaussdb=# CREATE TYPE rec_type IS (c1 INT, c2 INT);
CREATE TYPE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SET behavior_compat_options = 'proc_outparam_override';
SET
gaussdb=# CREATE OR REPLACE FUNCTION func(a in INT) RETURN rec_type IS
gaussdb$#     r rec_type;
gaussdb$# BEGIN
gaussdb$#     r.c1:=1;
gaussdb$#     r.c2:=1;
gaussdb$#     RETURN r;
gaussdb$# END;
gaussdb$# /
CREATE FUNCTION
gaussdb=# CALL func(0);
 c1 | c2
----+----
  1 |  1
(1 row)
gaussdb=# DROP FUNCTION func;
DROP FUNCTION
gaussdb=# DROP TYPE rec_type;
DROP TYPE
-- If the compatibility parameter proc_outparam_override is enabled, the data of the undefined record
type is returned and at least one out parameter is required for the function.
gaussdb=# SET behavior_compat_options = 'proc_outparam_override';
SET
gaussdb=# CREATE OR REPLACE FUNCTION func(a OUT INT) RETURN RECORD IS
gaussdb$#     TYPE rc IS RECORD(c1 INT, c2 INT);
gaussdb$#     r rc;
gaussdb$# BEGIN
gaussdb$#     r.c1 := 1;
gaussdb$#     r.c2 := 1;
gaussdb$#     a := 999;
gaussdb$#     RETURN r;
gaussdb$# END;
gaussdb$# /
CREATE FUNCTION
gaussdb=# CALL func(1);
 func  |  a
-------+-----
 (1,1) | 999
(1 row)
gaussdb=# DROP FUNCTION func;
DROP FUNCTION

-- Create an ORA-compatible database and switch to this database.
gaussdb=#  CREATE DATABASE ora_compatible_db DBCOMPATIBILITY 'ORA';
CREATE DATABASE
gaussdb=#  \c ora_compatible_db;
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "ora_compatible_db" as user "omm".
-- Define the record in a package. Use => to assign values to all input parameters of the record constructor.
The execution is successful.
ora_compatible_db=#
CREATE OR REPLACE PACKAGE pa_test1
AS
 PROCEDURE p_display1(a NUMBER, b  VARCHAR);
 TYPE t1 IS RECORD(va INT,vb INT);
 v1 t1;
END pa_test1;
/
ora_compatible_db=#
CREATE OR REPLACE PACKAGE BODY pa_test1
AS
  PROCEDURE p_display1(a NUMBER, b  VARCHAR) AS
 BEGIN
  v1 := t1(va => 1, vb => 2);
  DBE_OUTPUT.PUT_LINE('pkg test1');
 END;
END pa_test1;
/
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ora_compatible_db=# CALL pa_test1.p_display1(b => 'm', a => 1);
 p_display1
------------


(1 row)
-- Define a record in a package. The input parameter values of the record constructor are not consecutively assigned using =>, and an error message is displayed.
ora_compatible_db=#
CREATE OR REPLACE PACKAGE pa_test1
AS
 PROCEDURE p_display1(a NUMBER, b  VARCHAR);
 TYPE t1 IS RECORD(va INT,vb INT);
 v1 t1;
END pa_test1;
/
CREATE OR REPLACE PACKAGE BODY pa_test1
AS
  PROCEDURE p_display1(a NUMBER, b  VARCHAR) AS
 BEGIN
  v1 := t1(va => 1, 2);
  DBE_OUTPUT.PUT_LINE('pkg test1');
 END;
END pa_test1;
/
ora_compatible_db=# CALL pa_test1.p_display1(b => 'm', a => 1);
ERROR:  positional argument cannot follow named argument
LINE 1: SELECT "17347.t1"(va => 1, 2)
                          ^
QUERY:  SELECT "17347.t1"(va => 1, 2)
CONTEXT:  referenced column: 17347.t1
PL/pgSQL function p_display1(numeric,character varying) line 2 at assignment
-- Switch back to the original database and drop the created database.
ora_compatible_db=# \c postgres;
Non-SSL connection (SSL connection is recommended when requiring high-security)
You are now connected to database "postgres" as user "omm".
gaussdb=# DROP DATABASE ora_compatible_db;
DROP DATABASE
```

2.5 DECLARE Syntax

2.5.1 Basic Structure

Structure

A PL/SQL block can contain a sub-block which can be placed in any section. The following describes the architecture of a PL/SQL block:

- Declaration section: declares variables, types, cursors, and local stored procedures and functions used in the PL/SQL block.
  ```
  DECLARE
  ```

  NOTE

  This section is optional if no variables need to be declared.
  - An anonymous block may omit the DECLARE keyword if no variable needs to be declared.
  - For a stored procedure, if the DECLARE keyword is not used, the AS keyword is equivalent to the DECLARE keyword. The AS keyword must be retained in the syntax structure even if there is no explicit variable declaration.
- Execution section: specifies procedure and SQL statements. It is the main section of a program. It is required.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
BEGIN
```

- Exception-handling section: processes errors. It is optional.

```
EXCEPTION
```

- End section: It is required.

```
END;
/
```

NOTICE

You are not allowed to use consecutive tabs in the PL/SQL block because they may result in an exception when the gsql tool is executed with the **-r** parameter specified.

Types

PL/SQL blocks are classified into the following types:

- Anonymous block: a dynamic block that can be executed only for once. For details about the syntax, see **Figure 2-6**.
- Subprogram: a stored procedure, function, operator, or advanced package stored in a database. A subprogram can be called in other programs after it is created.

2.5.2 Anonymous Blocks

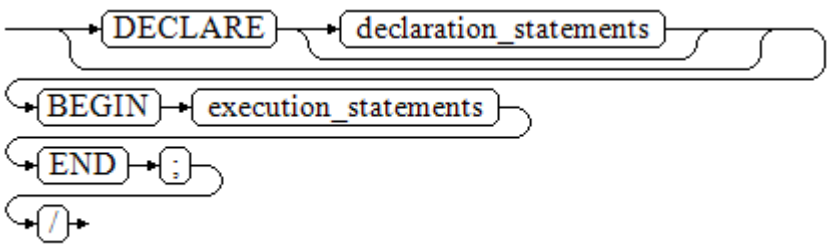
An anonymous block applies to a script infrequently executed or a one-off activity. An anonymous block is executed in a session and is not stored.

Syntax

```
[ DECLARE [ declaration_statements ] ]
BEGIN execution_statements
END ;
/
```

Figure 2-6 shows the syntax diagram of an anonymous block.

Figure 2-6 anonymous_block::=



- The execute part of an anonymous block starts with a BEGIN statement, has a break with an END statement, and ends with a semicolon (;). Type a slash (/) and press **Enter** to execute the statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

The terminator "/" must be written in an independent row.

- The declaration section includes the variable definition, type, and cursor definition.
- A simplest anonymous block does not execute any commands. However, at least one statement, even a **NULL** statement, must be presented in any implementation blocks.

Examples

The following lists basic anonymous block programs:

```
-- Null statement block
gaussdb=# BEGIN
    NULL;
END;
/

-- Display information on the console.
gaussdb=# BEGIN
    dbe_output.print_line('hello world!');
END;
/
hello world!
ANONYMOUS BLOCK EXECUTE
-- Display variable content on the console.
gaussdb=# DECLARE
    my_var VARCHAR2(30);
BEGIN
    my_var :='world';
    dbe_output.print_line('hello'||my_var);
END;
/
helloworld
ANONYMOUS BLOCK EXECUTE
```

Using the JDBC Driver to Bind Parameters in an Anonymous Block

NOTICE

- This function supports only ORA-compatible databases and only Oracle Database-style anonymous blocks.
- In the full encryption scenario:
  - When **enable_ce** is set to **1** for JDBC connections, parameters cannot be bound to anonymous blocks.
- If the variables (including the explicitly defined variables or types) defined in an anonymous block conflict with bind variables ($1, $2...) in names, an error will be reported.
- An anonymous block declared as an autonomous transaction does not support bind parameters.
- The maximum number of parameters that can be bound to an anonymous block is 32767, and the maximum number of output parameters that can be bound is 1664.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Prerequisites: You have connected to the database using the connection object **conn**.

To use the JDBC driver to bind parameters to an anonymous block, perform the following steps:

Step 1 Create the call statement object cstmt by calling the prepareCall method of Connection.

```
CallableStatement cstmt = conn.prepareCall("DECLARE var int; BEGIN var := ?; ? := 50; END;");
```

Step 2 Set parameters by calling the setInt method of CallableStatement.

```
cstmt.setInt(1, 20);
```

Step 3 Register an output parameter by calling the registerOutParameter method in CallableStatement.

```
cstmt.registerOutParameter(2, Types.INTEGER);  // Register an out parameter of the integer type.
```

Step 4 Call the execute method of CallableStatement to execute the anonymous block statement.

```
cstmt.execute();
```

Step 5 Obtain the out parameter by calling the getInt method of CallableStatement.

```
int out = cstmt.getInt(2);
```

Step 6 Close the call statement object cstmt by calling the close method of CallableStatement.

```
cstmt.close();
```

----End

NOTE

- When the JDBC driver for 507.0.0 is used to connect to a database of an earlier version (503.1.0 or later), the JDBC driver cannot be used to bind parameters to anonymous blocks. When the JDBC driver for 503.1.0 or later is used to connect to a database of version 507.0.0, the JDBC driver can be used to bind input parameters to anonymous blocks, but cannot be used to bind output parameters to anonymous blocks.
- During the upgrade observation period after the upgrade from a version earlier than 507.0.0 to 507.0.0 or later, if the JDBC driver for the target version is used to connect to the database, the JDBC driver can be used to bind parameters to anonymous blocks. If the JDBC driver for the source version (503.1.0 or later) is used to connect to the database of version 507.0.0, the JDBC driver can be used to bind input parameters to anonymous blocks, but cannot be used to bind output parameters to anonymous blocks.

2.5.3 Subprograms

Subprograms are objects stored in a database, including stored procedures, functions, operators, and advanced packages.

After a subprogram is created in the database, it can be called by other programs.

2.5.3.1 Standalone Subprograms

Standalone subprograms include stored procedures, functions, and packages created in a schema.

For details, see:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **CREATE PACKAGE**
- **CREATE FUNCTION**
- **CREATE PROCEDURE**

2.5.3.2 Package Subprograms

A subprogram created inside a package is a package subprogram. You can declare and create a stored procedure or function in the package body.

For details, see **CREATE PACKAGE**.

2.5.3.3 Nested Subprograms

A subprogram created inside a PL/SQL block is a nested subprogram. You can declare and create a stored procedure or function inside an anonymous block, stored procedure, function, or a stored procedure/function inside a package.

Precautions

- It is used in ORA-compatible databases.
- The maximum number of nesting layers is specified by the GUC parameter **max_subpro_nested_layers**. The default value is **3**, and the value range is 0 to 100. If a subprogram is nested under an anonymous block, the layer of the anonymous block is not counted in the total number of layers.
- Nested subprograms do not support overloading or SETOF.
- Nested subprograms cannot be defined as autonomous transactions. They can call stored procedures or functions that contain autonomous transactions.
- Nested functions cannot be directly called and must have return values. Nested stored procedures cannot be called in expressions.
- Nested subprograms cannot be called by PERFORM. Dynamic statements cannot contain nested subprograms.
- Currently, only the following modifiers are supported for nested subprograms:
  ```
  {IMMUTABLE  | STABLE  | VOLATILE }
  {CALLED ON NULL INPUT | RETURNS NULL ON NULL INPUT | STRICT }
  ```
- Only one qualifier can reference a nested subprogram or a variable of a nested subprogram.
- When the return value type of a nested function is the record type customized by the function, subfunc().col cannot be used to access the column attribute of the return value of the nested function. As a result, an error is reported during execution.
- The declaration of a nested subprogram must be at the end of the declarative part (declaring the nested subprogram after the declaration of variables, cursors, and types is complete).
- Nested subprograms can be called only inside declared functions or stored procedures and cannot be used externally.
- Nested subprograms do not support debugger breakpoints, but support step-by-step debugging.
- Other precautions are the same as those for stored procedures and functions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Syntax

- Syntax format for creating a nested stored procedure:

```
PROCEDURE procedure_name [ (parameters) ]
    [{IMMUTABLE  | STABLE  | VOLATILE }
     | {CALLED ON NULL INPUT | RETURNS NULL ON NULL INPUT | STRICT }]
    { IS | AS }
    [ declarations ]
BEGIN
    plsql_body
END;
```

- Syntax for creating a nested function:

```
FUNCTION function_name  [ (parameters) ] RETURN rettype
    [{IMMUTABLE  | STABLE  | VOLATILE }
     | {CALLED ON NULL INPUT | RETURNS NULL ON NULL INPUT | STRICT }]
    { IS | AS }
    [ declarations ]
BEGIN
    plsql_body
END;
```

  In the **declarations** part, you can define the nested subprograms of the lower layer.

- Example:

```
-- Create a stored procedure.
CREATE OR REPLACE PROCEDURE proc_test() AS
    -- Declare and define a nested stored procedure.
    PROCEDURE proc_sub() IS
    BEGIN
        dbe_output.put_line('this is subprogram');
    END;
BEGIN
    dbe_output.put_line('this is a procedure');
    -- Call the nested stored procedure inside an execution block.
    proc_sub();
END;
/
    -- Call a stored procedure externally.
BEGIN
    proc_test;
END;
/
-- Output the result.
this is a procedure
this is subprogram
ANONYMOUS BLOCK EXECUTE
```

Declaration and Definition Rules of Nested Subprograms

- Nested subprograms cannot be declared or defined repeatedly. Overloading is not supported.
- The identifier of a nested subprogram cannot be the same as the variable name or keyword.
- You can declare a subprogram before defining it. All declared subprograms must be defined in the subsequent declaration blocks.

Calling Rules of Nested Subprograms

- Nested subprograms can call themselves to implement recursive calls.
- Nested subprograms can call subprograms declared in the upper scope.
- Nested subprograms can call locally declared lower-layer subprograms, but cannot access nested subprograms in lower-layer subprograms.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Nested subprograms can call subprograms declared before themselves in the same scope.

Example:

```
-- Call itself.
CREATE OR REPLACE PROCEDURE proc_test(var1 int) AS
    PROCEDURE proc_sub(var2 int) IS
    BEGIN
        dbe_output.put_line('var = ' || var2);
        IF var2 > 1 THEN
            proc_sub(var2 - 1);
        END IF;
    END;
BEGIN
    proc_sub(var1);
END;
/
BEGIN
    proc_test(3);
END;
/
-- Output the result.
var = 3
var = 2
var = 1

-- Call the upper-layer subprogram.
CREATE OR REPLACE PROCEDURE proc_test(var1 int) AS
    PROCEDURE procsub_1(var2 int) IS
    BEGIN
 proc_test(var2 - 1);
    END;
BEGIN
    dbe_output.put_line('proc_test var1 = ' || var1);
 IF var1 > 1 THEN
       procsub_1(var1);
 END IF;
END;
/
BEGIN
    proc_test(3);
END;
/
-- Output the result.
proc_test var1 = 3
proc_test var1 = 2
proc_test var1 = 1

-- Call the lower-layer subprogram declared locally.
CREATE OR REPLACE PROCEDURE proc_test() AS
    PROCEDURE proc_sub1 IS
        procedure proc_sub2 IS
        BEGIN
            dbe_output.put_line('--this is subprogram2 begin');
            dbe_output.put_line('--this is subprogram2 end');
        END;
    BEGIN
 dbe_output.put_line('this is subprogram1 begin');
 proc_sub2();
 dbe_output.put_line('this is subprogram1 end');
    END;
BEGIN
    dbe_output.put_line('this is a procedure begin');
    proc_sub1();
    dbe_output.put_line('this is a procedure end');
END;
/
BEGIN
    proc_test;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
END;
/
-- Output the result.
this is a procedure begin
this is subprogram1 begin
--this is subprogram2 begin
--this is subprogram2 end
this is subprogram1 end
this is a procedure end
```

Variables of Nested Subprograms

Variable types include basic types, record types, TABLE OF types, cursor types, and varray types supported by PL/SQL.

- Accessible variables:
  - Variable declared by itself.
  - Variable declared by the upper-layer subprogram.
- Variable access rules:
  - If a variable does not have a qualifier, it is searched in the current program. If the variable name does not exist, the variable is searched at the upper layer, and so on.
  - If a variable has a qualifier, it is searched in the area of the qualifier. Currently, only one qualifier can be used.

2.6 Basic Statements

In PL/SQL programming, you need to define variables, assign values, and call stored procedures. This section describes basic PL/SQL statements, including variable definition statements, value assignment statements, call statements, and return statements.

NOTE

Do not call the SQL statements containing passwords in the stored procedures because authorized users may view the stored procedure file in the database, causing password leakage. If the stored procedure contains other sensitive information, you are advised to configure the access permission for the stored procedure to ensure the security of sensitive data.

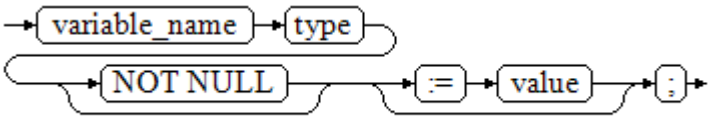
2.6.1 Variable Definition Statements

This section describes the declaration of variables in the PL/SQL and the scope of this variable in code.

Variable Declaration

Figure 2-7 shows the variable declaration syntax.

Figure 2-7 declare_variable::=



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The syntax is described as follows:

- **variable_name** indicates the name of a variable.
- **type** indicates the type of a variable.
- **value** indicates the initial value of the variable. (If the initial value is not given, **NULL** is taken as the initial value.) **value** can also be an expression.

Examples

```
gaussdb=# DECLARE
    emp_id  INTEGER := 7788; -- Define a variable and assign a value to it.
BEGIN
    emp_id := 5*7784; -- Assign a value to the variable.
END;
/
```

In addition to the declaration of basic variable types, **%TYPE** and **%ROWTYPE** can be used to declare variables related to table columns or table definitions.

%TYPE Attribute

%TYPE declares a variable to be of the same data type as a previously declared variable (for example, a column in a table). For example, if you want to define the *my_name* variable whose data type is the same as the data type of the **firstname** column in the **employee** table, you can define the variable as follows:

```
my_name employee.firstname%TYPE
-- Example
DROP TABLE IF EXISTS employee;
CREATE TABLE employee(firstname varchar,secondname varchar);
DECLARE
    my_name employee.firstname%TYPE;
BEGIN
    my_name = 'abc';
    DBE_OUTPUT.PRINT_LINE(my_name);
END;
/
```

In this way, you do not need to know the data type of the **firstname** column in the **employee** table. In addition, you do not need to modify the type definition of **my_name** when the data type of the **firstname** column changes.

%ROWTYPE Attribute

%ROWTYPE declares data types of a set of data. It stores a row of table data or obtains results from a cursor. If you want to define a set of data with the same column names and column types as the **employee** table, you can define the data as follows:

```
my_employee employee%ROWTYPE
DROP TABLE IF EXISTS employee;
CREATE TABLE employee(firstname varchar,secondname varchar);
DECLARE
    my_employee employee%ROWTYPE;
BEGIN
    my_employee.firstname := 'ab1';
    my_employee.secondname := 'ab2';
    DBE_OUTPUT.PRINT_LINE(my_employee.firstname);
    DBE_OUTPUT.PRINT_LINE(my_employee.secondname);
END;
/
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- In the environment with multiple CNs, the **%ROWTYPE** and **%TYPE** attributes of the temporary table cannot be declared in a stored procedure. The temporary table is valid only in the current session. During compilation, other CNs cannot view the temporary table of the current CN. Therefore, if there are multiple CNs, the system displays a message indicating that the temporary table does not exist.
- **%TYPE** cannot reference the record variables that are not across packages or the cursor variables of a column type.
- **%ROWTYPE** cannot reference the type of a composite variable or a record variable.
- The non-PACKAGE *record%TYPE* cannot be used as the variable type or input/output parameter type.
- *View%ROWTYPE* or **SCHEMA.***view%ROWTYPE* cannot be used as the input/output parameter type.
- The **PACKAGE.***record%TYPE* or **SCHEMA.PACKAGE.***record%TYPE* cannot be used as the input/output parameter type.
- *Table/View*.**column.***column%TYPE* or **SCHEMA.***Table/View*.**column.***column%TYPE* cannot be nested with one or more layers as variable types or the input/output parameter type.
- *Record*.**column.***column%TYPE* and **PACKAGE.***record*.**column.***column%TYPE* cannot be nested with a column type of records at one or more layers as the variable type or input/output parameter type.
- A cursor variable of more layers, such as **PACKAGE.***cursor%ROWTYPE* and **SCHEMA.PACKAGE.***cursor%ROWTYPE*, cannot be used as the parameter type or input/output parameter type.
- In an anonymous block, *cursor_name%ROWTYPE* does not contain column information. When it is used as the base type of a collection element, for example, BULK COLLECT INTO, an exception may occur.

Scope of a Variable

The scope of a variable defines the accessibility and availability of the variable in a code block. A variable is valid only within its scope.

- A variable must be declared in the DECLARE part, that is, the BEGIN-……END block structure must be established. The structure requires that a variable must be declared before being used. That is, a variable has different scopes and lifetimes during a process.
- A variable can be defined repeatedly in different scopes. The definition of the inner scope overwrites that of the outer scope.
- Variables defined in an outer block can be used in nested blocks, but variables defined in nested blocks cannot be accessed by the outer block.

Example

```
gaussdb=# DECLARE
    emp_id  INTEGER :=7788; -- Define a variable and assign a value to it.
    outer_var  INTEGER :=6688; -- Define a variable and assign a value to it.
BEGIN
    DECLARE
        emp_id INTEGER :=7799; -- Define a variable and assign a value to it.
        inner_var  INTEGER :=6688; -- Define a variable and assign a value to it.
    BEGIN
        dbe_output.print_line('inner emp_id ='||emp_id); -- Display the value 7799.
        dbe_output.print_line('outer_var ='||outer_var); -- Reference a variable of an outer block.
    END;
    dbe_output.print_line('outer emp_id ='||emp_id); -- Display the value 7788.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

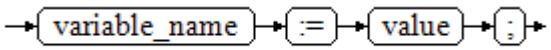
```
END;
/
```

2.6.2 Assignment Statements

Variable Syntax

Figure 2-8 shows the syntax for assigning a value to a variable.

Figure 2-8 assignment_value::=



The syntax is described as follows:

- **variable_name** indicates the name of a variable.
- **value** can be a value or an expression. The type of **value** must be compatible with the type of **variable_name**.

Variable Value Assignment Example

```
DECLARE
    emp_id  INTEGER := 7788; -- Assign a value.
BEGIN
    emp_id := 5; -- Assign a value.
    DBE_OUTPUT.PRINT_LINE(emp_id);
    emp_id := 5*7784;
    DBE_OUTPUT.PRINT_LINE(emp_id);
END;
/
-- The result is as follows:
5
38920
ANONYMOUS BLOCK EXECUTE
```

NOTE

After the GUC parameter is enabled (by running **SET behavior_compat_options = 'proc_outparam_override';**), the **value** of an assignment statement cannot call an overloaded function whose return type is composite or array type in the form of a reference column (for example, func(x).a or func(x)[1]).

INTO/BULK COLLECT INTO

Values returned by statements in a stored procedure are stored in variables. BULK COLLECT INTO allows some or all returned values to be temporarily stored in an array.

Syntax

```
SELECT select_expressions INTO [STRICT] target FROM ...
SELECT INTO [STRICT] target [FROM ..]
EXECUTE [IMMEDIATE] select_expressions BULK COLLECT INTO target ...
```

The syntax is described as follows:

- **select_expressions**: SQL statement for query. You can use basic SQL commands and INTO clauses to assign the result of a single row or multiple columns to a variable (such as record, row, or scalar variable list).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **target**: target variable, which can be a record variable, a row variable, or a comma-separated list of simple variables and record/row fields.
- **STRICT** (Optional): If **behavior_compat_options** is set to **'select_into_return_null'** (the parameter is disabled by default) and this option is specified, the query result must contain a row of result set that is not empty. Otherwise, an error is reported. The error information may be NO_DATA_FOUND (no row), TOO_MANY_ROWS (more than one row), or QUERY_RETURNED_NO_ROWS (no data is returned). If the option is not specified, the empty result set can be returned.

NOTE

- BULK COLLECT INTO can assign values to arrays or collections in batches. Use the LIMIT clause properly to prevent performance deterioration caused by excessive data operations.
- INTO and BULK COLLECT INTO support only direct nesting of record values with less than four layers.
- To return an empty result set, you need to enable PG-compatible mode during database initialization. Set the GUC parameter **behavior_compat_options** to **'select_into_return_null'**. The compatibility mode is enabled. If the GUC parameter **behavior_compat_options** is not set, the compatibility mode is disabled.
- For array variables, elements in parentheses (()) are preferentially identified as index sets. Therefore, expressions with parentheses cannot be written after array variables. For example, **SELECT (1+3) INTO va(5)** cannot be written as **SELECT INTO va(5) (1+3)** or **SELECT INTO va[5] (1+3)**.
- BULK COLLECT INTO can be used only in an ORA-compatible database.
- The IMMEDIATE keyword is used only for syntax compatibility and has no actual meaning.
- The SELECT INTO data in a stored procedure is stored in the specified output variables. If the number of output variables is greater than the number of data columns in the actual query result set, null values are assigned to the extra variables, and no error is reported. If the number of output variables is less than the number of data columns in the actual cursor query result set, the extra columns are silently discarded, and no error is reported.

Examples

```
gaussdb=# DROP TABLE IF EXISTS customers;
gaussdb=# CREATE TABLE customers(id int,name varchar);
gaussdb=# INSERT INTO customers VALUES(1,'ab');
gaussdb=# DECLARE
    my_id integer;
BEGIN
    select id into my_id from customers limit 1; -- Assign a value.
END;
/
ANONYMOUS BLOCK EXECUTE
gaussdb=# DECLARE
    type id_list is varray(6) of customers.id%type;
    id_arr id_list;
BEGIN
    select id bulk collect into id_arr from customers order by id DESC limit 20; -- Assign values in batches.
END;
/
ANONYMOUS BLOCK EXECUTE
gaussdb=#  DECLARE
    TYPE id_list IS varray(6) OF customers.id%type;
    id_arr id_list;
    sql_qry varchar2(150);
BEGIN
    sql_qry := 'SELECT id FROM customers ORDER BY id DESC LIMIT 20';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

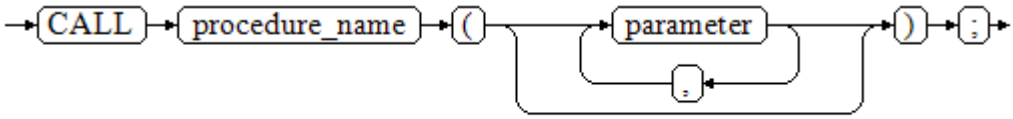
```
   EXECUTE IMMEDIATE sql_qry BULK COLLECT INTO id_arr; -- Assign values in batches.
END;
/
ANONYMOUS BLOCK EXECUTE
```

2.6.3 Call Statements

Syntax

Figure 2-9 shows the syntax diagram for a call statement.

Figure 2-9 call_clause::=



The syntax is described as follows:

- **procedure_name** specifies the name of a stored procedure.
- **parameter** specifies the parameters for the stored procedure. You can set no parameter or multiple parameters.

Examples

```
-- Create a table.
gaussdb=# CREATE SCHEMA hr;
gaussdb=# SET CURRENT_SCHEMA = hr;
gaussdb=# CREATE TABLE staffs
(
  section_id INTEGER,
  salary INTEGER
);
gaussdb=# INSERT INTO staffs VALUES (30, 10);
gaussdb=# INSERT INTO staffs VALUES (30, 20);

-- Create the stored procedure proc_staffs.
gaussdb=# CREATE OR REPLACE PROCEDURE proc_staffs
(
section     NUMBER(6),
salary_sum out NUMBER(8,2),
staffs_count out INTEGER
)
IS
BEGIN
SELECT sum(salary), count(*) INTO salary_sum, staffs_count FROM hr.staffs where section_id = section;
END;
/
CREATE PROCEDURE
-- Create the stored procedure proc_return.
gaussdb=# CREATE OR REPLACE PROCEDURE proc_return
AS
v_num NUMBER(8,2);
v_sum INTEGER;
BEGIN
proc_staffs(30, v_sum, v_num);  -- Call statement.
dbe_output.print_line(v_sum||'#'||v_num);
RETURN;  -- Return statement.
END;
/
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
CREATE PROCEDURE
-- Call the stored procedure proc_return.
gaussdb=# CALL proc_return();
30#2.00
 proc_return
-------------

(1 row)

-- Drop the stored procedures.
gaussdb=# DROP PROCEDURE proc_staffs;
gaussdb=# DROP PROCEDURE proc_return;

-- Create the function func_return.
gaussdb=# CREATE OR REPLACE FUNCTION func_return returns void
language plpgsql
AS $$
DECLARE
v_num INTEGER := 1;
BEGIN
dbe_output.print_line(v_num);
RETURN;  -- Return statement.
END $$;
CREATE FUNCTION

-- Call the function func_return.
gaussdb=# CALL func_return();
1
 func_return
-------------

(1 row)

-- Drop the function.
gaussdb=# DROP FUNCTION func_return;

-- Drop the current schema.
gaussdb=# DROP SCHEMA hr CASCADE;
```

2.7 Dynamic Statements

2.7.1 Dynamically Executing Query Statements

This section describes dynamic query statements and their execution examples. GaussDB supports dynamic query using EXECUTE IMMEDIATE and OPEN FOR. The former dynamically executes SELECT statements while the latter uses the cursor. If you need to store query results in a data set, use OPEN FOR.

EXECUTE IMMEDIATE

Figure 2-10 shows the syntax diagram.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 2-10 EXECUTE IMMEDIATE dynamic_select_clause::=

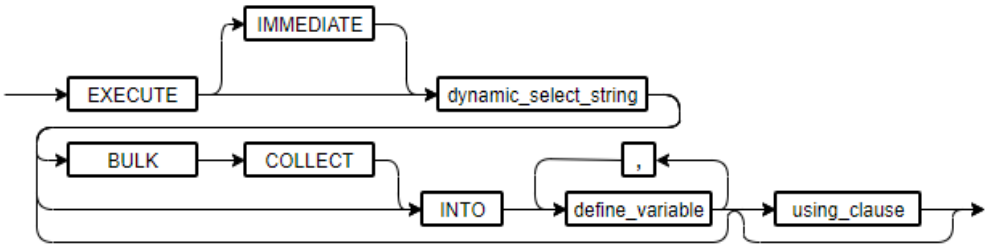
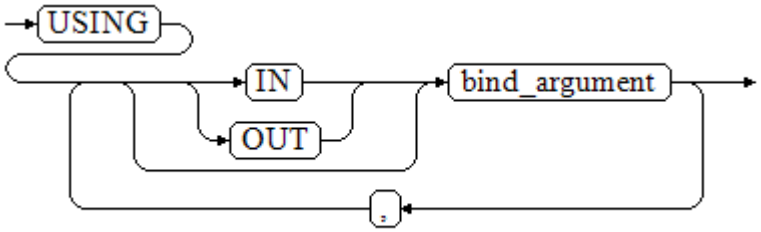


Figure 2-11 shows the syntax diagram for the USING clause.

Figure 2-11 using_clause::=



The syntax is described as follows:

- **define_variable**: specifies a variable to store query results.
- **USING IN** *bind_argument*: specifies a variable whose value is passed to a dynamic SQL statement. The variable is used when a dynamic placeholder exists in *dynamic_select_string*.
- **USING OUT** *bind_argument*: specifies a variable that stores a value returned by the dynamic SQL statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- In query statements, INTO and OUT cannot coexist.
- A placeholder name starts with a colon (:) and can be followed by digits, characters, or character strings (excluding quoted digits, characters, or character strings). A placeholder name corresponds to *bind_argument* in the USING clause.
- *bind_argument* can only be a value, variable, or expression. It cannot be a database object such as a table name, column name, and data type. That is, *bind_argument* cannot be used to specify schema objects for dynamic SQL statements. If a stored procedure needs to specify database objects by declaring parameters to construct dynamic SQL statements (generally, DDL statements), you are advised to use double vertical bars (||) to concatenate *dynamic_select_clause* with a database object.
- A dynamic PL/SQL block allows duplicate placeholders. That is, a placeholder can correspond to only one *bind_argument* in the USING clause. When the GUC parameter **behavior_compat_options** is set to **dynamic_sql_compat**, the bind arguments in the USING clause are matched in sequence based on the placeholder sequence. Duplicate placeholders will not be identified as the same placeholder.
- The IMMEDIATE keyword is used only for syntax compatibility and has no actual meaning.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS hr CASCADE;
gaussdb=# CREATE SCHEMA hr;
gaussdb=# SET CURRENT_SCHEMA = hr;
gaussdb=# CREATE TABLE staffs
(
  staff_id NUMBER,
  first_name VARCHAR2,
  salary NUMBER
);
gaussdb=# INSERT INTO staffs VALUES (200, 'mike', 5800);
gaussdb=# INSERT INTO staffs VALUES (201, 'lily', 3000);
gaussdb=# INSERT INTO staffs VALUES (202, 'john', 4400);

-- Retrieve values from dynamic statements (INTO clause).
gaussdb=# DECLARE
   staff_count  VARCHAR2(20);
BEGIN
   EXECUTE IMMEDIATE 'select count(*) from hr.staffs'
      INTO staff_count;
   dbe_output.print_line(staff_count);
END;
/
3
ANONYMOUS BLOCK EXECUTE
-- Pass and retrieve values (the INTO clause is used before the USING clause).
gaussdb=# CREATE OR REPLACE PROCEDURE dynamic_proc
AS
   staff_id     NUMBER(6) := 200;
   first_name   VARCHAR2(20);
   salary       NUMBER(8,2);
BEGIN
   EXECUTE IMMEDIATE 'select first_name, salary from hr.staffs where staff_id = :1'
      INTO first_name, salary
      USING IN staff_id;
   dbe_output.print_line(first_name || ' ' || salary);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

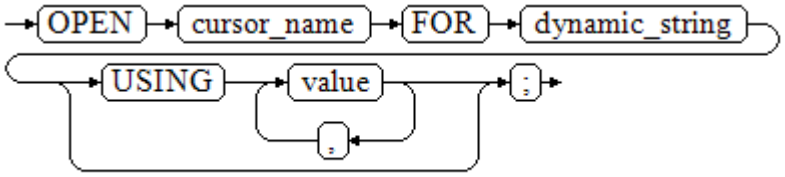
```
END;
/
CREATE PROCEDURE
-- Call a stored procedure.
gaussdb=# CALL dynamic_proc();
mike 5800.00
 dynamic_proc
--------------

(1 row)

-- Drop a stored procedure.
gaussdb=# DROP PROCEDURE dynamic_proc;

-- Drop the current schema.
gaussdb=# DROP SCHEMA hr CASCADE;
```

OPEN FOR

Dynamic query statements can be executed by using OPEN FOR to open dynamic cursors.

Figure 2-12 shows the syntax diagram.

Figure 2-12 open_for::=



Parameters:

- *cursor_name*: specifies the name of the cursor to be opened.
- *dynamic_string*: specifies the dynamic query statement.
- **USING** *value*: applies when a placeholder exists in *dynamic_string*.

For details about use of cursors, see **Cursors**.

Example:

```
gaussdb=# CREATE SCHEMA hr;
gaussdb=# SET CURRENT_SCHEMA = hr;
gaussdb=# DROP TABLE IF EXISTS staffs;
gaussdb=# CREATE TABLE staffs
(
  section_id NUMBER,
  first_name VARCHAR2,
  phone_number VARCHAR2,
  salary NUMBER
  );
gaussdb=# INSERT INTO staffs VALUES (30, 'mike', '13567829252', 5800);
gaussdb=# INSERT INTO staffs VALUES (40, 'john', '17896354637', 4000);

gaussdb=# DECLARE
    name          VARCHAR2(20);
    phone_number  VARCHAR2(20);
    salary        NUMBER(8,2);
    sqlstr        VARCHAR2(1024);

    TYPE app_ref_cur_type IS REF CURSOR; -- Define the cursor type.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    my_cur app_ref_cur_type; -- Define the cursor variable.

BEGIN
    sqlstr := 'select first_name,phone_number,salary from hr.staffs
        where section_id = :1';
    OPEN my_cur FOR sqlstr USING '30'; -- Open the cursor. USING is optional.
    FETCH my_cur INTO name, phone_number, salary; -- Retrieve the data.
    WHILE my_cur%FOUND LOOP
        dbe_output.print_line(name||'#'||phone_number||'#'||salary);
        FETCH my_cur INTO name, phone_number, salary;
    END LOOP;
    CLOSE my_cur; -- Close the cursor.
END;
/
mike#13567829252#5800.00
ANONYMOUS BLOCK EXECUTE
gaussdb=# DROP TABLE staffs;
```

2.7.2 Dynamically Executing Non-Query Statements

Syntax

Figure 2-13 shows the syntax diagram.

Figure 2-13 noselect::=

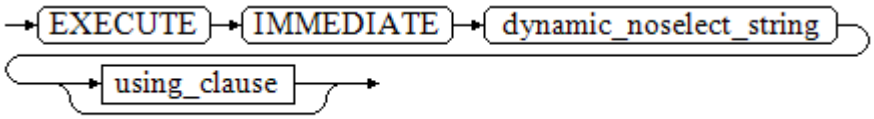
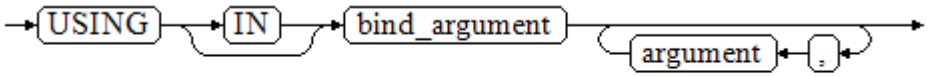


Figure 2-14 shows the syntax diagram for the USING clause.

Figure 2-14 using_clause::=



The above syntax diagram is explained as follows:

USING IN *bind_argument* is used to specify the variable whose value is passed to the dynamic SQL statement. The variable is used when a placeholder exists in *dynamic_noselect_string*. That is, a placeholder is replaced by the corresponding *bind_argument* when a dynamic SQL statement is executed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- *bind_argument* can only be a value, variable, or expression, and cannot be a database object such as a table name, column name, and data type. If a stored procedure needs to specify database objects by declaring parameters to construct dynamic SQL statements (generally, DDL statements), you are advised to use double vertical bars (||) to concatenate *dynamic_select_clause* with a database object.
- In addition, duplicate placeholders are allowed in dynamic statements. A placeholder can correspond to only one *bind_argument* by position. When the GUC parameter **behavior_compat_options** is set to **dynamic_sql_compat**, the *bind_argument* in the USING clause is matched in the placeholder sequence. Duplicate placeholders will not be identified as the same placeholder (placeholder names cannot contain quoted digits, characters, or strings).

Examples

```
-- Create a table.
gaussdb=# CREATE TABLE sections_t1
(
   section       NUMBER(4) ,
   section_name  VARCHAR2(30),
   manager_id    NUMBER(6),
   place_id      NUMBER(4)
)
DISTRIBUTE BY hash(manager_id);

-- Declare a variable.
gaussdb=# DECLARE
   section       NUMBER(4) := 280;
   section_name  VARCHAR2(30) := 'Info support';
   manager_id    NUMBER(6) := 103;
   place_id      NUMBER(4) := 1400;
   new_colname   VARCHAR2(10) := 'sec_name';
BEGIN
-- Execute the query.
    EXECUTE IMMEDIATE 'insert into sections_t1 values(:1, :2, :3, :4)'
       USING section, section_name, manager_id,place_id;
-- Execute the query (duplicate placeholders).
    EXECUTE IMMEDIATE 'insert into sections_t1 values(:1, :2, :3, :1)'
       USING section, section_name, manager_id;
-- Run the ALTER statement. You are advised to use double vertical bars (||) to concatenate the dynamic
DDL statement with a database object.
    EXECUTE IMMEDIATE 'alter table sections_t1 rename section_name to ' || new_colname;
END;
/
ANONYMOUS BLOCK EXECUTE
-- Query data.
gaussdb=# SELECT * FROM sections_t1;
 section |   sec_name   | manager_id | place_id
---------+--------------+------------+----------
     280 | Info support |        103 |     1400
     280 | Info support |        103 |      280
(2 rows)

-- Drop a table.
gaussdb=# DROP TABLE sections_t1;
```

2.7.3 Dynamically Calling Stored Procedures

When dynamically calling a stored procedure, you must use an anonymous statement block to wrap the stored procedure or statement block. The EXECUTE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

IMMEDIATE...USING statement is used together with IN and OUT to implement parameter input and output.

Syntax

Figure 2-15 shows the syntax diagram.

Figure 2-15 call_procedure::=

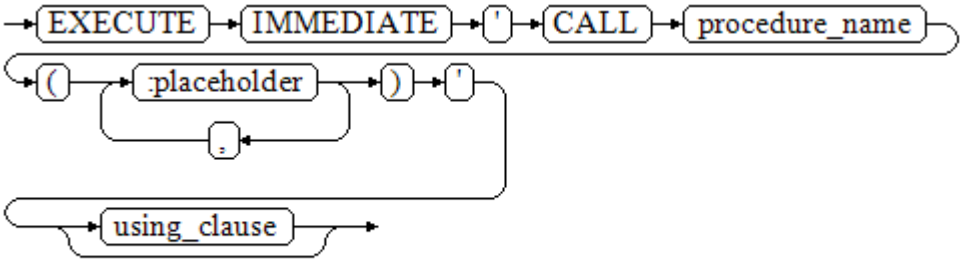
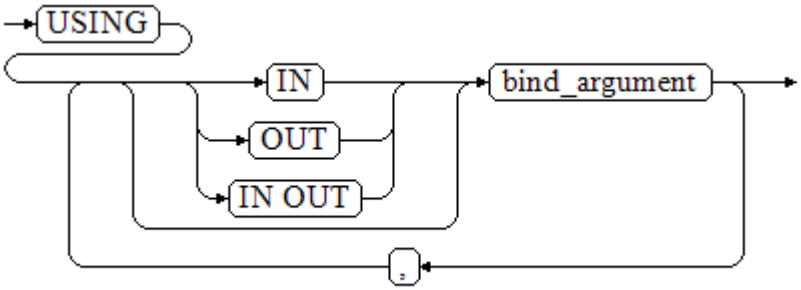


Figure 2-16 shows the syntax diagram for the USING clause.

Figure 2-16 using_clause::=



The above syntax diagram is explained as follows:

- **CALL** *procedure_name*: calls a stored procedure.
- **[:placeholder1,:placeholder2, ...]**: indicates the list of placeholders for stored procedure parameters. Placeholder names cannot contain quoted digits, characters, or character strings. The numbers of the placeholders and parameters are the same.
- **USING [IN|OUT|IN OUT]** *bind_argument*: specifies where the variable passed to the stored procedure parameter value is stored. The modifiers in front of *bind_argument* and of the corresponding parameter are the same.

When a stored procedure is called, => cannot be used to skip parameters.

Examples

```
-- Create the stored procedure proc_add.
gaussdb=# CREATE OR REPLACE PROCEDURE proc_add
(
    param1    in   INTEGER,
    param2    out  INTEGER,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    param3    in   INTEGER
)
AS
BEGIN
   param2:= param1 + param3;
END;
/

gaussdb=# DECLARE
    input1 INTEGER:=1;
    input2 INTEGER:=2;
    statement  VARCHAR2(200);
    param2     INTEGER;
BEGIN
   -- Declare the call statement.
    statement := 'call proc_add(:col_1, :col_2, :col_3)';
   -- Execute the statement.
    EXECUTE IMMEDIATE statement
        USING IN input1, OUT param2, IN input2;
    dbe_output.print_line('result is: '||to_char(param2));
END;
/
result is: 3
ANONYMOUS BLOCK EXECUTE
-- Drop the stored procedure.
gaussdb=# DROP PROCEDURE proc_add;
```

2.7.4 Dynamically Calling Anonymous Blocks

Dynamically calling anonymous blocks refers to executing anonymous blocks in dynamic statements. The EXECUTE IMMEDIATE...USING statement is used together with IN and OUT to implement parameter input and output.

Syntax

Figure 2-17 shows the syntax for dynamically calling an anonymous block.

Figure 2-17 call_anonymous_block::=

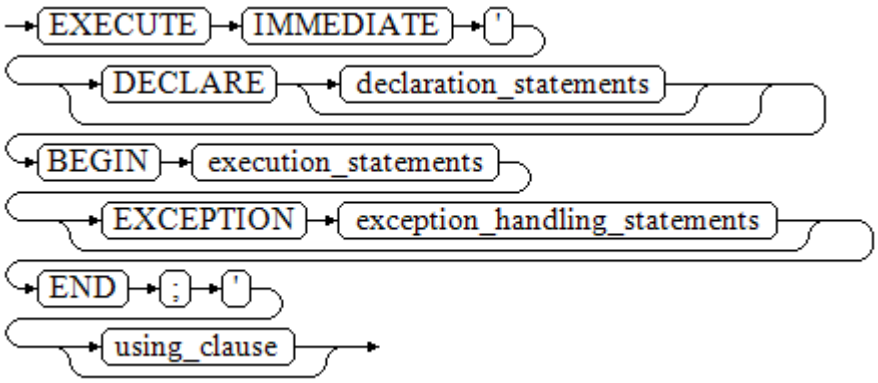
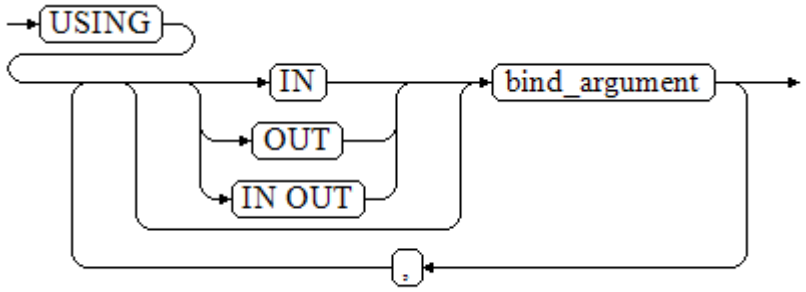


Figure 2-18 shows the syntax diagram for the USING clause.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 2-18 using_clause::=



The above syntax diagram is explained as follows:

- The execute part of an anonymous block starts with a BEGIN statement, has a break with an END statement, and ends with a semicolon (;).
- **USING [IN|OUT|IN OUT]** *bind_argument*: specifies where the variable passed to the stored procedure parameter value is stored. The modifiers in front of *bind_argument* and of the corresponding parameter are the same.
- The INOUT parameters in the middle of an anonymous block are designated by placeholders. The numbers of the placeholders and parameters are the same. The sequences of the parameters corresponding to the placeholders and the USING parameters are the same.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- Parameters can be bound only when SQL statements and stored procedures are called in anonymous blocks. For example, expressions and cursors are used in anonymous blocks, and dynamic statements are called in nested mode in anonymous blocks.
- Output parameters cannot be bound when the SELECT INTO statement in an anonymous block calls FUNCTION/PROCEDURE that contains output parameters.
- Output parameters of the composite type cannot be bound to an anonymous block.
- Variables declared in an anonymous block and bind variables cannot be used in the same statement at the same time.
- The PERFORM keyword cannot be used to call a stored procedure when bind variables are used.
- When a stored procedure is called, only bind variables can be used as INOUT parameters. Expressions (for example, 1+:va) cannot be used as INOUT parameters. In addition, placeholder names cannot contain quoted digits, characters, or character strings.
- The IMMEDIATE keyword is used only for syntax compatibility and has no actual meaning.
- When the bound input parameter type is refcursor, the modification in the stored procedure is isolated from the input parameter.
- When the **dynamic_sql_check** parameter is enabled and the number of placeholders is the same as the number of parameters, an error will be reported if placeholders with the same name are used as anonymous block parameters. In this case, you need to rename the placeholders. For details, see **Examples**.

Examples

```
gaussdb=# DROP SCHEMA IF EXISTS hr CASCADE;
gaussdb=# CREATE SCHEMA hr;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = hr;
SET
gaussdb=# CREATE TABLE staffs
(
  staff_id NUMBER,
  first_name VARCHAR2,
  salary NUMBER
);
CREATE TABLE
gaussdb=# INSERT INTO staffs VALUES (200, 'mike', 5800);
INSERT 0 1
gaussdb=# INSERT INTO staffs VALUES (201, 'lily', 3000);
INSERT 0 1
gaussdb=# INSERT INTO staffs VALUES (202, 'john', 4400);
INSERT 0 1

-- Create overloaded functions.
gaussdb=# CREATE OR REPLACE PACKAGE pkg1
IS
   PROCEDURE plus(var1 in int, var2 int, var3 out int);
   PROCEDURE plus(var1 in out int);
END pkg1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
/
CREATE PACKAGE
gaussdb=# CREATE OR REPLACE PACKAGE BODY pkg1 IS
  PROCEDURE plus(var1 in int, var2 int, var3 out int)
  AS
  BEGIN
    var3 = var1 + var2 + 1;
  END;
  PROCEDURE plus(var1 in out int)
  AS
  BEGIN
    var1 = var1 + 1;
  END;
END pkg1;
/
CREATE PACKAGE BODY

-- Create the stored procedure dynamic_proc.
gaussdb=# CREATE OR REPLACE PROCEDURE dynamic_proc
AS
  staff_id     NUMBER(6) := 200;
  first_name   VARCHAR2(20);
  salary       NUMBER(8,2);
BEGIN
   -- Execute the anonymous block.
   EXECUTE IMMEDIATE 'begin select first_name, salary into :first_name, :salary from hr.staffs where
staff_id= :dno; end;'
      USING OUT first_name, OUT salary, IN staff_id;
  dbe_output.print_line(first_name|| ' ' || salary);
END;
/
CREATE PROCEDURE
-- Create a stored procedure to call overloaded functions.
gaussdb=# CREATE OR REPLACE PROCEDURE dynamic_proc1
AS
    v_sql  VARCHAR2(200);
    var1   NUMBER(6) := 1;
    var2   NUMBER(6) := 2;
    var3   NUMBER(6);
BEGIN
    v_sql := 'begin pkg1.plus(:1, :2, :3); end;';
    EXECUTE IMMEDIATE v_sql USING var1, var2, out var3;
    dbe_output.print_line('var3: ' || var3);
END;
/
CREATE PROCEDURE

-- Call the stored procedure.
gaussdb=# CALL dynamic_proc();
mike 5800.00
 dynamic_proc
--------------

(1 row)
gaussdb=# CALL dynamic_proc1();
var3: 4
 dynamic_proc1
---------------

(1 row)
-- Drop the stored procedure.
gaussdb=# DROP PROCEDURE dynamic_proc;
DROP PROCEDURE
gaussdb=# DROP PROCEDURE dynamic_proc1;
DROP PROCEDURE
-- Example of an error reported when dynamic_sql_check is enabled
gaussdb=# SET behavior_compat_options = 'dynamic_sql_check';
SET
gaussdb=# CREATE OR REPLACE PROCEDURE test_proc_exception001(a out integer, b inout integer, c
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
integer)
as
BEGIN
 a := 1;
 begin b := 1/0; end;
EXCEPTION
 WHEN others THEN
  b := 2;
END;
/
CREATE PROCEDURE
gaussdb=#
DECLARE
 a integer := 1;
 c integer;
BEGIN
  execute immediate 'begin test_proc_exception001(:1,:2,:1); end;' using in out a, out c, a;
END;
/
ERROR:  The numbers of parameters do not match in Dynamic SQL, expected number of parameters : 3 ,
actual number of SQL parameters : 2.
CONTEXT:  PL/pgSQL function inline_code_block line 4 at EXECUTE statement
-- Change the placeholders with the same name.
gaussdb=#
DECLARE
 a integer := 1;
 c integer;
BEGIN
  execute immediate 'begin test_proc_exception001(:1,:2,:3); end;' using in out a, out c, a;
END;
/
ANONYMOUS BLOCK EXECUTE
gaussdb=# DROP PROCEDURE test_proc_exception001;
DROP PROCEDURE
```

2.8 Control Statements

2.8.1 RETURN Statements

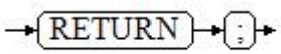
In GaussDB, data can be returned in either of the following ways: **RETURN** or **RETURN NEXT**/**RETURN QUERY**. **RETURN NEXT**/**RETURN QUERY** is used only for functions and cannot be used for stored procedures.

2.8.1.1 RETURN

Syntax

Figure 2-19 shows the syntax diagram for a RETURN statement.

Figure 2-19 return_clause::=



The above syntax diagram is explained as follows:

Assign the result returned by a stored procedure or function to the caller.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

See **Examples** for call statement examples.

2.8.1.2 RETURN NEXT/RETURN QUERY

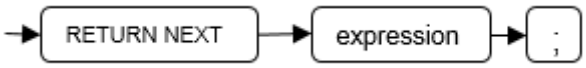
Syntax

When creating a function, specify **SETOF** *datatype* for the return values.

- return_next_clause::=

  Figure 2-20 shows the return_next_clause::= statement.

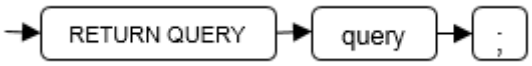
  Figure 2-20 return_next_clause::=

- return_query_clause::=

  Figure 2-21 shows the return_query_clause::= statement.

  Figure 2-21 return_query_clause::=

If a function needs to return a result set, use RETURN NEXT or RETURN QUERY to add results to the result set, and then continue to execute the next statement of the function. As the RETURN NEXT or RETURN QUERY statement is executed repeatedly, more and more results will be added to the result set. After the function is executed, all results are returned.

RETURN NEXT can be used for scalar and compound data types.

RETURN QUERY has a variant RETURN QUERY EXECUTE. You can add dynamic queries and add parameters to the queries by USING.

Examples

```
gaussdb=# DROP TABLE t1;
gaussdb=# CREATE TABLE t1(a int);
gaussdb=# INSERT INTO t1 VALUES(1),(10);

--RETURN NEXT
gaussdb=# CREATE OR REPLACE FUNCTION fun_for_return_next() RETURNS SETOF t1 AS $$
DECLARE
  r t1%ROWTYPE;
BEGIN
  FOR r IN select * from t1
  LOOP
    RETURN NEXT r;
  END LOOP;
  RETURN;
END;
$$ LANGUAGE plpgsql;
gaussdb=# call fun_for_return_next();
 a
---
 1
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 10
(2 rows)
-- RETURN QUERY
gaussdb=# CREATE OR REPLACE FUNCTION fun_for_return_query() RETURNS SETOF t1 AS $$
DECLARE
  r t1%ROWTYPE;
BEGIN
  RETURN QUERY select * from t1;
END;
$$
language plpgsql;
gaussdb=# call fun_for_return_query();
 a
---
 1
 10
(2 rows)
```

2.8.2 Conditional Statements

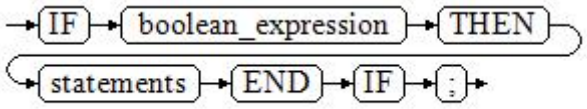
Conditional statements are used to decide whether given conditions are met. Operations are executed based on the decisions made.

GaussDB supports five usages of IF:

- IF_THEN

  Figure 2-22 shows the IF_THEN::= statement.

  Figure 2-22 IF_THEN::=

  IF_THEN is the simplest form of IF. If the condition is true, statements are executed. If it is false, they are skipped.

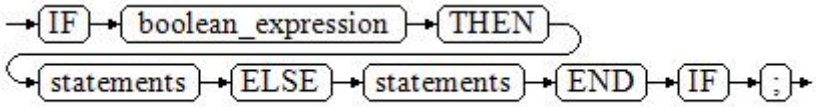
  Example:

  ```
  gaussdb=# DECLARE
      v_user_id integer default 1;
  BEGIN
  IF v_user_id <> 0 THEN
      raise info 'v_user_id is NOT 0';
  END IF;
  END;
  /
  INFO:  v_user_id is NOT 0
  ```

- IF_THEN_ELSE

  Figure 2-23 shows the IF_THEN_ELSE::= statement.

  Figure 2-23 IF_THEN_ELSE::=

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

IF_THEN_ELSE has an ELSE branch and can be executed if the condition is false.

Example:

```
gaussdb=# DECLARE
    v_user_id integer default 0;
BEGIN
    IF v_user_id <> 0 THEN
        raise info 'v_user_id is NOT 0';
    ELSE
        raise info 'v_user_id is 0';
    END IF;
END;
/
INFO:  v_user_id is 0
```

- IF_THEN_ELSE IF

  IF statements can be nested in the following way:

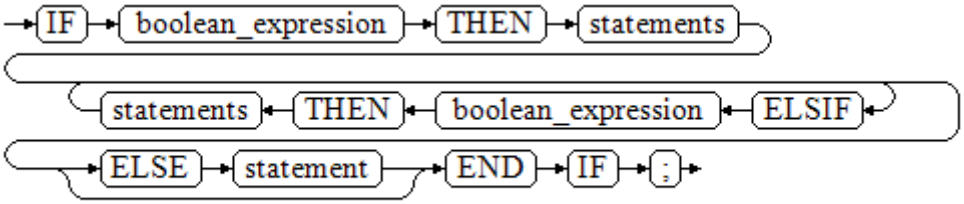
```
gaussdb=# DECLARE
    v_user_id integer default 1;
BEGIN
    IF v_user_id = 0 THEN
        raise info 'v_user_id is  0';
    ELSE
        IF v_user_id > 0 THEN
            raise info 'v_user_id > 0';
        END IF;
    END IF;
END;
/
INFO:  v_user_id > 0
```

  Actually, this is a way of an IF statement nesting in the ELSE part of another IF statement. Therefore, an END IF statement is required for each nesting IF statement and another END IF statement is required to end the parent IF-ELSE statement. To set multiple options, use the following form:

- IF_THEN_ELSIF_ELSE

  Figure 2-24 shows the IF_THEN_ELSIF_ELSE::= statement.

  Figure 2-24 IF_THEN_ELSIF_ELSE::=



  Example:

```
gaussdb=# DECLARE
    v_user_id integer default NULL;
BEGIN
    IF v_user_id = 0 THEN
        raise info 'v_user_id is  0';
    ELSIF v_user_id > 0 THEN
        raise info 'v_user_id > 0';
    ELSIF v_user_id < 0 THEN
        raise info 'v_user_id < 0';
    ELSE
        raise info 'v_user_id is NULL';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    END IF;
END;
/
INFO:  v_user_id is NULL
```

- IF_THEN_ELSEIF_ELSE

  ELSEIF is an alias of ELSIF.

  Example:

```
gaussdb=# CREATE OR REPLACE PROCEDURE proc_control_structure(i in integer)
AS
    BEGIN
        IF i > 0 THEN
            raise info 'i:% is greater than 0. ',i;
        ELSIF i < 0 THEN
            raise info 'i:% is smaller than 0. ',i;
        ELSE
            raise info 'i:% is equal to 0. ',i;
        END IF;
        RETURN;
    END;
/
CREATE PROCEDURE

gaussdb=# CALL proc_control_structure(3);
INFO:  i:3 is greater than 0.
 proc_control_structure
------------------------

(1 row)

-- Drop a stored procedure.
gaussdb=# DROP PROCEDURE proc_control_structure;
DROP PROCEDURE
```

2.8.3 Loop Statements

Simple Loop Statements

Syntax diagram

Figure 2-25 shows the loop::= statement.

Figure 2-25 loop::=

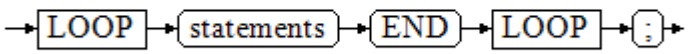
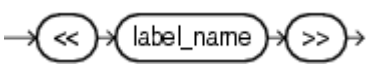


Figure 2-26 shows the label declaration ::= statement.

Figure 2-26 label declaration ::=



Example:

```
gaussdb=# CREATE OR REPLACE PROCEDURE proc_loop(i in integer, count out integer)
AS
```

```
    BEGIN
        count:=0;
        LOOP
        IF count > i THEN
            raise info 'count is %. ', count;
            EXIT;
        ELSE
            count:=count+1;
        END IF;
        END LOOP;
    END;
/
CREATE PROCEDURE

gaussdb=# CALL proc_loop(10,5);
INFO:  count is 11.
 count
-------
    11
(1 row)
```

NOTICE

The loop must be exploited together with EXIT; otherwise, a dead loop occurs.

WHILE_LOOP

Syntax diagram

Figure 2-27 shows the while_loop::= statement.

Figure 2-27 while_loop::=

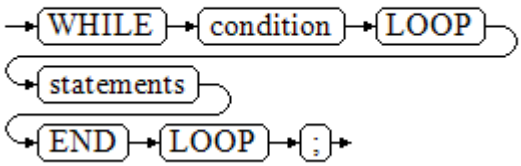
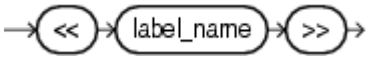


Figure 2-28 shows the label declaration ::= statement.

Figure 2-28 label declaration ::=



If the conditional expression is true, a series of statements in the WHILE statement are repeatedly executed and the condition is decided each time the loop body is executed.

Example:

```
gaussdb=# CREATE TABLE integertable(c1 integer) DISTRIBUTE BY hash(c1);
CREATE TABLE
gaussdb=# CREATE OR REPLACE PROCEDURE proc_while_loop(maxval in integer)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
AS
    DECLARE
    i int :=1;
    BEGIN
        WHILE i < maxval LOOP
            INSERT INTO integertable VALUES(i);
            i:=i+1;
        END LOOP;
    END;
/
CREATE PROCEDURE

-- Call a stored procedure.
gaussdb=# CALL proc_while_loop(10);
 proc_while_loop
-----------------

(1 row)

-- Drop a stored procedure and a table.
gaussdb=# DROP PROCEDURE proc_while_loop;
DROP PROCEDURE

gaussdb=# DROP TABLE integertable;
DROP TABLE
```

FOR_LOOP (*Integer variable*) *Statement*

Syntax diagram

Figure 2-29 shows the for_loop::= statement.

Figure 2-29 for_loop::=

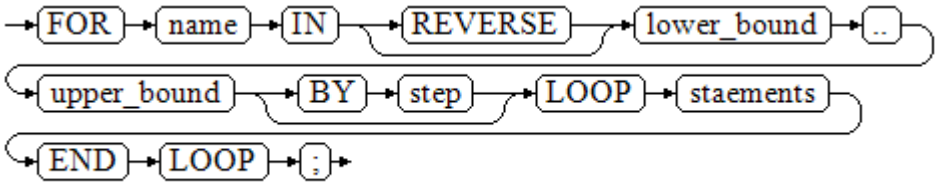
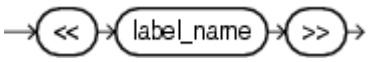


Figure 2-30 shows the label declaration ::= statement.

Figure 2-30 label declaration ::=



NOTE

- The variable *name* is automatically defined as the integer type and exists only in this loop. The variable name falls between lower_bound and upper_bound.
- When the keyword REVERSE is used, the lower bound must be greater than or equal to the upper bound; otherwise, the loop body is not executed.
- When FORALL is used together with BULK COLLECT INTO, the INTO variable returns only the execution result of a single DML statement.

Example:

```
-- Loop from 0 to 5.
gaussdb=# CREATE OR REPLACE PROCEDURE proc_for_loop()
```

```
AS
    BEGIN
    FOR I IN 0..5 LOOP
        DBE_OUTPUT.PRINT_LINE('It is '||to_char(I) || ' time;') ;
    END LOOP;
END;
/
CREATE PROCEDURE

-- Call a stored procedure.
gaussdb=# CALL proc_for_loop();
It is 0 time;
It is 1 time;
It is 2 time;
It is 3 time;
It is 4 time;
It is 5 time;
 proc_for_loop
---------------

(1 row)

-- Drop a stored procedure.
gaussdb=# DROP PROCEDURE proc_for_loop;
DROP PROCEDURE
```

FOR_LOOP

Syntax diagram

Figure 2-31 shows the for_loop_query::= statement.

Figure 2-31 for_loop_query::=

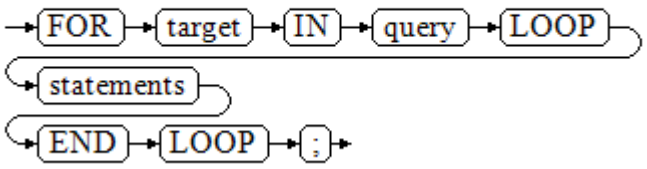
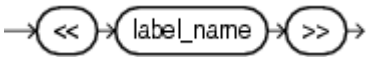


Figure 2-32 shows the label declaration ::= statement.

Figure 2-32 label declaration ::=



NOTE

- The variable *target* is automatically defined, its type is the same as that in the query result, and it is valid only in this loop. The target value is the query result.
- You can use EXECUTE to add a dynamic query and use USING to insert parameters to the query. For details, see the dynamic query content in **Example**.

Example:

```
-- Display the query result from the loop.
gaussdb=# CREATE OR REPLACE PROCEDURE proc_for_loop_query()
```

```
AS
    record VARCHAR2(50);
BEGIN
    FOR record IN SELECT spcname FROM pg_tablespace LOOP
    dbe_output.print_line(record);
    END LOOP;
END;
/
CREATE PROCEDURE

-- Call a stored procedure.
gaussdb=# CALL proc_for_loop_query();
pg_default
pg_global
 proc_for_loop_query
---------------------

(1 row)

-- Drop a stored procedure.
gaussdb=# DROP PROCEDURE proc_for_loop_query;
DROP PROCEDURE

-- Perform dynamic query.
gaussdb=# CREATE TABLE t1(id int);
gaussdb=# INSERT INTO t1 VALUES(1);
gaussdb=#
DECLARE
a int := 1;
item t1%rowtype;
BEGIN
    FOR item IN EXECUTE 'SELECT * FROM t1 WHERE id = :1' USING a LOOP
        RAISE INFO '%', item;
    END LOOP;
END;
/
INFO:  (1)
ANONYMOUS BLOCK EXECUTE
gaussdb=# DROP TABLE t1;
```

FORALL

Syntax diagram

Figure 2-33 shows the forall::= statement.

Figure 2-33 forall::=

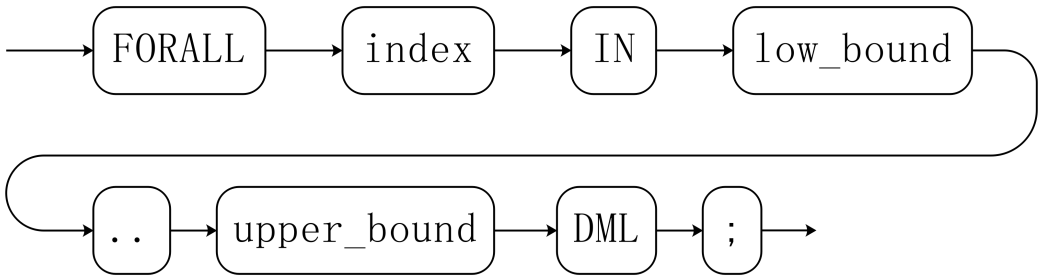
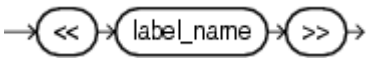


Figure 2-34 shows the label declaration:: statement.

Figure 2-34 label declaration ::=



 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The variable *index* is automatically defined as the integer type and exists only in this loop. The index value falls between low_bound and upper_bound.
- The FORALL loop body can contain only DML statements.

Example:

```
gaussdb=# CREATE TABLE TEST_t1 (
  title NUMBER(6),
  did VARCHAR2(20),
  data_period VARCHAR2(25),
  kind VARCHAR2(25),
  interval VARCHAR2(20),
  time DATE,
  isModified VARCHAR2(10)
)
DISTRIBUTE BY hash(did);
CREATE TABLE

gaussdb=# INSERT INTO TEST_t1 VALUES( 8, 'Donald', 'OConnell', 'DOCONNEL', '650.507.9833',
to_date('21-06-1999', 'dd-mm-yyyy'), 'SH_CLERK' );
INSERT 0 1
gaussdb=# CREATE OR REPLACE PROCEDURE proc_forall()
AS
BEGIN
    FORALL i IN 100..120
        update TEST_t1 set title = title + 100*i;
END;
/
CREATE PROCEDURE

-- Call a stored procedure.
gaussdb=# CALL proc_forall();
 proc_forall
-------------

(1 row)

-- Query the calling result of the stored procedure.
gaussdb=# SELECT * FROM TEST_t1;
 title  |  did   | data_period |   kind   |   interval   |        time         | ismodified
--------+--------+-------------+----------+--------------+---------------------+------------
 231008 | Donald | OConnell    | DOCONNEL | 650.507.9833 | 1999-06-21 00:00:00 | SH_CLERK
(1 row)

-- Drop a stored procedure and a table.
gaussdb=# DROP PROCEDURE proc_forall;
DROP PROCEDURE

gaussdb=# DROP TABLE TEST_t1;
DROP TABLE
```

2.8.4 Branching Statements

Syntax

Figure 2-35 shows the syntax diagram for a branching statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 2-35 case_when::=

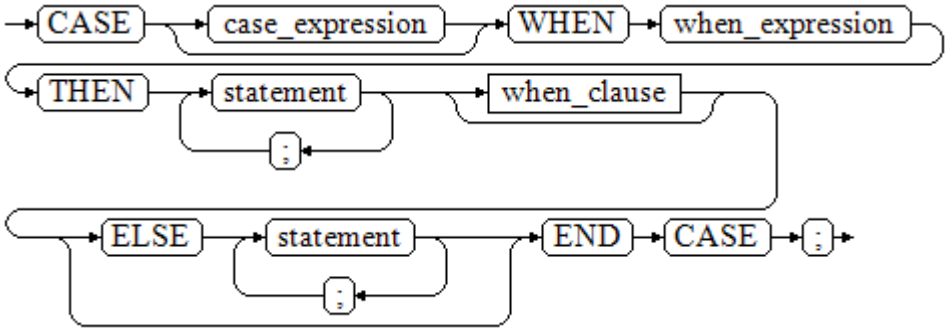
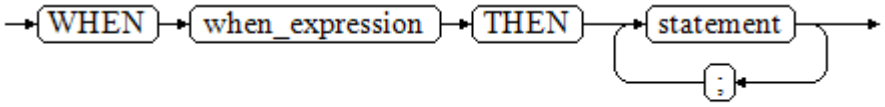


Figure 2-36 shows the syntax diagram for the WHEN clause.

Figure 2-36 when_clause::=



Parameters:

- *case_expression*: specifies a variable or an expression.
- *when_expression*: specifies a constant or conditional expression.
- *statement*: specifies the statement to be executed.

Examples

```
gaussdb=# CREATE OR REPLACE PROCEDURE proc_case_branch(pi_result in integer, pi_return out integer)
AS
   BEGIN
      CASE pi_result
         WHEN 1 THEN
            pi_return := 111;
         WHEN 2 THEN
            pi_return := 222;
         WHEN 3 THEN
            pi_return := 333;
         WHEN 6 THEN
            pi_return := 444;
         WHEN 7 THEN
            pi_return := 555;
         WHEN 8 THEN
            pi_return := 666;
         WHEN 9 THEN
            pi_return := 777;
         WHEN 10 THEN
            pi_return := 888;
         ELSE
            pi_return := 999;
      END CASE;
      raise info 'pi_return : %',pi_return ;
END;
/
CREATE PROCEDURE

gaussdb=# CALL proc_case_branch(3,0);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
INFO:  pi_return : 333
 pi_return
-----------
       333
(1 row)

-- Drop a stored procedure.
gaussdb=# DROP PROCEDURE proc_case_branch;
DROP PROCEDURE
```

2.8.5 NULL Statements

In PL/SQL programs, NULL statements are used to indicate "nothing should be done", equal to placeholders. They grant meanings to some statements and improve program readability.

Syntax

The following shows example use of NULL statements:

```
DECLARE
   ...
BEGIN
   ...
   IF v_num IS NULL THEN
      NULL; -- No data needs to be processed.
   END IF;
END;
/
```

Parameters:

- *v_num*: specifies a variable or expression.

Example

```
gaussdb=# DECLARE
   v_num integer default NULL;
BEGIN
   IF v_num IS NOT NULL THEN
      raise info 'v_num is NULL';
   ELSE
      NULL; -- No data needs to be processed.
   END IF;
END;
/
ANONYMOUS BLOCK EXECUTE
```

2.8.6 Error Trapping Statements

By default, any error occurring in a PL/SQL function aborts execution of the function, and indeed of the surrounding transaction as well. You can trap errors and restore from them by using a BEGIN block with an EXCEPTION clause. The syntax is an extension of the normal syntax for a BEGIN block:

```
[<<label>>]
[DECLARE
   declarations]
BEGIN
   statements
EXCEPTION
   WHEN condition [OR condition ...] THEN
      handler_statements
   [WHEN condition [OR condition ...] THEN
      handler_statements
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    ...]
END;
```

If no error occurs, this form of block simply executes all the statements, and then control passes to the next statement after END. But if an error occurs within the statements, further processing of the statements is abandoned and the statements will be rolled back, and then the control passes to the EXCEPTION list. The list is searched for the first condition matching the error that occurred. If a match is found, the corresponding **handler_statements** are executed, and then the control passes to the next statement after END. If no match is found, the error propagates out as if the EXCEPTION clause was not there at all: Error codes can be used to catch other error codes of the same type.

The error can be caught by an enclosing block with EXCEPTION, or if there is none it aborts processing of the function.

The condition name can be any value listed as an SQL standard error code description. The special condition name **OTHERS** matches every error type except **QUERY_CANCELED**.

If a new error occurs within the selected **handler_statements**, it cannot be caught by this EXCEPTION clause, but is propagated out. A surrounding EXCEPTION clause could catch it.

When an error is caught by an EXCEPTION clause, the local variables of the PL/SQL function remain as they were when the error occurred, but all changes to persistent database state within the block are rolled back.

Example:

```
gaussdb=# CREATE TABLE mytab(id INT,firstname VARCHAR(20),lastname VARCHAR(20)) DISTRIBUTE BY
hash(id);
CREATE TABLE

gaussdb=# INSERT INTO mytab(firstname, lastname) VALUES('Tom', 'Jones');
INSERT 0 1

gaussdb=# CREATE FUNCTION fun_exp() RETURNS INT
AS $$
DECLARE
    x INT :=0;
    y INT;
BEGIN
    UPDATE mytab SET firstname = 'Joe' WHERE lastname = 'Jones';
    x := x + 1;
    y := x / 0;
EXCEPTION
    WHEN division_by_zero THEN
        RAISE NOTICE 'caught division_by_zero';
        RETURN x;
END;$$
LANGUAGE plpgsql;
CREATE FUNCTION

gaussdb=# call fun_exp();
NOTICE:  caught division_by_zero
 fun_exp
---------
       1
(1 row)

gaussdb=# select * from mytab;
 id | firstname | lastname
----+-----------+----------
    | Tom       | Jones
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)

gaussdb=# DROP FUNCTION fun_exp();
DROP FUNCTION

gaussdb=# DROP TABLE mytab;
DROP TABLE
```

When the control reaches the assignment to **y**, it will fail with a **division_by_zero** error. This will be caught by the EXCEPTION clause. The value returned from the RETURN statement will be the incremented value of **x**.

NOTE

- A block containing an EXCEPTION clause is more expensive to enter and exit than a block without one. Therefore, do not use EXCEPTION unless necessary.
- The following exceptions cannot be captured and the entire stored procedure is rolled back: 1. The node is faulty. 2. The thread of the node involved in the stored procedure exits due to a network fault. In addition, the definition of the source table is inconsistent with that of the target table in the COPY FROM operation. 3. An exception occurs during subtransaction clearing. (For a stored procedure that contains the EXCEPTION statement, an implicit subtransaction is started when the statement is executed. After the statement is executed, the subtransaction is automatically cleared. During the clearing, an exception may occur due to memory control.)

Example: UPDATE/INSERT exception

This example explains how to perform either UPDATE or INSERT appropriately with exception handler:

```
gaussdb=# CREATE TABLE db (a INT, b TEXT);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# CREATE FUNCTION merge_db(key INT, data TEXT) RETURNS VOID AS
$$
BEGIN
    LOOP

-- First try to update the key
        UPDATE db SET b = data WHERE a = key;
        IF found THEN
            RETURN;
        END IF;
-- The key does not exist. Therefore, try to insert one. If someone else inserts the same key concurrently, the
unique key may fail to be obtained.
        BEGIN
            INSERT INTO db(a,b) VALUES (key, data);
            RETURN;
        EXCEPTION WHEN unique_violation THEN
        -- Do nothing, and loop to try the UPDATE operation again.
        END;
    END LOOP;
END;
$$
LANGUAGE plpgsql;
CREATE FUNCTION

gaussdb=# SELECT merge_db(1, 'david');
 merge_db
----------

(1 row)

gaussdb=# SELECT merge_db(1, 'dennis');
 merge_db
----------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)

-- Drop the function and table.
gaussdb=# DROP FUNCTION merge_db;
DROP FUNCTION

gaussdb=# DROP TABLE db ;
DROP TABLE
```

In ORA-compatible mode, if the **'on(PLSQL_RETURN_EXCEPTION)'** option is enabled for the GUC parameter **exec_behavior_knob**, the error returned during statement type conversion is captured by the EXCEPTION clause. If the option is disabled, the error is not captured by the EXCEPTION clause. For details, see **exec_behavior_knob**.

```
gaussdb=# CREATE OR REPLACE FUNCTION func RETURN INT AS
BEGIN
    RETURN 'abc';
    EXCEPTION WHEN OTHERS THEN
    RETURN 123;
END;
/
CREATE FUNCTION
gaussdb=# set exec_behavior_knob='on(PLSQL_RETURN_EXCEPTION)';
SET
gaussdb=# SELECT func();
 func
------
  123
(1 row)
gaussdb=# set exec_behavior_knob='oFF(PLSQL_RETURN_EXCEPTION)';
SET
gaussdb=# SELECT func();
ERROR:  invalid input syntax for integer: "abc"
CONTEXT:  PL/pgSQL function func() while casting return value to function's return type
referenced column: func
gaussdb=# DROP FUNCTION IF EXISTS func;
DROP FUNCTION
```

2.8.6.1 Obtaining Exception Information

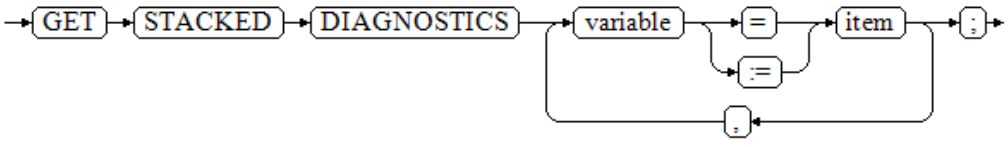
You can use the GET STACKED DIAGNOSTICS statement to query the exception information during the execution of the current PL/SQL function.

Syntax

```
GET STACKED DIAGNOSTICS {{variable { = | := } item }[ , ... ]};
```

Figure 2-37 shows the syntax diagram.

Figure 2-37 Syntax for obtaining error information



In the preceding information, **item** indicates the diagnosis item dedicated to exceptions. Currently, the following diagnosis items are available:

- **PG_EXCEPTION_DETAIL**: error details.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **PG_EXCEPTION_HINT**: error prompt.
- **PG_EXCEPTION_CONTEXT**: call stack information about the exception.
- **MESSAGE_TEXT**: main error information.
- **RETURNED_SQLSTATE**: SQLSTATE error code of the exception.

Examples

```
gaussdb=# CREATE TABLE diag_table(int_col int, str_col varchar2);
CREATE TABLE
gaussdb=# CREATE OR REPLACE PROCEDURE get_diag_proc() AS
DECLARE
    sql_state text;
    message_text text;
    pg_exception_detail text;
    pg_exception_hint text;
    pg_exception_context text;
BEGIN
    UPDATE diag_table set int_col = 'aaa';
    EXCEPTION WHEN others THEN
        GET STACKED DIAGNOSTICS sql_state = returned_sqlstate;
        GET STACKED DIAGNOSTICS message_text = message_text;
        GET STACKED DIAGNOSTICS pg_exception_detail = pg_exception_detail;
        GET STACKED DIAGNOSTICS pg_exception_hint = pg_exception_hint;
        GET STACKED DIAGNOSTICS pg_exception_context = pg_exception_context;
        raise info 'sql_state: %, message_text: %', sql_state, message_text;
END;
/
CREATE PROCEDURE
gaussdb=# CALL get_diag_proc();
INFO:  sql_state: 22P02, message_text: invalid input syntax for integer: "aaa"
 get_diag_proc
---------------

(1 row)

gaussdb=# DROP TABLE diag_table;
DROP TABLE
gaussdb=# DROP PROCEDURE get_diag_proc();
DROP PROCEDURE
```

2.8.6.2 Obtaining Current Execution Information

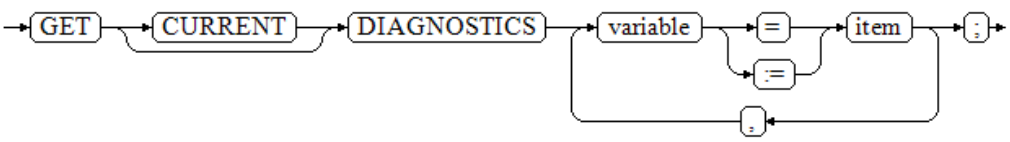
You can use the GET [CURRENT] DIAGNOSTICS statement to query the status information during the execution of the current PL/SQL function.

Syntax

```
GET [ CURRENT ] DIAGNOSTICS {{variable { = | := } item }[ , ... ]};
```

Figure 2-38 shows the syntax diagram.

Figure 2-38 Syntax for obtaining current execution information



In the preceding information, **item** indicates the diagnosis item in the execute state. Currently, the following diagnosis items are available:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **ROW_COUNT**: number of rows processed by the previous SQL statement.
- **RESULT_OID**: OID of the previous INSERT statement.

Examples

```
gaussdb=# CREATE TABLE diag_table(int_col int, str_col varchar2);
CREATE TABLE
gaussdb=# CREATE OR REPLACE PROCEDURE get_diag_proc() AS
DECLARE
    nrows INT;
    result_oid oid;
BEGIN
    INSERT INTO diag_table values(1, 'aaa');
    GET CURRENT DIAGNOSTICS nrows = row_count;
    GET CURRENT DIAGNOSTICS result_oid = result_oid;
    raise info 'row_count: %, result_oid: %', nrows, result_oid;
END;
/
CREATE PROCEDURE
gaussdb=# CALL get_diag_proc();
INFO:  row_count: 1, result_oid: 0
 get_diag_proc
---------------

(1 row)

gaussdb=# DROP TABLE diag_table;
DROP TABLE
gaussdb=# DROP PROCEDURE get_diag_proc();
DROP PROCEDURE
```

2.8.7 GOTO Statements

A GOTO statement unconditionally transfers the control from the current statement to a labeled statement. The GOTO statement changes the execution logic. Therefore, use this statement only when necessary. Alternatively, you can use the EXCEPTION statement to handle issues in special scenarios. To execute a GOTO statement, the labeled statement must be unique.

Syntax

Figure 2-39 shows the label declaration ::= statement.

Figure 2-39 label declaration ::

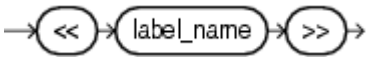


Figure 2-40 shows the goto statement ::= statement.

Figure 2-40 goto statement ::



Examples

```
gaussdb=# CREATE OR REPLACE PROCEDURE GOTO_test()
AS
DECLARE
   v1  int;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
BEGIN
   v1  := 0;
      LOOP
      EXIT WHEN v1 > 100;
            v1 := v1 + 2;
            if v1 > 25 THEN
                  GOTO pos1;
            END IF;
      END LOOP;
<<pos1>>
v1 := v1 + 10;
raise info 'v1 is %. ', v1;
END;
/
CREATE PROCEDURE
gaussdb=# call GOTO_test();
INFO:  v1 is 36.
 goto_test
-----------

(1 row)
```

Limitations

Using GOTO statements has the following limitations:

- Do not use multiple GOTO statements with the same label even if the statements are in different blocks.

  ```
  BEGIN
    GOTO pos1;
    <<pos1>>
    SELECT * FROM ...
    <<pos1>>
    UPDATE t1 SET ...
  END;
  ```

- A GOTO statement cannot transfer control to the IF, CASE, or LOOP statement.

  ```
  BEGIN
    GOTO pos1;
    IF valid THEN
      <<pos1>>
      SELECT * FROM ...
    END IF;
  END;
  ```

- A GOTO statement cannot transfer control from one IF clause to another, or from one WHEN clause in the CASE statement to another.

  ```
  BEGIN
    IF valid THEN
      GOTO pos1;
      SELECT * FROM ...
    ELSE
      <<pos1>>
      UPDATE t1 SET ...
    END IF;
  END;
  ```

- Transferring control from an outer block to an inner BEGIN-END block is not supported.

  ```
  BEGIN
    GOTO pos1;
    BEGIN
      <<pos1>>
      UPDATE t1 SET ...
    END;
  END;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Transferring control from an exception handler to the current BEGIN-END block is not supported. However, transferring control to the upper-layer BEGIN-END block is supported.

```
BEGIN
  <<pos1>>
  UPDATE t1 SET ...
  EXCEPTION
    WHEN condition THEN
      GOTO pos1;
END;
```

- To branch to a position that does not have an executable statement, you need to add the NULL statement.

```
DECLARE
  done  BOOLEAN;
BEGIN
  FOR i IN 1..50 LOOP
    IF done THEN
      GOTO end_loop;
    END IF;
    <<end_loop>>  -- You cannot run the GOTO statement to go here unless there is an executable
statement following it.
    NULL; -- The statement added here is used to avoid errors.
  END LOOP;  -- If the previous sentence NULL does not exist, an error will be reported.
END;
/
```

2.9 Transaction Statements

A stored procedure itself is automatically in a transaction. A transaction is automatically started when the outmost stored procedure is called. In addition, the transaction is automatically committed when the calling ends, or is rolled back when an exception occurs during calling. In addition to automatic transaction control, you can also use COMMIT/ROLLBACK to control transactions in stored procedures. Running the COMMIT/ROLLBACK commands in a stored procedure will commit or roll back the current transaction and automatically starts a new transaction. All subsequent operations will be performed in the new transaction.

A savepoint is a special mark inside a transaction. It allows all commands that are executed after it was established to be rolled back, restoring the transaction state to what it was at the time of the savepoint. In a stored procedure, you can use savepoints to manage transactions. Currently, you can create, roll back, and release savepoints. If a savepoint for rollback is used in a stored procedure, only the modification of the current transaction is rolled back. The execution process of the stored procedure is not changed, and the values of local variables in the stored procedure are not rolled back.

Syntax

```
Define a savepoint.
   SAVEPOINT savepoint_name;
Roll back a savepoint.
   ROLLBACK TO [SAVEPOINT] savepoint_name;
Release a savepoint.
   RELEASE [SAVEPOINT] savepoint_name;
```

Application Scenarios

The applicable contexts are as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- COMMIT, ROLLBACK, and SAVEPOINT can be used in PL/SQL stored procedures/functions.
- COMMIT, ROLLBACK, and SAVEPOINT can be used in stored procedures and functions that contain EXCEPTION.
- COMMIT, ROLLBACK, and SAVEPOINT can be used in EXCEPTION statements of stored procedures.
- A stored procedure that contains COMMIT, ROLLBACK, or SAVEPOINT (which means the stored procedure is controlled by BEGIN, START, or END) can be called in a transaction block.
- A stored procedure that contains savepoints can be called in a subtransaction and an externally defined savepoint is used to roll back the transaction to the savepoint defined outside the stored procedure.
- A savepoint defined in the stored procedure can be viewed outside the stored procedure. That is, the modification of the transaction can be rolled back to the savepoint defined in the stored procedure.
- COMMIT, ROLLBACK, and SAVEPOINT, as well as IF, FOR, CURSOR LOOP, and WHILE, can be called in most PL/SQL contexts and statements.

The following content can be committed or rolled back:

- DDL statements after COMMIT or ROLLBACK can be committed or rolled back.
- DML statements after COMMIT or ROLLBACK can be committed.
- GUC parameters in stored procedures can be committed or rolled back.

Limitations

The following contexts are not applicable:

- COMMIT, ROLLBACK, and SAVEPOINT cannot be called in stored procedures other than PL/SQL, such as PL/Python.
- After SAVEPOINT is called in a transaction block, stored procedures that contain COMMIT/ROLLBACK cannot be called.
- Stored procedures that contain COMMIT, ROLLBACK, or SAVEPOINT cannot be called in TRIGGER.
- COMMIT, ROLLBACK, and SAVEPOINT cannot be called in EXECUTE statements.
- Stored procedures that contain COMMIT, ROLLBACK, or SAVEPOINT cannot be called in CURSOR statements.
- Stored procedures that contain IMMUTABLE or SHIPPABLE cannot call COMMIT, ROLLBACK, SAVEPOINT or another stored procedure that contains COMMIT, ROLLBACK, or SAVEPOINT.
- Stored procedures that contain COMMIT, ROLLBACK, or SAVEPOINT cannot be called in SQL statements other than SELECT PROC and CALL PROC.
- COMMIT, ROLLBACK, or SAVEPOINT cannot be called in a stored procedure whose header contains GUC parameters.
- COMMIT, ROLLBACK, or SAVEPOINT cannot be called in expressions or CURSOR and EXECUTE statements.
- Stored procedures that contain COMMIT, ROLLBACK, or SAVEPOINT cannot be called in the return values and expression calculation of stored procedures.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Savepoints defined outside a stored procedure cannot be released in the stored procedure.
- A stored procedure transaction and its autonomous transaction are two independent transactions and cannot use savepoints defined in each other's transaction.
- Advanced packages cannot call stored procedures that contain COMMIT, ROLLBACK, or SAVEPOINT statements through DBE_SQL.

The following content cannot be committed or rolled back:

- Variables declared or imported in stored procedures cannot be committed or rolled back.
- In stored procedures, GUC parameters that take effect only after a restart cannot be committed or rolled back.

NOTE

Executing the **COMMIT** or **ROLLBACK** statement in a stored procedure, function, or anonymous block does not update the statistics records in the PG_STAT-related system catalogs immediately. The statistics are updated only after the outermost transaction is complete. Therefore, when a large amount of data is updated through stored procedures, functions, or anonymous blocks, related statistics may not be updated in a timely manner.

Examples

- Example 1: COMMIT and ROLLBACK can be used in PL/SQL stored procedures. Subsequent examples depend on this case.

```
gaussdb=# DROP TABLE IF EXISTS EXAMPLE1;
gaussdb=# CREATE TABLE EXAMPLE1(COL1 INT);

gaussdb=# CREATE OR REPLACE PROCEDURE TRANSACTION_EXAMPLE()
AS
BEGIN
    FOR i IN 0..20 LOOP
        INSERT INTO EXAMPLE1(COL1) VALUES (i);
        IF i % 2 = 0 THEN
            COMMIT;
        ELSE
            ROLLBACK;
        END IF;
    END LOOP;
END;
/
gaussdb=# call TRANSACTION_EXAMPLE();
 transaction_example
---------------------

(1 row)
```

- Example 2:

  COMMIT and ROLLBACK can be used in stored procedures that contain EXCEPTION.

  COMMIT and ROLLBACK can be used in EXCEPTION statements of stored procedures.

  DDL statements after COMMIT or ROLLBACK can be committed or rolled back.

```
gaussdb=# DROP TABLE IF EXISTS EXAMPLE1;
gaussdb=# CREATE TABLE EXAMPLE1(COL1 INT);
gaussdb=# CREATE OR REPLACE PROCEDURE TEST_COMMIT_INSERT_EXCEPTION_ROLLBACK()
AS
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
BEGIN
 DROP TABLE IF EXISTS TEST_COMMIT;
 CREATE TABLE TEST_COMMIT(A INT, B INT);
 INSERT INTO TEST_COMMIT SELECT 1, 1;
 COMMIT;
       CREATE TABLE TEST_ROLLBACK(A INT, B INT);
 RAISE EXCEPTION 'RAISE EXCEPTION AFTER COMMIT';
EXCEPTION
   WHEN OTHERS THEN
 INSERT INTO TEST_COMMIT SELECT 2, 2;
 ROLLBACK;
END;
/
gaussdb=# call TEST_COMMIT_INSERT_EXCEPTION_ROLLBACK();
NOTICE:  table "test_commit" does not exist, skipping
CONTEXT:  SQL statement "DROP TABLE IF EXISTS TEST_COMMIT"
PL/pgSQL function test_commit_insert_exception_rollback() line 3 at SQL statement
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CONTEXT:  SQL statement "CREATE TABLE TEST_COMMIT(A INT, B INT)"
PL/pgSQL function test_commit_insert_exception_rollback() line 4 at SQL statement
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CONTEXT:  SQL statement "CREATE TABLE TEST_ROLLBACK(A INT, B INT)"
PL/pgSQL function test_commit_insert_exception_rollback() line 7 at SQL statement
 test_commit_insert_exception_rollback
---------------------------------------

(1 row)
```

- Example 3: A stored procedure that contains COMMIT or ROLLBACK (which means the stored procedure is controlled by BEGIN, START, or END) can be called in a transaction block.

```
gaussdb=# BEGIN;
    -- For the definition of TEST_COMMIT_INSERT_EXCEPTION_ROLLBACK, see example 2.
    CALL TEST_COMMIT_INSERT_EXCEPTION_ROLLBACK();
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CONTEXT:  SQL statement "CREATE TABLE TEST_COMMIT(A INT, B INT)"
PL/pgSQL function test_commit_insert_exception_rollback() line 4 at SQL statement
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CONTEXT:  SQL statement "CREATE TABLE TEST_ROLLBACK(A INT, B INT)"
PL/pgSQL function test_commit_insert_exception_rollback() line 7 at SQL statement
 test_commit_insert_exception_rollback
---------------------------------------

(1 row)

gaussdb=# END;
COMMIT
```

- Example 4: COMMIT and ROLLBACK, as well as IF, FOR, CURSOR LOOP, and WHILE, can be called in most PL/SQL contexts and statements.

```
gaussdb=# CREATE OR REPLACE PROCEDURE TEST_COMMIT2()
IS
BEGIN
    DROP TABLE IF EXISTS TEST_COMMIT;
    CREATE TABLE TEST_COMMIT(A INT);
    FOR I IN REVERSE 3..0 LOOP
 INSERT INTO TEST_COMMIT SELECT I;
 COMMIT;
    END LOOP;
    FOR I IN REVERSE 2..4 LOOP
 UPDATE TEST_COMMIT SET A=I;
 COMMIT;
    END LOOP;
EXCEPTION
WHEN OTHERS THEN
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 INSERT INTO TEST_COMMIT SELECT 4;
    COMMIT;
END;
/
gaussdb=# call TEST_COMMIT2();
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CONTEXT:  SQL statement "CREATE TABLE TEST_COMMIT(A INT)"
PL/pgSQL function test_commit2() line 4 at SQL statement
 test_commit2
--------------

(1 row)
```

- Example 5: GUC parameters in stored procedures can be committed or rolled back.

```
gaussdb=# SHOW explain_perf_mode;
 explain_perf_mode
-------------------
 normal
(1 row)

gaussdb=# SHOW enable_force_vector_engine;
 enable_force_vector_engine
----------------------------
 off
(1 row)

gaussdb=# CREATE OR REPLACE PROCEDURE GUC_ROLLBACK()
AS
BEGIN
    SET enable_force_vector_engine = on;
    COMMIT;
    SET explain_perf_mode TO pretty;
    ROLLBACK;
END;
/

gaussdb=# call GUC_ROLLBACK();
 guc_rollback
--------------

(1 row)

gaussdb=# SHOW explain_perf_mode;
 explain_perf_mode
-------------------
 normal
(1 row)

gaussdb=# SHOW enable_force_vector_engine;
 enable_force_vector_engine
----------------------------
 on
(1 row)

gaussdb=# SET enable_force_vector_engine = off;
```

- Example 6: Savepoints can be used in a PL/SQL stored procedure to roll back some transaction modifications.

```
gaussdb=# DROP TABLE IF EXISTS EXAMPLE1;
gaussdb=# CREATE TABLE EXAMPLE1(COL1 INT);
gaussdb=# CREATE OR REPLACE PROCEDURE STP_SAVEPOINT_EXAMPLE1()
AS
BEGIN
    INSERT INTO EXAMPLE1 VALUES(1);
    SAVEPOINT s1;
    INSERT INTO EXAMPLE1 VALUES(2);
    ROLLBACK TO s1;  -- Roll back the insertion of record 2.
    INSERT INTO EXAMPLE1 VALUES(3);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
END;
/
gaussdb=# call STP_SAVEPOINT_EXAMPLE1();
 stp_savepoint_example1
------------------------

(1 row)
```

- Example 7: Savepoints in a PL/SQL stored procedure can be used for rollback to savepoints defined outside the stored procedure.

```
gaussdb=# DROP TABLE IF EXISTS EXAMPLE1;
gaussdb=# CREATE TABLE EXAMPLE1(COL1 INT);
gaussdb=# CREATE OR REPLACE PROCEDURE STP_SAVEPOINT_EXAMPLE2()
AS
BEGIN
    INSERT INTO EXAMPLE1 VALUES(2);
    ROLLBACK TO s1;  -- Roll back the insertion of record 2.
    INSERT INTO EXAMPLE1 VALUES(3);
END;
/

gaussdb=# BEGIN;
INSERT INTO EXAMPLE1 VALUES(1);
SAVEPOINT s1;
gaussdb=# CALL STP_SAVEPOINT_EXAMPLE2();
 stp_savepoint_example2
------------------------

(1 row)

gaussdb=# SELECT * FROM EXAMPLE1;
 col1
------
    0
    4
   10
   14
   16
   18
   20
    3
    3
    3
    2
    6
    8
   12
    1
    1
(16 rows)

gaussdb=# COMMIT;
COMMIT
```

- Example 8: Savepoints defined outside a PL/SQL stored procedure can be used for rollback to a savepoint in the stored procedure.

```
gaussdb=# DROP TABLE IF EXISTS EXAMPLE1;
gaussdb=# CREATE TABLE EXAMPLE1(COL1 INT);
gaussdb=# CREATE OR REPLACE PROCEDURE STP_SAVEPOINT_EXAMPLE3()
AS
BEGIN
    INSERT INTO EXAMPLE1 VALUES(1);
    SAVEPOINT s1;
    INSERT INTO EXAMPLE1 VALUES(2);
END;
/

gaussdb=# BEGIN;
gaussdb=# INSERT INTO EXAMPLE1 VALUES(3);
gaussdb=# CALL STP_SAVEPOINT_EXAMPLE3();
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 stp_savepoint_example3
------------------------

(1 row)

gaussdb=# ROLLBACK TO SAVEPOINT s1; -- Roll back the insertion of record 2 to the stored
procedure.
ROLLBACK
gaussdb=# SELECT * FROM EXAMPLE1;
 col1
------
    0
    4
   10
   14
   16
   18
   20
    3
    3
    3
    3
    2
    6
    8
   12
    1
    1
    1
(18 rows)

gaussdb=# COMMIT;
COMMIT
```

- Example 9: COMMIT and ROLLBACK can be called in functions.

```
gaussdb=# DROP TABLE IF EXISTS EXAMPLE1;
gaussdb=# CREATE TABLE EXAMPLE1(COL1 INT);
gaussdb=# CREATE OR REPLACE FUNCTION FUNCTION_EXAMPLE1() RETURN INT
AS
EXP INT;
BEGIN
    FOR i IN 0..20 LOOP
        INSERT INTO EXAMPLE1(col1) VALUES (i);
        IF i % 2 = 0 THEN
            COMMIT;
        ELSE
            ROLLBACK;
        END IF;
    END LOOP;
    SELECT COUNT(*) FROM EXAMPLE1 INTO EXP;
    RETURN EXP;
END;
/
gaussdb=# call FUNCTION_EXAMPLE1();
 function_example1
-------------------
                29
(1 row)
```

- Example 10: A stored procedure containing COMMIT or ROLLBACK cannot be triggered or called.

```
gaussdb=# DROP TABLE IF EXISTS EXAMPLE1;
gaussdb=# CREATE TABLE EXAMPLE1(COL1 INT);
gaussdb=# INSERT INTO EXAMPLE1(col1) VALUES (1);
gaussdb=# CREATE OR REPLACE FUNCTION FUNCTION_TRI_EXAMPLE2() RETURN TRIGGER
AS
EXP INT;
BEGIN
    FOR i IN 0..20 LOOP
        INSERT INTO EXAMPLE1(col1) VALUES (i);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    IF i % 2 = 0 THEN
        COMMIT;
    ELSE
        ROLLBACK;
    END IF;
  END LOOP;
  SELECT COUNT(*) FROM EXAMPLE1 INTO EXP;
END;
/

gaussdb=# CREATE TRIGGER TRIGGER_EXAMPLE AFTER DELETE ON EXAMPLE1
FOR EACH ROW EXECUTE PROCEDURE FUNCTION_TRI_EXAMPLE2();
CREATE TRIGGER
gaussdb=# DELETE FROM EXAMPLE1;
ERROR:  Can not commit/rollback if it's atomic is true: can not use commit rollback in Complex SQL
CONTEXT:  PL/pgSQL function function_tri_example2() line 7 at COMMIT
```

- Example 11: Stored procedures that contain IMMUTABLE or SHIPPABLE cannot call COMMIT, ROLLBACK, or another stored procedure that contains COMMIT or ROLLBACK.

```
gaussdb=# DROP TABLE IF EXISTS EXAMPLE1;
gaussdb=# CREATE TABLE EXAMPLE1(COL1 INT);
gaussdb=# CREATE OR REPLACE PROCEDURE TRANSACTION_EXAMPLE1()
IMMUTABLE
AS
EXP INT;
BEGIN
  FOR i IN 0..20 LOOP
    SELECT COUNT(*) FROM EXAMPLE1 INTO EXP;
    IF i % 2 = 0 THEN
      COMMIT;
    ELSE
      ROLLBACK;
    END IF;
  END LOOP;
END;
/
gaussdb=# CALL TRANSACTION_EXAMPLE1();
ERROR:  Can not commit/rollback if it's atomic is true: commit/rollback/savepoint is not allowed in a
non-volatile function
CONTEXT:  PL/pgSQL function transaction_example1() line 7 at COMMIT
```

- Example 12: Variables declared or specified in stored procedures cannot be committed.

```
gaussdb=# CREATE OR REPLACE PROCEDURE TRANSACTION_EXAMPLE2(EXP_OUT OUT INT)
AS
EXP INT;
BEGIN
  EXP_OUT := 0;
  COMMIT;
  DBE_OUTPUT.PRINT_LINE('EXP IS:'||EXP);
  EXP_OUT := 1;
  ROLLBACK;
  DBE_OUTPUT.PRINT_LINE('EXP IS:'||EXP);
END;
/
gaussdb=# CALL TRANSACTION_EXAMPLE2(100);
EXP IS:
EXP IS:
 exp_out
---------
       1
(1 row)
```

- Example 13: Calling in SQL statements (except SELECT PROCEDURE) is not supported.

```
gaussdb=# CREATE OR REPLACE PROCEDURE TRANSACTION_EXAMPLE3()
AS
BEGIN
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    FOR i IN 0..20 LOOP
        INSERT INTO EXAMPLE1 (col1) VALUES (i);
        IF i % 2 = 0 THEN
            EXECUTE IMMEDIATE 'COMMIT';
        ELSE
            EXECUTE IMMEDIATE 'ROLLBACK';
        END IF;
    END LOOP;
END;
/
gaussdb=# CALL TRANSACTION_EXAMPLE3();
ERROR:  cannot call transaction statements in EXECUTE IMMEDIATE statement.
CONTEXT:  PL/pgSQL function transaction_example3() line 6 at EXECUTE statement
```

- Example 14: COMMIT and ROLLBACK cannot be called in a stored procedure whose header contains GUC parameters.

```
gaussdb=# CREATE OR REPLACE PROCEDURE TRANSACTION_EXAMPLE4()
SET ARRAY_NULLS TO "ON"
AS
BEGIN
    FOR i IN 0..20 LOOP
        INSERT INTO EXAMPLE1 (col1) VALUES (i);
        IF i % 2 = 0 THEN
            COMMIT;
        ELSE
            ROLLBACK;
        END IF;
    END LOOP;
END;
/
gaussdb=# CALL TRANSACTION_EXAMPLE4();
ERROR:  Can not commit/rollback if it's atomic is true: transaction statement in store procedure with GUC setting in option clause is not supported
CONTEXT:  PL/pgSQL function transaction_example4() line 6 at COMMIT
```

- Example 15: A stored procedure whose cursor is open cannot contain COMMIT or ROLLBACK.

```
gaussdb=# CREATE OR REPLACE PROCEDURE TRANSACTION_EXAMPLE5(INTIN IN INT, INTOUT OUT INT)
AS
BEGIN
INTOUT := INTIN + 1;
COMMIT;
END;
/

gaussdb=# CREATE OR REPLACE PROCEDURE TRANSACTION_EXAMPLE6()
AS
CURSOR CURSOR1(EXPIN INT)
IS SELECT TRANSACTION_EXAMPLE5(EXPIN);
INTEXP INT;
BEGIN
    FOR i IN 0..20 LOOP
        OPEN CURSOR1(i);
        FETCH CURSOR1 INTO INTEXP;
        INSERT INTO EXAMPLE1(COL1) VALUES (INTEXP);
        IF i % 2 = 0 THEN
            COMMIT;
        ELSE
            ROLLBACK;
        END IF;
        CLOSE CURSOR1;
    END LOOP;
END;
/
gaussdb=# CALL TRANSACTION_EXAMPLE6();
ERROR:  Can not commit/rollback if it's atomic is true: transaction statement in store procedure used as cursor is not supported
CONTEXT:  PL/pgSQL function transaction_example5(integer) line 4 at COMMIT
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
referenced column: transaction_example5
PL/pgSQL function transaction_example6() line 8 at FETCH
```

- Example 16: COMMIT or ROLLBACK cannot be called in expressions or CURSOR and EXECUTE statements.

```
gaussdb=# CREATE OR REPLACE PROCEDURE exec_func1()
AS
BEGIN
    CREATE TABLE TEST_exec(A INT);
COMMIT;
END;
/
gaussdb=# CREATE OR REPLACE PROCEDURE exec_func2()
AS
BEGIN
EXECUTE exec_func1();
COMMIT;
END;
/
gaussdb=# CALL exec_func2();
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CONTEXT:  SQL statement "CREATE TABLE TEST_exec(A INT)"
PL/pgSQL function exec_func1() line 3 at SQL statement
PL/pgSQL function exec_func2() line 3 at EXECUTE statement
ERROR:  Can not commit/rollback if it's atomic is true: transaction statement in store procedure used as a expression is not supported
CONTEXT:  PL/pgSQL function exec_func1() line 4 at COMMIT
PL/pgSQL function exec_func2() line 3 at EXECUTE statement
```

- Example 17: Direct value returning from and expression calculation by stored procedures are not supported.

```
gaussdb=# CREATE OR REPLACE PROCEDURE exec_func3(RET_NUM OUT INT)
AS
BEGIN
    RET_NUM := 1+1;
COMMIT;
END;
/
gaussdb=# CREATE OR REPLACE PROCEDURE exec_func4(ADD_NUM IN INT)
AS
SUM_NUM INT;
BEGIN
SUM_NUM := ADD_NUM + exec_func3();
COMMIT;
END;
/
gaussdb=# CALL exec_func4(1);
ERROR:  Can not commit/rollback if it's atomic is true: transaction statement in store procedure used as a expression is not supported
CONTEXT:  PL/pgSQL function exec_func3() line 4 at COMMIT
PL/pgSQL function exec_func4(integer) line 4 at assignment
```

- Example 18: Savepoints defined outside a stored procedure cannot be released inside the stored procedure.

```
gaussdb=# CREATE OR REPLACE PROCEDURE STP_SAVEPOINT_EXAMPLE3()
AS
BEGIN
    INSERT INTO EXAMPLE1 VALUES(2);
    RELEASE SAVEPOINT s1; -- Release the savepoint defined outside the stored procedure.
    INSERT INTO EXAMPLE1 VALUES(3);
END;
/

gaussdb=# BEGIN;
INSERT INTO EXAMPLE1 VALUES(1);
SAVEPOINT s1;
CALL STP_SAVEPOINT_EXAMPLE3();
COMMIT;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ERROR:  cannot release outer savepoint
CONTEXT:  PL/pgSQL function stp_savepoint_example3() line 4 at RELEASE SAVEPOINT
```

2.10 Other Statements

2.10.1 Lock Operations

GaussDB provides multiple lock modes to control concurrent access to table data.

These modes are used when Multi-Version Concurrency Control (MVCC) cannot deliver expected behaviors.

Most GaussDB commands automatically apply appropriate locks to ensure that referenced tables are not deleted or modified in an incompatible manner during command execution. For example, when concurrent operations exist, ALTER TABLE cannot be executed on the same table.

2.10.2 Cursor Operations

GaussDB provides cursors as a data buffer for users to store execution results of SQL statements. Each cursor area has a name. Users can use SQL statements to fetch records one by one from cursors and grant the records to host variables, then being processed further by the host language.

Cursor operations include cursor definition, opening, fetching, and closing.

For the complete example of cursor operations, see **Explicit Cursors**.

2.11 Cursors

2.11.1 Overview

To process SQL statements, the stored procedure process assigns a memory segment to store context information. Cursors are handles or pointers pointing to context areas. With cursors, stored procedures can control alterations in context areas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- When a stored procedure uses returns to return a cursor, if JDBC is used to call the stored procedure, the returned cursor is unavailable. Using the **out** parameters is not affected.
- In distributed mode, if the parameter of the stored procedure called in the SQL statement is of the refcursor type or the return value is of the refcursor type, pushdown is not supported.
- When a stored procedure contains COMMIT/ROLLBACK, an explicit cursor caches all data of the cursor to ensure that the cursor is still available after COMMIT/ROLLBACK. If the cursor data volume is large, this process may take a long time.
- The FETCH cursor data in a stored procedure is stored in the specified output variables. If the number of output variables is greater than the number of data columns in the actual cursor query result set, null values are assigned to the extra variables, and no error is reported. If the number of output variables is less than the number of data columns in the actual cursor query result set, the extra columns are silently discarded, and no error is reported.
- When the GUC parameter **forbid_hold_cursor_with_forupdate** is enabled, if a cursor has the FOR UPDATE or FOR SHARE feature and is used in a cursor loop or FOR loop query statement, an error will be reported when the cursor is accessed in the next loop after a transaction is committed or rolled back.
- When the **plstmt_implicit_savepoint** option of the GUC parameter **behavior_compat_options** is enabled, if an error is reported when a cursor executes an SQL statement pushed down to a DN, the transaction will be abnormal and cannot be captured by EXCEPTION.

Cursors are classified into explicit cursors and implicit cursors. **Table 2-2** shows the usage conditions of explicit and implicit cursors for different SQL statements.

Table 2-2 Cursor usage conditions

SQL Statement	Cursor
Non-query statements	Implicit
Query statements with single-line results	Implicit or explicit
Query statements with multi-line results	Explicit

2.11.2 Explicit Cursors

An explicit cursor is used to process query statements, particularly when the query result contains multiple records.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

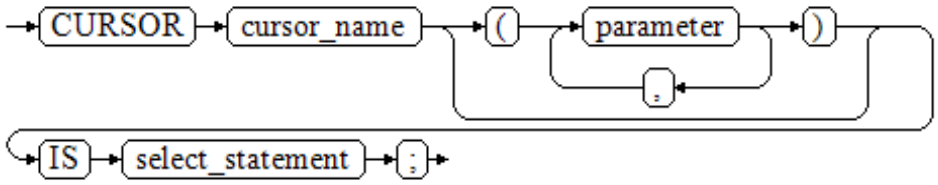
Procedure

An explicit cursor performs the following six PL/SQL steps to process query statements:

Step 1 Define a static cursor: Define a cursor name and its SELECT statement.

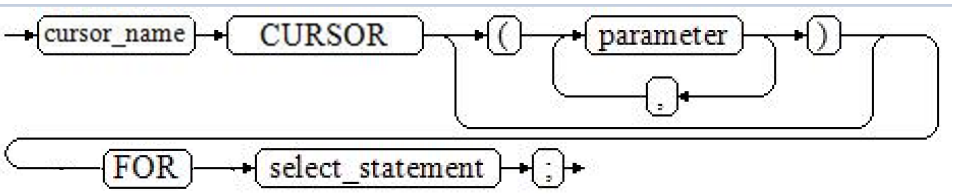
Figure 2-41 and **Figure 2-42** show the syntax diagrams for defining a static cursor.

Figure 2-41 static_cursor_define::=



or

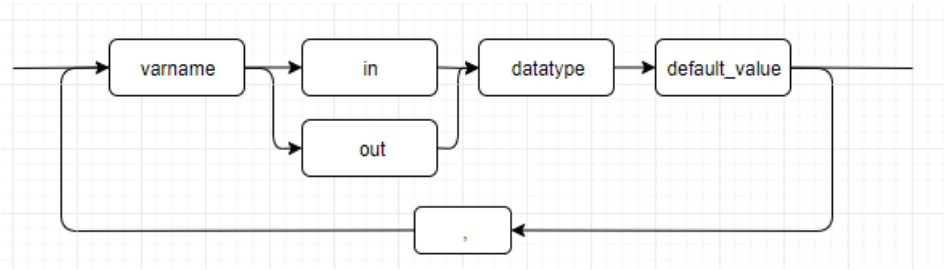
Figure 2-42 static_cursor_define::=



Parameters:

- **cursor_name**: defines a cursor name.
- **parameter**: cursor parameter, which can only be an input parameter. The default value can be defined by **:=**, **=**, or **default**. For details about the format, see **Figure 2-43**.

  Figure 2-43 Reference diagram for using the parameter format

- *select_statement*: specifies a query statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

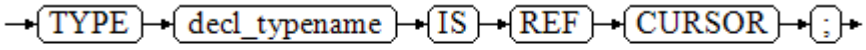
NOTE

- The system automatically determines whether the cursor can be used for backward fetching based on the execution plan.
- In syntax, **parameter** can be an output parameter, but its behavior is the same as that of an input parameter.
- It is automatically disabled in the exception.
- For a cursor with parameters created in a package, the cursor parameters can be used as variables.

Define a dynamic cursor: Define a ref cursor, which means that the cursor can be opened dynamically by a set of static SQL statements. Define the type of the ref cursor first, and then the cursor variable of this cursor type. Dynamically bind a SELECT statement through OPEN FOR when the cursor is opened.

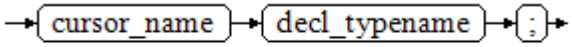
Figure 2-44 and **Figure 2-45** show the syntax diagrams for defining a dynamic cursor.

Figure 2-44 cursor_typename::=



GaussDB supports the dynamic cursor type **sys_refcursor**. A function or stored procedure can use the **sys_refcursor** parameter to pass on or pass out the cursor result set. A function can return **sys_refcursor** to return the cursor result set.

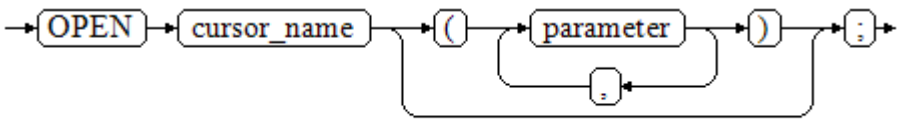
Figure 2-45 dynamic_cursor_define::=



Step 2 Open the static cursor: Execute the SELECT statement corresponding to the cursor. After that, the query result will be placed in the workspace and the pointer directs to the head of the workspace to identify the cursor result set. If the cursor query statement carries the **FOR UPDATE** option, the OPEN statement locks the data rows corresponding to the cursor result set in the database table.

Figure 2-46 shows the syntax diagram for opening a static cursor.

Figure 2-46 open_static_cursor::=

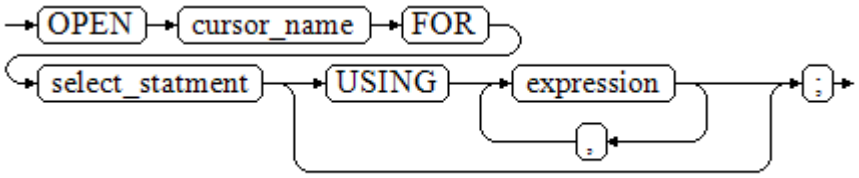


Open the dynamic cursor: Use the OPEN FOR statement to open the dynamic cursor, so that the SQL statement is dynamically bound.

Figure 2-47 shows the syntax diagram for opening a dynamic cursor.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 2-47 open_dynamic_cursor::=

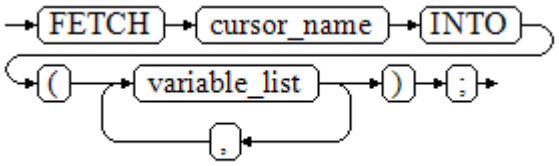


A PL/SQL program cannot use the OPEN statement to repeatedly open a cursor.

Step 3 Fetch cursor data: Retrieve data rows in the result set and place them in specified output variables.

Figure 2-48 shows the syntax diagram for fetching cursor data.

Figure 2-48 fetch_cursor::=



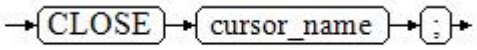
Step 4 Process the record.

Step 5 Continue to process until the active set has no record.

Step 6 Close the cursor: After you fetch and process the data in the cursor result set, close the cursor in time to release system resources used by the cursor and invalidate the workspace of the cursor so that the FETCH statement cannot be used to fetch data anymore. A closed cursor can be reopened by an OPEN statement.

Figure 2-49 shows the syntax diagram for closing a cursor.

Figure 2-49 close_cursor::=



----End

Attributes

Cursor attributes are used to control program procedures or learn about program status. When a DML statement is executed, the PL/SQL opens a built-in cursor and processes its result. A cursor is a memory segment for maintaining query results. It is opened when a DML statement is executed and closed when the execution is finished. An explicit cursor has the following attributes:

- **%FOUND**: Boolean attribute, which returns **TRUE** if the last fetch returns a row.
- **%NOTFOUND**: Boolean attribute, which returns **TRUE** if the last fetch fails to return a row.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **%ISOPEN**: Boolean attribute, which returns **TRUE** if the cursor has been opened.
- **%ROWCOUNT**: numeric attribute, which returns the number of records fetched from the cursor.

Examples

DDL and DML statements are prepared. Subsequent examples in this section depend on this case.

```
gaussdb=# drop schema if exists hr cascade;
gaussdb=# create schema hr;
gaussdb=# set current_schema = hr;
gaussdb=# drop table if exists sections;
gaussdb=# drop table if exists staffs;
gaussdb=# drop table if exists department;
-- Create a department table.
gaussdb=# create table sections(
    section_name varchar(100),
    place_id     int,
    section_id   int
);
gaussdb=# insert into sections values ('hr',1,1);

-- Create an employee table.
gaussdb=# create table staffs(
    staff_id number(6),
    salary number(8,2),
    section_id int,
    first_name varchar(20)
);
gaussdb=# insert into staffs values (1,100,1,'Tom');

-- Create a department table.
gaussdb=# create table department(
    section_id int
);
-- Specify the method for passing cursor parameters.
gaussdb=# CREATE OR REPLACE PROCEDURE cursor_proc1()
AS
DECLARE
    DEPT_NAME VARCHAR(100);
    DEPT_LOC NUMBER(4);
    -- Define a cursor.
    CURSOR C1 IS
        SELECT section_name, place_id FROM hr.sections WHERE section_id <= 50;
    CURSOR C2(sect_id INTEGER) IS
        SELECT section_name, place_id FROM hr.sections WHERE section_id <= sect_id;
    TYPE CURSOR_TYPE IS REF CURSOR;
    C3 CURSOR_TYPE;
    SQL_STR VARCHAR(100);
BEGIN
    OPEN C1;-- Open the cursor.
    LOOP
        -- Fetch data from the cursor.
        FETCH C1 INTO DEPT_NAME, DEPT_LOC;
        EXIT WHEN C1%NOTFOUND;
        DBE_OUTPUT.PRINT_LINE(DEPT_NAME||'---'||DEPT_LOC);
    END LOOP;
    CLOSE C1;-- Close the cursor.

    OPEN C2(10);
    LOOP
        FETCH C2 INTO DEPT_NAME, DEPT_LOC;
        EXIT WHEN C2%NOTFOUND;
        DBE_OUTPUT.PRINT_LINE(DEPT_NAME||'---'||DEPT_LOC);
    END LOOP;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    CLOSE C2;

    SQL_STR := 'SELECT section_name, place_id FROM hr.sections WHERE section_id <= :DEPT_NO;';
    OPEN C3 FOR SQL_STR USING 50;
    LOOP
        FETCH C3 INTO DEPT_NAME, DEPT_LOC;
        EXIT WHEN C3%NOTFOUND;
        DBE_OUTPUT.PRINT_LINE(DEPT_NAME||'---'||DEPT_LOC);
    END LOOP;
    CLOSE C3;
END;
/
gaussdb=# CALL cursor_proc1();

hr---1
hr---1
hr---1
 cursor_proc1
--------------

(1 row)


gaussdb=# DROP PROCEDURE cursor_proc1;
-- Give a salary raise to employees whose salary is lower than 3000 by adding 500.
gaussdb=# CREATE TABLE hr.staffs_t1 AS TABLE hr.staffs;

gaussdb=# CREATE OR REPLACE PROCEDURE cursor_proc2()
AS
DECLARE
    V_EMPNO  NUMBER(6);
    V_SAL    NUMBER(8,2);
    CURSOR C IS SELECT staff_id, salary FROM hr.staffs_t1;
BEGIN
    OPEN C;
    LOOP
      FETCH C INTO V_EMPNO, V_SAL;
      EXIT WHEN C%NOTFOUND;
      IF V_SAL<=3000 THEN
           UPDATE hr.staffs_t1 SET salary =salary + 500 WHERE staff_id = V_EMPNO;
      END IF;
    END LOOP;
    CLOSE C;
END;
/

gaussdb=# CALL cursor_proc2();
 cursor_proc2
--------------

(1 row)

-- Drop a stored procedure.
gaussdb=# DROP PROCEDURE cursor_proc2;
gaussdb=# DROP TABLE hr.staffs_t1;
-- Use function parameters of the SYS_REFCURSOR type.
gaussdb=# CREATE OR REPLACE PROCEDURE proc_sys_ref(O OUT SYS_REFCURSOR)
IS
C1 SYS_REFCURSOR;
BEGIN
OPEN C1 FOR SELECT section_id FROM HR.sections ORDER BY section_id;
O := C1;
END;
/

gaussdb=# DECLARE
C1 SYS_REFCURSOR;
TEMP NUMBER(4);
BEGIN
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
proc_sys_ref(C1);
LOOP
  FETCH C1 INTO TEMP;
  DBE_OUTPUT.PRINT_LINE(C1%ROWCOUNT);
  EXIT WHEN C1%NOTFOUND;
END LOOP;
END;
/
1
1
ANONYMOUS BLOCK EXECUTE
-- Drop a stored procedure.
gaussdb=# DROP PROCEDURE proc_sys_ref;
```

2.11.3 Implicit Cursors

Implicit cursors are automatically set by the system for non-query statements such as modification and deletion operations, along with their workspace. Implicit cursors are named **SQL**, which is defined by the system.

Overview

Implicit cursor operations, such as definition, opening, data fetching, and closing, are automatically performed by the system and do not need to be processed by users. Users can use only attributes related to implicit cursors to complete operations. In workspace of implicit cursors, the data of the latest SQL statement is stored and is not related to explicit cursors defined by users.

Format call: **SQL%**

NOTE

- INSERT, UPDATE, DELETE, and SELECT statements do not need defined cursors.
- Implicit cursor attributes are not affected by the commit or rollback operation.
- If an exception is triggered by a DML statement in a stored procedure, the implicit cursor attribute value is not affected by the execution of the DML statement and retains the value before the DML statement is executed.

Attributes

An implicit cursor has the following attributes:

- **SQL%FOUND**: Boolean attribute, which returns **TRUE** if the last fetch returns a row.
- **SQL%NOTFOUND**: Boolean attribute, which returns **TRUE** if the last fetch fails to return a row.
- **SQL%ROWCOUNT**: numeric attribute, which returns the number of records fetched from the cursor.
- **SQL%ISOPEN**: Boolean attribute, whose value is always **FALSE**. Close implicit cursors immediately after an SQL statement is run.

Examples

```
-- Delete all employees in a department from the hr.staffs table. If the department has no employees, delete the department from the hr.department table.

gaussdb=# CREATE OR REPLACE PROCEDURE proc_cursor3()
AS
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    DECLARE
    V_DEPTNO NUMBER(4) := 100;
    BEGIN
        DELETE FROM hr.staffs WHERE section_id = V_DEPTNO;
        -- Proceed based on cursor status.
        IF SQL%NOTFOUND THEN
        DELETE FROM hr.department WHERE section_id = V_DEPTNO;
        END IF;
    END;
/

gaussdb=# CALL proc_cursor3();
 proc_cursor3
--------------

(1 row)

-- Drop the stored procedure and the temporary table.
gaussdb=# DROP PROCEDURE proc_cursor3;
```

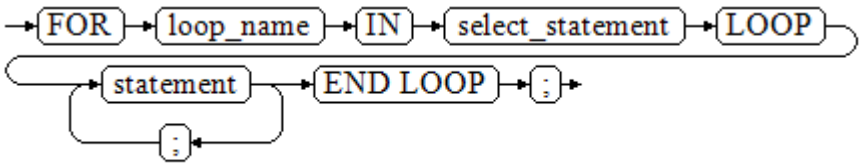
2.11.4 Cursor Loop

Use of cursors in WHILE and LOOP statements is called a cursor loop. Generally, OPEN, FETCH, and CLOSE statements are called in this kind of loop. The following describes a loop that simplifies a cursor loop without the need for these operations. This mode is applicable to a static cursor loop, without executing four steps about a static cursor.

Syntax

Figure 2-50 shows the syntax diagram of a FOR AS loop.

Figure 2-50 FOR_AS_loop::=



Precautions

- The UPDATE operation for the queried table is not allowed in the loop statement.
- The variable *loop_name* is automatically defined and is valid only in this loop. Its type is the same as that in the query result of *select_statement*. The value of *loop_name* is the query result of *select_statement*.
- The specific type of the **loop_name** variable is not parsed during compilation. If the specific type needs to be parsed (for example, **loop_name** is used as the INOUT parameters of an overloaded function or stored procedure), a compilation error is reported.

Examples

```
gaussdb=# DROP SCHEMA IF EXISTS hr CASCADE;
gaussdb=# CREATE SCHEMA hr;
gaussdb=# SET current_schema = hr;
-- Create an employee table.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CREATE TABLE staffs(
    staff_id number(6),
    salary number(8,2),
    section_id int,
    first_name varchar(20)
);
gaussdb=# INSERT INTO staffs VALUES (1,100,1,'Tom');
gaussdb=# BEGIN
FOR ROW_TRANS IN
        SELECT first_name FROM hr.staffs
    LOOP
        DBE_OUTPUT.PRINT_LINE (ROW_TRANS.first_name );
    END LOOP;
END;
/
-- Execution result.
gaussdb=# Tom
ANONYMOUS BLOCK EXECUTE
```

2.12 Advanced Packages

Advanced packages are classified into two sets of APIs: basic APIs and secondary encapsulation APIs which are used to improve usability. The second set is recommended.

2.12.1 Basic APIs

2.12.1.1 PKG_SERVICE

Table 2-3 lists APIs supported by PKG_SERVICE.

Table 2-3 PKG_SERVICE

API	Description
PKG_SERVICE.SQL_IS_CONTEXT_ACTIVE	Checks whether a context is registered.
PKG_SERVICE.SQL_CLEAN_ALL_CONTEXTS	Deregisters all contexts.
PKG_SERVICE.SQL_REGISTER_CONTEXT	Registers a context.
PKG_SERVICE.SQL_UNREGISTER_CONTEXT	Deregisters a context.
PKG_SERVICE.SQL_SET_SQL	Sets an SQL statement for a context. Currently, only the SELECT statement is supported.
PKG_SERVICE.SQL_RUN	Executes the SQL statement configured for a context.
PKG_SERVICE.SQL_NEXT_ROW	Reads the next row of data in a context.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
PKG_SERVICE.SQL_GET_VALUE	Reads a dynamically defined column value in a context.
PKG_SERVICE.SQL_SET_RESULT_TYPE	Dynamically defines a column of a context based on the type OID.
PKG_SERVICE.JOB_CANCEL	Removes a scheduled task by task ID.
PKG_SERVICE.JOB_FINISH	Disables or enables scheduled job execution.
PKG_SERVICE.JOB_SUBMIT	Submits a scheduled job. Job ID can be automatically generated by the system or specified manually.
PKG_SERVICE.JOB_UPDATE	Modifies attributes of a scheduled job, including the job content, next-execution time, and execution interval.
PKG_SERVICE.SUBMIT_ON_NODES	Submits a job to all nodes. The job ID is automatically generated by the system.
PKG_SERVICE.ISUBMIT_ON_NODES	Submits a job to all nodes. The job ID is specified by the user.
PKG_SERVICE.SQL_GET_ARRAY_RESULT	Obtains the array value returned from the context.
PKG_SERVICE.SQL_GET_VARIABLE_RESULT	Obtains the column value returned from the context.

NOTE

When the EXECUTE DIRECT ON syntax is used to execute functions in the PKG_SERVICE package, if the GUC parameter **enable_udf_execute_direct** is set to **on**, the functions can be executed; if the parameter is set to **off**, the functions cannot be executed.

- PKG_SERVICE.SQL_IS_CONTEXT_ACTIVE

  Checks whether a context is registered. This function passes the ID of the context to be queried. If the context exists, **true** is returned. Otherwise, **false** is returned.

  The prototype of PKG_SERVICE.SQL_IS_CONTEXT_ACTIVE is as follows:

```
PKG_SERVICE.SQL_IS_CONTEXT_ACTIVE(
 context_id     IN INTEGER
)
RETURN BOOLEAN;
```

Table 2-4 PKG_SERVICE.SQL_IS_CONTEXT_ACTIVE API parameters

Parameter	Description
context_id	ID of the context to be queried.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- PKG_SERVICE.SQL_CLEAN_ALL_CONTEXTS

  Deregisters all contexts.

  The prototype of PKG_SERVICE.SQL_CLEAN_ALL_CONTEXTS is as follows:

  ```
  PKG_SERVICE.SQL_CLEAN_ALL_CONTEXTS(
  )
  RETURN VOID;
  ```

- PKG_SERVICE.SQL_REGISTER_CONTEXT

  Registers a context. Registering a context is the prerequisite for performing operations on the context. This function does not require any input parameter. It automatically generates a context ID in ascending order and assigns the context ID as a return value to an integer variable.

  The prototype of PKG_SERVICE.SQL_REGISTER_CONTEXT is as follows:

  ```
  DBE_SQL.REGISTER_CONTEXT(
  )
  RETURN INTEGER;
  ```

- PKG_SERVICE.SQL_UNREGISTER_CONTEXT

  Deregisters a context. Deregistering a context is the end of all operations on the context. If this function is not called when a stored procedure ends, the memory occupied by the context will not be released. Therefore, remember to deregister the context. If an exception occurs, the stored procedure may exit prematurely, resulting in the context not being deregistered. You are advised to include this API in the exception handling of the stored procedure.

  The prototype of PKG_SERVICE.SQL_UNREGISTER_CONTEXT is as follows:

  ```
  PKG_SERVICE.SQL_UNREGISTER_CONTEXT(
   context_id     IN INTEGER
  )
  RETURN INTEGER;
  ```

  Table 2-5 PKG_SERVICE.SQL_UNREGISTER_CONTEXT API parameters

Parameter	Description
context_id	ID of the context to be deregistered.

- PKG_SERVICE.SQL_SET_SQL

  This function parses the query statement of a specified cursor. The passed query statement is executed immediately. Currently, only the SELECT query statement can be parsed. The statement parameters can be specified only through the TEXT type. The length cannot exceed 1 GB.

  The prototype of PKG_SERVICE.SQL_SET_SQL is as follows:

  ```
  PKG_SERVICE.SQL_SET_SQL(
  context_id     IN INTEGER,
  query_string   IN TEXT,
  language_flag  IN INTEGER
  )
  RETURN BOOLEAN;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-6 PKG_SERVICE.SQL_SET_SQL API parameters

Parameter	Description
context_id	ID of the context whose query statement is to be parsed.
query_string	Query statement to be parsed.
language_flag	Version language number, which specifies the behavior of different versions. ● **1**: incompatible version. ● **2**: ORA-compatible version.

- PKG_SERVICE.SQL_RUN

  Executes the SQL statement configured for a context. It receives a context ID first, and the data obtained after execution is used for subsequent operations. Currently, only the SELECT query statement can be executed.

  The prototype of PKG_SERVICE.SQL_RUN is as follows:

```
PKG_SERVICE.SQL_RUN(
context_id     IN INTEGER,
)
RETURN INTEGER;
```

Table 2-7 PKG_SERVICE.SQL_RUN API parameters

Parameter	Description
context_id	ID of the context whose query statement is to be parsed.

- PKG_SERVICE.SQL_NEXT_ROW

  Returns the number of data rows returned from the SQL statement executed. Each time the API is called, the system reads the next row of data until all data is read.

  The prototype of PKG_SERVICE.SQL_NEXT_ROW is as follows:

```
PKG_SERVICE.SQL_NEXT_ROW(
context_id     IN INTEGER,
)
RETURN INTEGER;
```

Table 2-8 PKG_SERVICE.SQL_NEXT_ROW API parameters

Parameter	Description
context_id	ID of the context to be executed.

- PKG_SERVICE.SQL_GET_VALUE

  Returns the value of the element at the specified position in the specified context. This API accesses the data obtained by PKG_SERVICE.SQL_NEXT_ROW.

  The prototype of PKG_SERVICE.SQL_GET_VALUE is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
PKG_SERVICE.SQL_GET_VALUE(
context_id          IN    INTEGER,
pos                 IN    INTEGER,
col_type            IN    ANYELEMENT
)
RETURN ANYELEMENT;
```

Table 2-9 PKG_SERVICE.SQL_GET_VALUE API parameters

Parameter	Description
context_id	ID of the context to be executed.
pos	Position of a dynamically defined column in the query.
col_type	Variable of any type, which defines the return value type of columns.

- PKG_SERVICE.SQL_SET_RESULT_TYPE

  Defines columns returned from the specified context. This API applies only to the context defined by SELECT. The defined columns are identified by the relative positions in the query list. The prototype of PKG_SERVICE.SQL_SET_RESULT_TYPE is as follows:

  ```
  PKG_SERVICE.SQL_SET_RESULT_TYPE(
  context_id     IN INTEGER,
  pos            IN INTEGER,
  coltype_oid    IN ANYELEMENT,
  maxsize        IN INTEGER
  )
  RETURN INTEGER;
  ```

  Table 2-10 PKG_SERVICE.SQL_SET_RESULT_TYPE API parameters

Parameter	Description
context_id	ID of the context to be executed.
pos	Position of a dynamically defined column in the query.
coltype_oid	Variable of any type. The OID of the corresponding type can be obtained based on the variable type.
maxsize	Length of a defined column.

- PKG_SERVICE.JOB_CANCEL

  Removes a specified scheduled job.

  The prototype of PKG_SERVICE.JOB_CANCEL is as follows:

  ```
  PKG_SERVICE.JOB_CANCEL(
  id  IN  INTEGER);
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-11 PKG_SERVICE.JOB_CANCEL API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
id	INTEGER	IN	No	ID of the scheduled job to be deleted.

- PKG_SERVICE.JOB_FINISH

  Disables or enables a scheduled job.

  The prototype of PKG_SERVICE.JOB_FINISH is as follows:

```
PKG_SERVICE.JOB_FINISH(
id          IN   INTEGER,
broken      IN   BOOLEAN,
next_time   IN   TIMESTAMP  DEFAULT  sysdate);
```

Table 2-12 PKG_SERVICE.JOB_FINISH API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
id	INTEGER	IN	No	ID of the scheduled job to be disabled or enabled.
broken	BOOLEAN	IN	No	Status flag: **true** for broken and **false** for not broken. The job status is updated based on the parameter value **true** or **false**. If it is null, the job status remains unchanged.
next_time	TIMESTAMP	IN	Yes	Next execution time. The default value is the current system time. This parameter can be omitted, and its default value will be used in this case. ● If the value of **broken** is **true**, the value of this parameter is updated to **'4000-1-1'**. ● If the parameter **broken** is set to **false** and the parameter **next_time** is not null, the **next_time** value of a specified job is updated. If the parameter **next_time** is null, the **next_time** value is not updated.

- PKG_SERVICE.JOB_SUBMIT

  Submits a system scheduled job.

  The prototype of PKG_SERVICE.JOB_SUBMIT is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
PKG_SERVICE.JOB_SUBMIT(
id            IN   BIGINT,
content       IN   TEXT,
next_time     IN   TIMESTAMP DEFAULT sysdate,
interval_time IN   TEXT  DEFAULT 'null',
job           OUT  INTEGER);
```

NOTE

- When a scheduled job is created, the system binds the current database and the username to the job by default.
- This function can be called by using the CALL or SELECT statement. If you call this function by using the SELECT statement, there is no need to specify output parameters. To call this function within a stored procedure, use the PERFORM statement.
- If the submitted SQL statement job accesses a non-**PUBLIC** schema, specify the schema where a table or a function resides, or add **SET current_schema =** *xxx***;** before the SQL statement.

Table 2-13 PKG_SERVICE.JOB_SUBMIT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
id	BIGINT	IN	No	ID of a scheduled job. If the input ID is **NULL**, a job ID is generated internally.
content	TEXT	IN	No	SQL statement to be executed. One or multiple DML statements, anonymous blocks, and statements for calling stored procedures, or all three combined are supported.
next_time	TIMESTAMP	IN	No	Next time the job will be executed. The default value is the current system time (**sysdate**). If the specified time has past, the job is executed immediately after being submitted.
interval_time	TEXT	IN	Yes	Expression for calculating the next time the job will be executed. It can be an interval expression, or **sysdate** followed by a NUMERIC value, for example, **sysdate+1.0/24**. If the value is left empty or set to **"null"**, the job will be executed only once, and the job status will change to **'d'** afterward, indicating that the job is no longer executed.
job	INTEGER	OUT	No	Job ID. The value ranges from 1 to 3000000. When pkg_service.job_submit is called using the SELECT statement, this parameter can be omitted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- PKG_SERVICE.JOB_UPDATE

  Modifies attributes of a scheduled job, including the job content, next-execution time, and execution interval.

  The prototype of PKG_SERVICE.JOB_UPDATE is as follows:

```
PKG_SERVICE.JOB_UPDATE(
id            IN   BIGINT,
next_time     IN   TIMESTAMP,
interval_time IN   TEXT,
content       IN   TEXT);
```

Table 2-14 PKG_SERVICE.JOB_UPDATE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
id	INTEGER	IN	No	ID of a specified scheduled job.
next_time	TIMESTAMP	IN	Yes	Next execution time. If this parameter is null, the system does not update the **next_time** parameter for the specified job. Otherwise, the system updates the **next_time** parameter for the specified job.
interval_time	TEXT	IN	Yes	Expression for calculating the next time the job will be executed. ● If this parameter is null, the **interval_time** value of the specified job is not updated. ● If this parameter is not null, the system checks whether **interval_time** is a valid time type or interval type. If it is valid, the **interval_time** value of the specified job is updated. ● If the value is set to **"null"**, the job will be executed only once, and the job status will change to **'d'** afterward, indicating that the job is no longer executed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
content	TEXT	IN	Yes	Name of the stored procedure or SQL statement block that is executed. If this parameter is null, the system does not update the **content** parameter for the specified job. Otherwise, the system updates the **content** parameter for the specified job.

Example:

```
CREATE TABLE test_table(a int);
CREATE TABLE

CREATE OR REPLACE PROCEDURE test_job(a in int) IS
BEGIN
INSERT INTO test_table VALUES(a);
COMMIT;
END;
/
CREATE PROCEDURE

--PKG_SERVICE.JOB_SUBMIT
SELECT PKG_SERVICE.JOB_SUBMIT(NULL, 'call test_job(1);', to_date('20180101','yyyymmdd'),'sysdate
+1');
 job_submit
------------
      28269
(1 row)

SELECT PKG_SERVICE.JOB_SUBMIT(NULL, 'call test_job(1);', to_date('20180101','yyyymmdd'),'sysdate
+1.0/24');
 job_submit
------------
       1506
(1 row)

CALL PKG_SERVICE.JOB_SUBMIT(NULL, 'INSERT INTO test_table VALUES(1);  call test_job(1); call
test_job(1);', add_months(to_date('201701','yyyymm'),1), 'date_trunc(''day'',SYSDATE) + 1
+(8*60+30.0)/(24*60)' ,:jobid);
  job
-------
 14131
(1 row)

SELECT PKG_SERVICE.JOB_SUBMIT (101, 'insert_msg_statistic1;', sysdate, 'sysdate+3.0/24');
 job_submit
------------
       101
(1 row)

--PKG_SERVICE.JOB_UPDATE
CALL PKG_SERVICE.JOB_UPDATE(101, sysdate, 'sysdate + 1.0/1440', 'call test_job(1);');
 job_update
------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)

CALL PKG_SERVICE.JOB_UPDATE(101, sysdate, 'sysdate + 1.0/1440', 'insert into test_table values(1);');
 job_update
------------

(1 row)

--PKG_SERVICE.JOB_FINISH
CALL PKG_SERVICE.JOB_FINISH(101,true);
 job_finish
------------

(1 row)

--PKG_SERVICE.JOB_CANCEL
CALL PKG_SERVICE.JOB_CANCEL(101);
 job_cancel
------------

(1 row)

DROP TABLE test_table;
DROP TABLE
```

- PKG_SERVICE.SUBMIT_ON_NODES

  Creates a scheduled job on all CNs and DNs. Only users with the SYSADMIN or MONADMIN permission can call this function.

  The prototype of PKG_SERVICE.SUBMIT_ON_NODES is as follows:

```
PKG_SERVICE.SUBMIT_ON_NODES(
node_name    IN   NAME,
database     IN   NAME,
what         IN   TEXT,
next_date    IN   TIMESTAMP WITHOUT TIME ZONE,
job_interval IN   TEXT,
job          OUT  INTEGER);
```

Table 2-15 PKG_SERVICE.SUBMIT_ON_NODES API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
node_name	TEXT	IN	No	Job execution node. Currently, the value can only be **ALL_NODE** (indicating that the job is executed on all nodes) or **CCN** (indicating that the job is executed on the CCN).
database	TEXT	IN	No	Database used by a cluster job. When the node type is **ALL_NODE**, the value can only be **postgres**.
what	TEXT	IN	No	SQL statement to be executed. One or multiple DML statements, anonymous blocks, and statements for calling stored procedures, or all three combined are supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
nextdate	TIMESTAMP	IN	No	Next time the job will be executed. The default value is the current system time (**sysdate**). If the specified time has past, the job is executed immediately after being submitted.
job_interval	TEXT	IN	No	Expression for calculating the next time the job will be executed. It can be an interval expression, or **sysdate** followed by a NUMERIC value, for example, **sysdate+1.0/24**. If the value is left empty or set to **"null"**, the job will be executed only once, and the job status will change to **'d'** afterward, indicating that the job is no longer executed.
job	INTEGER	OUT	No	Job ID. The value ranges from 1 to 3000000. When DBMS.SUBMIT_ON_NODES is called using the SELECT statement, this parameter can be omitted.

Example:

```
SELECT pkg_service.submit_on_nodes('ALL_NODE', 'postgres', 'select
capture_view_to_json(''dbe_perf.statement'', 0);', sysdate, 'interval ''60 second''');
 submit_on_nodes
-----------------
           12068
(1 row)

SELECT pkg_service.submit_on_nodes('CCN', 'postgres', 'select
capture_view_to_json(''dbe_perf.statement'', 0);', sysdate, 'interval ''60 second''');
 submit_on_nodes
-----------------
            9027
(1 row)
```

- PKG_SERVICE.ISUBMIT_ON_NODES

  ISUBMIT_ON_NODES has the same syntax function as **SUBMIT_ON_NODES**, but the first parameter of **ISUBMIT_ON_NODES** is an input parameter, that is, a specified job ID. In contrast, the last parameter of **ISUBMIT_ON_NODES** is an output parameter, indicating the job ID automatically generated by the system. Only users with the SYSADMIN or MONADMIN permission can call this function.

- PKG_SERVICE.SQL_GET_ARRAY_RESULT

  Returns the value of the bound **OUT** parameter of the array type and obtains the **OUT** parameter in a stored procedure.

  The prototype of PKG_SERVICE.SQL_GET_ARRAY_RESULT is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
PKG_SERVICE.SQL_GET_ARRAY_RESULT(
    context_id in int,
    pos in VARCHAR2,
    column_value inout anyarray,
    result_type in anyelement
);
```

Table 2-16 PKG_SERVICE.SQL_GET_ARRAY_RESULT API parameters

Parameter	Description
context_id	ID of the context to be queried.
pos	Name of the bound parameter.
column_value	Return value.
result_type	Return type.

- PKG_SERVICE.SQL_GET_VARIABLE_RESULT

  Returns the value of the bound **OUT** parameter of the non-array type and obtains the **OUT** parameter in a stored procedure.

  The prototype of PKG_SERVICE.SQL_GET_VARIABLE_RESULT is as follows:

```
PKG_SERVICE.SQL_GET_VARIABLE_RESULT(
    context_id in int,
    pos in VARCHAR2,
    result_type in anyelement
)
RETURNS anyelement;
```

Table 2-17 PKG_SERVICE.SQL_GET_VARIABLE_RESULT API parameters

Parameter	Description
context_id	ID of the context to be queried.
pos	Name of the bound parameter.
result_type	Return type.

2.12.1.2 PKG_UTIL

Table 2-18 lists all APIs supported by the PKG_UTIL package.

NOTE

LOB update APIs (such as LOB_WRITE_HUGE, LOB_COPY_HUGE, LOB_CONVERTTOBLOB_HUGE, and LOADBLOBFROMFILE) in the PKG_UTIL package update only LOB variables in memory and do not automatically update the associated physical table data. Even if the target row has been locked using SELECT ... FOR UPDATE, LOB data changes are not automatically persisted to the database table. To save the update result, you must explicitly execute the UPDATE statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-18 PKG_UTIL

API	Description
PKG_UTIL.LOB_READ	Reads a part of a LOB.
PKG_UTIL.LOB_WRITE	Writes the source object to the target object in the specified format.
PKG_UTIL.LOB_APPEND	Appends the source LOB to the target LOB.
PKG_UTIL.LOB_COMPARE	Compares two LOBs based on the specified length.
PKG_UTIL.LOB_MATCH	Returns the position of the *N*th occurrence of a character string in a LOB.
PKG_UTIL.LOB_RESET	Sets a segment of data in a BLOB to the specified value and returns the processed BLOB and the actually processed length.
PKG_UTIL.LOB_GET_LENGTH	Obtains and returns the specified length of a LOB.
PKG_UTIL.LOB_READ_HUGE	Reads a part of the LOB content based on the specified length and initial position offset, and returns the read LOB and length.
PKG_UTIL.LOB_WRITEAPPEND_HUGE	Reads the content of a specified length from the source BLOB or CLOB, appends the content to the target BLOB or CLOB, and returns the target object.
PKG_UTIL.LOB_APPEND_HUGE	Appends the source BLOB/CLOB to the target BLOB/CLOB and returns the target object.
PKG_UTIL.READ_BFILE_TO_BLOB	Loads the source BFILE file to the target BLOB and returns the target object.
PKG_UTIL.LOB_COPY_HUGE	Reads the content of a specified length from the specified offset position of the source BLOB or CLOB, writes the content to the specified offset position of the target BLOB or CLOB, and returns the target object.
PKG_UTIL.BLOB_RESET	Sets a segment of data in a BLOB to the specified value and returns the processed BLOB and the actually processed length.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
PKG_UTIL.CLOB_RESET	Sets a segment of data in a CLOB to spaces and returns the processed CLOB and the actually processed length.
PKG_UTIL.LOADBLOBFROMFILE	Reads the content of a specified length from the specified offset position of the source BFILE object, writes the content to the specified offset position of the target BLOB, and returns the target object, read position, and write position.
PKG_UTIL.LOADCLOBFROMFILE	Reads the content of a specified length from the specified offset position of the source BFILE object, writes the content to the specified offset position of the target CLOB, and returns the target object, read position, and write position.
PKG_UTIL.LOB_CONVERTTOBLOB_HUGE	Reads the content of a specified length from the specified offset position of the source CLOB, converts the content to a BLOB, and writes the BLOB to the specified position of target LOB. *amount* indicates the length to be converted.
PKG_UTIL.LOB_CONVERTTOCLOB_HUGE	Reads the content of a specified length from the specified offset position of the source CLOB, converts the content to a CLOB, and writes the CLOB to the specified position of target LOB. *amount* indicates the length to be converted.
PKG_UTIL.BFILE_GET_LENGTH	Obtains and returns the specified length of a BFILE file.
PKG_UTIL.BFILE_OPEN	Opens a BFILE file and returns its file descriptor.
PKG_UTIL.BFILE_CLOSE	Closes a BFILE file.
PKG_UTIL.LOB_WRITE_HUGE	Reads the specified length of the source object from the start position, writes the content to the specified offset position of the target LOB, overrides the original content, and returns the target LOB.
PKG_UTIL.IO_PRINT	Displays character strings.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
PKG_UTIL.RAW_GET_LENGTH	Obtains the length of RAW data.
PKG_UTIL.RAW_CAST_FROM_VARCHAR2	Converts VARCHAR2 data to RAW data.
PKG_UTIL.RAW_CAST_FROM_BINARY_INTEGER	Converts binary integers to RAW data.
PKG_UTIL.RAW_CAST_TO_BINARY_INTEGER	Converts RAW data to binary integer data.
PKG_UTIL.RANDOM_SET_SEED	Sets a random seed.
PKG_UTIL.RANDOM_GET_VALUE	Returns a random value.
PKG_UTIL.FILE_SET_DIRNAME	Sets the directory to be operated.
PKG_UTIL.FILE_OPEN	Opens a file based on the specified file name and directory.
PKG_UTIL.FILE_SET_MAX_LINE_SIZE	Sets the maximum length of a line to be written to a file.
PKG_UTIL.FILE_IS_CLOSE	Checks whether a file handle is closed.
PKG_UTIL.FILE_READ	Reads data of a specified length from an open file handle.
PKG_UTIL.FILE_READLINE	Reads a line of data from an open file handle.
PKG_UTIL.FILE_WRITE	Writes the data specified in the buffer to a file.
PKG_UTIL.FILE_WRITELINE	Writes the buffer to a file and adds newline characters.
PKG_UTIL.FILE_NEWLINE	Adds a line.
PKG_UTIL.FILE_READ_RAW	Reads binary data of a specified length from an open file handle.
PKG_UTIL.FILE_WRITE_RAW	Writes binary data to a file.
PKG_UTIL.FILE_FLUSH	Writes data from a file handle to a physical file.
PKG_UTIL.FILE_CLOSE	Closes a file handle.
PKG_UTIL.FILE_REMOVE	Removes a physical file. To do so, you must have the corresponding permission.
PKG_UTIL.FILE_RENAME	Renames a file on the disk, similar to **mv** in Unix.
PKG_UTIL.FILE_SIZE	Returns the size of a file.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
PKG_UTIL.FILE_BLOCK_SIZE	Returns the number of blocks contained in a file.
PKG_UTIL.FILE_EXISTS	Checks whether a file exists.
PKG_UTIL.FILE_GETPOS	Specifies the offset of a returned file, in bytes.
PKG_UTIL.FILE_SEEK	Sets the offset for determining the file position.
PKG_UTIL.FILE_CLOSE_ALL	Closes all file handles opened in a session.
PKG_UTIL.EXCEPTION_REPORT_ERROR	Throws an exception.
PKG_UTIL.APP_READ_CLIENT_INFO	Reads the client information.
PKG_UTIL.APP_SET_CLIENT_INFO	Sets the client information.
PKG_UTIL.LOB_CONVERTTOBLOB	Converts the CLOB type to the BLOB type.
PKG_UTIL.LOB_CONVERTTOCLOB	Converts the BLOB type to the CLOB type.
PKG_UTIL.LOB_RAWTOTEXT	Converts the input BLOB into a hexadecimal text output, automatically prefixing a leading zero for odd-length inputs. The input must consist of valid hexadecimal digits (0-9, A-F, a-f). If any invalid characters are included, an error will be reported.
PKG_UTIL.LOB_TEXTTORAW	Converts the input CLOB text into a RAW output, automatically prefixing a leading zero for odd-length inputs. The input text must consist of valid hexadecimal digits (0-9, A-F, a-f). If any invalid characters are included, an error will be reported.
PKG_UTIL.MATCH_EDIT_DISTANCE_SIMILARITY	Calculates the difference between two character strings.
PKG_UTIL.RAW_CAST_TO_VARCHAR2	Converts the RAW type to the VARCHAR2 type.
PKG_UTIL.SESSION_CLEAR_CONTEXT	Clears the attribute values in **session_context**.
PKG_UTIL.SESSION_SEARCH_CONTEXT	Searches for an attribute value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
PKG_UTIL.SESSION_SET_CONTEXT	Sets an attribute value.
PKG_UTIL.UTILITY_FORMAT_CALL_STACK	Displays the call stack of a stored procedure.
PKG_UTIL.UTILITY_FORMAT_ERROR_BACKTRACE	Displays the error stack of a stored procedure.
PKG_UTIL.UTILITY_FORMAT_ERROR_STACK	Displays the error information about a stored procedure.
PKG_UTIL.UTILITY_GET_TIME	Displays the Unix timestamp of the system.
PKG_UTIL.UTILITY_COMPILE_SCHEMA	Recompiles specified schemas, packages, functions, and stored procedures. If an error is reported when you compile a PL/SQL object, the PL/SQL object is returned and the recompilation stops. This API is not recommended. You are advised to use pkg_util.gs_compile_schema.
PKG_UTIL.GS_COMPILE_SCHEMA	Recompiles specified schemas, packages, functions, and stored procedures. If an error is reported when you compile a PL/SQL object, the exception is captured and the recompilation continues until all objects are compiled or the number of recompilation attempts reaches the upper limit. When the advanced package is executed through JDBC, **SQLSTATE** will return error code "00000", which indicates that the operation is successful..
PKG_UTIL.APP_SET_MODULE	Sets the value of a module.
PKG_UTIL.APP_READ_MODULE	Reads the value of a module.
PKG_UTIL.APP_SET_ACTION	Sets the value of an action.
PKG_UTIL.APP_READ_ACTION	Reads the value of an action.
PKG_UTIL.MODIFY_PACKAGE_STATE	Modifies the PL/SQL status of the current session.

- PKG_UTIL.LOB_READ

  Reads an object and returns the specified part.

  The prototype of the PKG_UTIL.LOB_READ function is as follows:

  ```
  PKG_UTIL.LOB_READ(
  lob       IN   ANYELEMENT,
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
len       IN   INT,
start     IN   INT,
mode      IN   INT
)
RETURN ANYELEMENT
```

Table 2-19 PKG_UTIL.LOB_READ API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
lob	CLOB / BLOB	IN	No	CLOB or BLOB data.
len	INT	IN	No	Length of the returned result.
start	INT	IN	No	Offset to the first character.
mode	INT	IN	No	Type of the read operation. **0** indicates **READ**, **1** indicates **TRIM**, and **2** indicates **SUBSTR**.

- PKG_UTIL.LOB_WRITE

  Writes the source object to the target object based on the specified parameters and returns the target object.

  The prototype of the PKG_UTIL.LOB_WRITE function is as follows:

```
PKG_UTIL.LOB_WRITE(
dest_lob    INOUT   BLOB,
src_lob     IN      RAW
len         IN      INT,
start_pos   IN      BIGINT
)
RETURN BLOB;
PKG_UTIL.LOB_WRITE(
dest_lob    INOUT   CLOB,
src_lob     IN      VARCHAR2
len         IN      INT,
start_pos   IN      BIGINT
)
RETURN CLOB;
```

Table 2-20 PKG_UTIL.LOB_WRITE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
dest_lob	CLOB / BLOB	INOUT	No	Target object to which data will be written.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
src_lob	CLOB / BLOB	IN	No	Source object to be written.
len	INT	IN	No	Write length of the source object.
start_pos	BIGINT	IN	No	Write start position of the target object.

- PKG_UTIL.LOB_APPEND

  Appends the source BLOB/CLOB to the target BLOB/CLOB and returns the target object.

  The prototype of the PKG_UTIL.LOB_APPEND function is as follows:

```
PKG_UTIL.LOB_APPEND(
dest_lob    INOUT   BLOB,
src_lob     IN      BLOB,
len         IN      INT DEFAULT NULL
)
RETURN BLOB;

PKG_UTIL.LOB_APPEND(
dest_lob    INOUT   CLOB,
src_lob     IN      CLOB,
len         IN      INT DEFAULT NULL
)
RETURN CLOB;
```

Table 2-21 PKG_UTIL.LOB_APPEND API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
dest_lob	BLOB/CLOB	INOUT	No	Target BLOB/CLOB to which data will be written.
src_lob	BLOB/CLOB	IN	No	BLOB/CLOB from which data is to be written.
len	INT	IN	Yes	Length read from *src* and appended to *dest*. The default value is **NULL**, indicating that all content of *src* is read and appended to *dest*.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- PKG_UTIL.LOB_COMPARE

  Checks whether objects are the same based on the specified start position and size. If **lob1** is larger, **1** is returned. If **lob2** is larger, **-1** is returned. If **lob1** is equal to **lob2**, **0** is returned.

  The prototype of the PKG_UTIL.LOB_COMPARE function is as follows:

```
PKG_UTIL.LOB_COMPARE(
lob1        IN   ANYELEMENT,
lob2        IN   ANYELEMENT,
len         IN   INT DEFAULT 1073741771,
start_pos1      IN   INT DEFAULT 1,
start_pos2      IN   INT DEFAULT 1
)
RETURN INTEGER;
```

  Table 2-22 PKG_UTIL.LOB_COMPARE API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
lob1	CLOB/ BLOB	IN	No	Character string for comparison.
lob2	CLOB/ BLOB	IN	No	Character string for comparison.
len	INT	IN	No	Length to be compared.
start_pos1	INT	IN	No	Start offset of **lob1**.
start_pos2	INT	IN	No	Start offset of **lob2**.

- PKG_UTIL.LOB_MATCH

  Returns the position where a pattern is displayed in a LOB for the *match_n*th time.

  The prototype of the PKG_UTIL.LOB_MATCH function is as follows:

```
PKG_UTIL.LOB_MATCH(
lob         IN   ANYELEMENT,
pattern      IN   ANYELEMENT,
start       IN   INT,
match_nth    IN   INT DEFAULT 1
)
RETURN INTEGER;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-23 PKG_UTIL.LOB_MATCH API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
lob	CLOB/BLOB	IN	No	Character string for comparison.
pattern	CLOB/BLOB	IN	No	Pattern to be matched.
start	INT	IN	No	Start position for LOB comparison.
match_nth	INT	IN	No	Number of matching times when the matching succeeds.

- PKG_UTIL.LOB_RESET

  Sets a segment of data in a BLOB to the specified value and returns the processed BLOB and the actually processed length.

  The prototype of the PKG_UTIL.LOB_RESET function is as follows:

```
PKG_UTIL.LOB_RESET(
lob         INOUT   BLOB,
len         INOUT   INT,
start       IN   INT DEFAULT 1,
value        IN   INT DEFAULT 0
)
RETURN RECORD;
```

Table 2-24 PKG_UTIL.LOB_RESET API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
lob	BLOB	INOUT	No	Character string for resetting. **NOTE** This function does not reset the content of the LOB parameter. The processed result is returned by the function.
len	INT	INOUT	No	Length of the string to be reset.
start	INT	IN	Yes	Start position for resetting. Default value: **'1'**.
value	INT	IN	Yes	Character to be set. Default value: **'0'**.

- PKG_UTIL.LOB_GET_LENGTH

  Obtains and returns the specified length of a LOB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The prototype of the PKG_UTIL.LOB_GET_LENGTH function is as follows:

```
PKG_UTIL.LOB_GET_LENGTH(
   lob IN BLOB)
RETURN BIGINT;

PKG_UTIL.LOB_GET_LENGTH(
   lob IN CLOB)
RETURN BIGINT;
```

Table 2-25 PKG_UTIL.LOB_GET_LENGTH API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
lob	BLOB/CLOB	IN	No	LOB type.

- PKG_UTIL.LOB_READ_HUGE

  Reads a part of the LOB content based on the specified length and initial position offset, and returns the read LOB and length.

  The prototype of the PKG_UTIL.LOB_READ_HUGE function is as follows:

```
PKG_UTIL.LOB_READ_HUGE(
   lob       IN CLOB,
   len       IN BIGINT,
   start_pos IN BIGINT,
   mode      IN INTEGER)
RETURN RECORD;

PKG_UTIL.LOB_READ_HUGE(
   lob       IN BLOB,
   len       IN BIGINT,
   start_pos IN BIGINT,
   mode      IN INTEGER)
RETURN RECORD;

PKG_UTIL.LOB_READ_HUGE(
   fd        IN INTEGER,
   len       IN BIGINT,
   start_pos IN BIGINT,
   mode      IN INTEGER)
RETURN RECORD;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-26 PKG_UTIL.LOB_READ_HUGE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
lob/fd	BLOB/CLOB/INTEGER	IN	No	File descriptor of the specified LOB or BFILE file.
len	BIGINT	IN	No	Length to read.
start_pos	BIGINT	IN	No	Offset position from which read starts.
mode	INTEGER	IN	No	Read mode (**0**: read, **1**: trim, **2**: substr).

- PKG_UTIL.LOB_WRITEAPPEND_HUGE

  Reads the content of a specified length from the source BLOB or CLOB, appends the content to the target BLOB or CLOB, and returns the target object.

  The prototype of the PKG_UTIL.LOB_WRITEAPPEND_HUGE function is as follows:

```
PKG_UTIL.LOB_WRITEAPPEND_HUGE(
   dest_lob INOUT CLOB,
   len      IN    INTEGER,
   src_lob  IN    VARCHAR2
)RETURN CLOB;

PKG_UTIL.LOB_WRITEAPPEND_HUGE(
   dest_lob INOUT BLOB,
   len      IN    INTEGER,
   src_lob  IN    RAW
)RETURN BLOB;
```

Table 2-27 PKG_UTIL.LOB_WRITEAPPEND_HUGE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
dest_lob	BLOB/CLOB	INOUT	No	Target BLOB/CLOB to which data will be written.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
len	INTEGER	IN	Yes	Length of the source object to be written. If the value is **NULL**, the entire source object is written by default.
src_lob	VARCHAR2/RAW	IN	No	BLOB/CLOB from which data is to be written.

- PKG_UTIL.LOB_APPEND_HUGE

  Appends the source BLOB/CLOB to the target BLOB/CLOB and returns the target object.

  The prototype of the PKG_UTIL.LOB_APPEND_HUGE function is as follows:

```
PKG_UTIL.LOB_APPEND_HUGE(
   dest_lob INOUT BLOB,
   src_lob  IN    BLOB)
RETURN BLOB;
PKG_UTIL.LOB_APPEND_HUGE(
   dest_lob INOUT CLOB,
   src_lob  IN    CLOB)
RETURN CLOB;
```

  Table 2-28 PKG_UTIL.LOB_APPEND_HUGE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
dest_lob	BLOB/CLOB	INOUT	No	Target BLOB/CLOB to which data will be written.
src_lob	BLOB/CLOB	IN	No	BLOB/CLOB from which data is to be written.

- PKG_UTIL.READ_BFILE_TO_BLOB

  Loads the source BFILE file to the target BLOB and returns the target object.

  The prototype of the PKG_UTIL.READ_BFILE_TO_BLOB function is as follows:

```
PKG_UTIL.READ_BFILE_TO_BLOB(
   fd IN INTEGER
)RETURN BLOB;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-29 PKG_UTIL.READ_BFILE_TO_BLOB API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
fd	INTEGER	IN	No	Source BFILE file to be read.

- PKG_UTIL.LOB_COPY_HUGE

  Reads the content of a specified length from the specified offset position of the source BLOB or CLOB, writes the content to the specified offset position of the target BLOB or CLOB, and returns the target object.

  The prototype of the PKG_UTIL.LOB_COPY_HUGE function is as follows:

```
PKG_UTIL.LOB_COPY_HUGE(
   lob_obj    INOUT BLOB,
   source_obj  IN    BLOB,
   amount      IN    BIGINT,
   dest_offset IN    BIGINT DEFAULT 1,
   src_offset  IN    BIGINT DEFAULT 1
)RETURN BLOB;

PKG_UTIL.LOB_COPY_HUGE(
   lob_obj    INOUT CLOB,
   source_obj  IN    CLOB,
   amount      IN    BIGINT,
   dest_offset IN    BIGINT DEFAULT 1,
   src_offset  IN    BIGINT DEFAULT 1
)RETURN CLOB;
```

Table 2-30 PKG_UTIL.LOB_COPY_HUGE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
lob_obj	BLOB/CLOB	INOUT	No	Target BLOB/CLOB.
source_obj	BLOB/CLOB	IN	No	Source BLOB/CLOB.
amount	BIGINT	IN	No	Length of the data to be copied (in bytes for BLOBs or in characters for CLOBs).
dest_offset	BIGINT	IN	No	Offset position of the target LOB to which the data is loaded.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
src_offset	BIGINT	IN	No	Offset position of the source LOB from which the data is read.

- PKG_UTIL.BLOB_RESET

  Sets a segment of data in a BLOB to the specified value and returns the processed BLOB and the actually processed length.

  The prototype of the PKG_UTIL.BLOB_RESET function is as follows:

```
PKG_UTIL.BLOB_RESET(
   lob       INOUT BLOB,
   len       INOUT BIGINT,
   start_pos IN    BIGINT DEFAULT 1,
   value     IN    INTEGER DEFAULT 0
)RETURN RECORD;
```

  Table 2-31 PKG_UTIL.BLOB_RESET API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
lob	BLOB	INOUT	No	LOB to be reset. **NOTE** This function does not reset the content of the LOB parameter. The processed result is returned by the function.
len	BIGINT	INOUT	No	Length to reset, in bytes.
start	BIGINT	IN	Yes	Start position for resetting. Default value: **'1'**.
value	INTEGER	IN	Yes	Character to be set. Default value: **'0'**.

- PKG_UTIL.CLOB_RESET

  Sets a piece of data to spaces.

  The prototype of the PKG_UTIL.CLOB_RESET function is as follows:

```
PKG_UTIL.CLOB_RESET(
   lob       INOUT CLOB,
   len       INOUT BIGINT,
   start_pos IN    BIGINT DEFAULT 1
)RETURN RECORD;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-32 PKG_UTIL.CLOB_RESET API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
lob	CLOB	INOUT	No	LOB to be reset.
len	BIGINT	INOUT	No	Length to reset, in characters.
start	BIGINT	IN	Yes	Reset start position. The default value is **1**.

- PKG_UTIL.LOADBLOBFROMFILE

  Reads the content of a specified length from the specified offset position of the source BFILE object, writes the content to the specified offset position of the target BLOB, and returns the target object, read position, and write position.

  The prototype of the PKG_UTIL.LOADBLOBFROMFILE function is as follows:

```
PKG_UTIL.LOADBLOBFROMFILE(
   dest_lob     INOUT  BLOB,
   fd           IN     INTEGER,
   amount       IN     BIGINT,
   dest_offset  INOUT  BIGINT,
   file_offset  INOUT  BIGINT
)RETURN RECORD;
```

Table 2-33 PKG_UTIL.LOADBLOBFROMFILE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
dest_lob	BLOB	INOUT	No	Target BLOB as the INOUT parameter.
fd	INTEGER	IN	No	File descriptor of the source BFILE object.
amount	BIGINT	IN	No	Length of the data to be copied (in bytes for BLOBs or in characters for CLOBs).
dest_offset	BIGINT	INOUT	No	Offset position of the target LOB to which the data is written as the INOUT parameter.
src_offset	BIGINT	INOUT	No	Offset position of the source BFILE file from which the data is read as the INOUT parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- PKG_UTIL.LOADCLOBFROMFILE

  Reads the content of a specified length from the specified offset position of the source BFILE object, writes the content to the specified offset position of the target CLOB, and returns the target object, read position, and write position.

  The prototype of the PKG_UTIL.LOADCLOBFROMFILE function is as follows:

```
PKG_UTIL.LOADCLOBFROMFILE(
   dest_lob     INOUT  CLOB,
   fd           IN     INTEGER,
   amount       IN     BIGINT,
   dest_offset  INOUT  BIGINT,
   file_offset  INOUT  BIGINT
)RETURN RECORD;
```

  Table 2-34 PKG_UTIL.LOADCLOBFROMFILE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
dest_lob	CLOB	INOUT	No	Target CLOB as the INOUT parameter.
fd	INTEGER	IN	No	File descriptor of the source BFILE object.
amount	BIGINT	IN	No	Length to copy (in characters for CLOBs).
dest_offset	BIGINT	INOUT	No	Offset position of the target LOB to which the data is written as the INOUT parameter.
src_offset	BIGINT	INOUT	No	Offset position of the source BFILE file from which the data is read as the INOUT parameter.

- PKG_UTIL.LOB_CONVERTTOBLOB_HUGE

  Reads the content of a specified length from the specified offset position of the source CLOB, converts the content to a BLOB, and writes the BLOB to the specified position of target LOB. *amount* indicates the length to be converted.

  The prototype of the PKG_UTIL.LOB_CONVERTTOBLOB_HUGE function is as follows:

```
PKG_UTIL.LOB_CONVERTTOBLOB_HUGE(
   dest_lob    INOUT  BLOB,
   src_clob    IN     CLOB,
   amount      IN     BIGINT,
   dest_offset INOUT  BIGINT,
   src_offset  INOUT  BIGINT)
)RETURN RECORD;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-35 PKG_UTIL.LOB_CONVERTTOBLOB_HUGE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
dest_lob	BLOB	INOUT	No	Target LOB.
src_clob	CLOB	IN	No	CLOB to be converted.
amount	BIGINT	IN	No	Length to convert, in characters.
dest_offset	BIGINT	INOUT	No	Offset position of the target LOB to which the data is written as the INOUT parameter.
src_offset	BIGINT	INOUT	No	Offset position of the source CLOB from which the data is read as the INOUT parameter.

- PKG_UTIL.LOB_CONVERTTOCLOB_HUGE

  Reads the content of a specified length from the specified offset position of the source CLOB, converts the content to a CLOB, and writes the CLOB to the specified position of target LOB. *amount* indicates the length to be converted.

  The prototype of the PKG_UTIL.LOB_CONVERTTOCLOB_HUGE function is as follows:

```
PKG_UTIL.LOB_CONVERTTOCLOB_HUGE(
   dest_lob    INOUT  CLOB,
   src_blob    IN     BLOB,
   amount      IN     BIGINT,
   dest_offset INOUT  BIGINT,
   src_offset  INOUT  BIGINT)
)RETURN RECORD;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-36 PKG_UTIL.LOB_CONVERTTOCLOB_HUGE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
dest_lob	CLOB	INOUT	No	Target LOB.
src_blob	BLOB	IN	No	BLOB to be converted.
amount	BIGINT	IN	No	Length to convert, in bytes.
dest_offset	BIGINT	INOUT	No	Offset position of the target LOB to which the data is written as the INOUT parameter.
src_offset	BIGINT	INOUT	No	Offset position of the source CLOB from which the data is read as the INOUT parameter.

- PKG_UTIL.BFILE_GET_LENGTH

  Obtains and returns the specified length of a BFILE file.

  The prototype of the PKG_UTIL.BFILE_GET_LENGTH function is as follows:

```
PKG_UTIL.BFILE_GET_LENGTH(
    fd INTEGER
)RETURN BIGINT;
```

Table 2-37 PKG_UTIL.LOB_GET_LENGTH API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
fd	INTEGER	IN	No	File descriptor of the specified BFILE file.

- PKG_UTIL.BFILE_OPEN

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Opens a BFILE file and returns its file descriptor.

The prototype of the PKG_UTIL.BFILE_OPEN function is as follows:

```
PKG_UTIL.BFILE_OPEN(
   file_name TEXT,
   open_mode TEXT)
RETURN INTEGER;
```

Table 2-38 PKG_UTIL.BFILE_OPEN API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file_name	TEXT	IN	No	Name of the specified BFILE file.
open_mode	TEXT	IN	No	Open mode, which can only be set to **r**, that is, the read mode.

- PKG_UTIL.BFILE_CLOSE

  Closes a BFILE file.

  The prototype of the PKG_UTIL.BFILE_CLOSE function is as follows:

  ```
  PKG_UTIL.BFILE_CLOSE(
     fd INTEGER)
  RETURN BOOL;
  ```

  Table 2-39 PKG_UTIL.BFILE_CLOSE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
fd	INTEGER	IN	No	File descriptor of the specified BFILE file.

- PKG_UTIL.LOB_WRITE_HUGE

  Reads the specified length of the source object from the start position, writes the content to the specified offset position of the target LOB, overrides the original content, and returns the target LOB.

  The prototype of the PKG_UTIL.LOB_WRITE_HUGE function is as follows:

  ```
  PKG_UTIL.LOB_WRITE_HUGE(
     dest_lob  INOUT BLOB,
     len       IN INTEGER,
     start_pos IN BIGINT,
     src_lob   IN RAW
  )RETURN BLOB;

  PKG_UTIL.LOB_WRITE_HUGE(
     dest_lob  INOUT CLOB,
     len       IN INTEGER,
     start_pos IN BIGINT,
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  src_lob   IN VARCHAR2
)RETURN CLOB;
```

Table 2-40 PKG_UTIL.LOB_WRITE_HUGE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
dest_lob	BLOB/CLOB	INOUT	No	Target LOB as the INOUT parameter, to which the content is to be written.
len	INTEGER	IN	No	Length of the data to be written (in bytes for BLOBs or in characters for CLOBs).
start_pos	BIGINT	IN	No	Offset position for writing data to *dest_lob*.
src_lob	RAW/VARCHAR2	IN	No	Source LOB.

- PKG_UTIL.IO_PRINT

  Prints a string.

  The prototype of the PKG_UTIL.IO_PRINT function is as follows:

```
PKG_UTIL.IO_PRINT(
format      IN   TEXT,
is_one_line  IN   BOOLEAN
)
RETURN VOID;
```

Table 2-41 PKG_UTIL.IO_PRINT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
format	TEXT	IN	No	String to be printed.
is_one_line	BOOLEAN	IN	No	Specifies whether to print the string as a line.

- PKG_UTIL.RAW_GET_LENGTH

  Obtains the length of RAW data.

  The prototype of the PKG_UTIL.RAW_GET_LENGTH function is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
PKG_UTIL.RAW_GET_LENGTH(
value       IN   RAW
)
RETURN INTEGER;
```

Table 2-42 PKG_UTIL.RAW_GET_LENGTH API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
raw	RAW	IN	No	Object whose length is to be obtained.

- PKG_UTIL.RAW_CAST_FROM_VARCHAR2

  Converts VARCHAR2 data to RAW data.

  The prototype of the PKG_UTIL.RAW_CAST_FROM_VARCHAR2 function is as follows:

```
PKG_UTIL.RAW_CAST_FROM_VARCHAR2(
str       IN   VARCHAR2
)
RETURN RAW;
```

Table 2-43 PKG_UTIL.RAW_CAST_FROM_VARCHAR2 API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
str	VARCHAR2	IN	No	Source data to be converted

- PKG_UTIL.RAW_CAST_FROM_BINARY_INTEGER

  Converts BIGINT data to RAW data.

  The prototype of the PKG_UTIL.RAW_CAST_FROM_BINARY_INTEGER function is as follows:

```
PKG_UTIL.RAW_CAST_FROM_BINARY_INTEGER(
value       IN   BIGINT,
endianess   IN   INTEGER
)
RETURN RAW;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-44 PKG_UTIL.RAW_CAST_FROM_BINARY_INTEGER API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
value	BIGINT	IN	No	Source data to be converted.
endianess	INTEGER	IN	No	Endianness, which is an integer. Currently, the value can be **1** (BIG_ENDIAN), **2** (LITTLE_ENDIAN), or **3** (MACHINE_ENDIAN).

- PKG_UTIL.RAW_CAST_TO_BINARY_INTEGER

  Converts RAW data into BINARY_INTEGER.

  The prototype of the PKG_UTIL.RAW_CAST_TO_BINARY_INTEGER function is as follows:

```
PKG_UTIL.RAW_CAST_TO_BINARY_INTEGER(
value       IN   RAW,
endianess   IN   INTEGER
)
RETURN INTEGER;
```

Table 2-45 PKG_UTIL.RAW_CAST_TO_BINARY_INTEGER API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
value	RAW	IN	No	Source data to be converted.
endianess	INTEGER	IN	No	Endianness, which is an integer. Currently, the value can be **1** (BIG_ENDIAN), **2** (LITTLE_ENDIAN), or **3** (MACHINE_ENDIAN).

- PKG_UTIL.RANDOM_SET_SEED

  Sets a random seed.

  The prototype of the PKG_UTIL.RANDOM_SET_SEED function is as follows:

```
PKG_UTIL.RANDOM_SET_SEED(
seed        IN   INT
)
RETURN INTEGER;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-46 PKG_UTIL.RANDOM_SET_SEED API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
seed	INT	IN	No	Random seed.

- PKG_UTIL.RANDOM_GET_VALUE

  Returns a 15-digit random number ranging from 0 to 1.

  The prototype of the PKG_UTIL.RANDOM_GET_VALUE function is as follows:

```
PKG_UTIL.RANDOM_GET_VALUE(
)
RETURN NUMERIC;
```

  NOTE

  You are advised not to use the PKG_UTIL.RANDOM_SET_SEED and PKG_UTIL.RANDOM_GET_VALUE functions in security scenarios.

- PKG_UTIL.FILE_SET_DIRNAME

  Sets the directory to be operated. It must be called to set a directory for each operation involving a single directory.

  The prototype of the PKG_UTIL.FILE_SET_DIRNAME function is as follows:

```
PKG_UTIL.FILE_SET_DIRNAME(
dir  IN  TEXT
)
RETURN BOOL
```

Table 2-47 PKG_UTIL.FILE_SET_DIRNAME API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
dir	TEXT	IN	No	Directory of a file. It is a string, indicating an object name. **NOTE** File directories need to be added to the system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. Functions that take **location** as parameters also comply with this rule.

- PKG_UTIL.FILE_OPEN

  Opens a file. A maximum of 50 files can be opened at a time. This function returns a handle of the INTEGER type.

  The prototype of the PKG_UTIL.FILE_OPEN function is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
PKG_UTIL.FILE_OPEN(
file_name    IN  TEXT,
open_mode    IN  INTEGER)
```

Table 2-48 PKG_UTIL.FILE_OPEN API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file_name	TEXT	IN	No	File name with an extension (file type), excluding the path name. A path contained in a file name is ignored in the OPEN function. In Unix, the file name cannot end with the combination of a slash and a dot (/.).
open_mode	INTEGER	IN	No	File opening mode, including **r** (read text), **w** (write text), and **a** (append text). **NOTE** For the write operation, the system checks the file type. If the ELF file is written, an error will be reported and the system exits.

- PKG_UTIL.FILE_SET_MAX_LINE_SIZE

  Sets the maximum length of a line to be written to a file.

  The prototype of the PKG_UTIL.FILE_SET_MAX_LINE_SIZE function is as follows:

  ```
  PKG_UTIL.FILE_SET_MAX_LINE_SIZE(
  max_line_size IN INTEGER)
  RETURN BOOL
  ```

  Table 2-49 PKG_UTIL.FILE_SET_MAX_LINE_SIZE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
max_line_size	INTEGER	IN	Yes	Maximum number of characters in each line, including newline characters. The minimum value is **1** and the maximum value is **32767**. If this parameter is not specified, the default value **1024** is used.

- PKG_UTIL.FILE_IS_CLOSE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Checks whether a file handle is closed.

The prototype of the PKG_UTIL.FILE_IS_CLOSE function is as follows:

```
PKG_UTIL.FILE_IS_CLOSE(
file IN INTEGER
)
RETURN BOOL
```

Table 2-50 PKG_UTIL.FILE_IS_CLOSE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER	IN	No	Opened file handle.

- PKG_UTIL.FILE_READ

  Reads a line of data from an open file handle based on the specified length.

  The prototype of the PKG_UTIL.FILE_READ function is as follows:

  ```
  PKG_UTIL.FILE_READ(
  file     IN   INTEGER,
  buffer   OUT  TEXT,
  len      IN   BIGINT DEFAULT 1024)
  ```

  Table 2-51 PKG_UTIL.FILE_READ API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER	IN	No	File handle opened by calling the OPEN function. The file must be opened in read mode. Otherwise, the INVALID_OPERATION exception is thrown.
buffer	TEXT	OUT	No	Buffer used to receive data.
len	BIGINT	IN	No	Number of bytes read from a file.

- PKG_UTIL.FILE_READLINE

  Reads a line of data from an open file handle based on the specified length.

  The prototype of the PKG_UTIL.FILE_READLINE function is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
PKG_UTIL.FILE_READLINE(
file    IN  INTEGER,
buffer  OUT TEXT,
len     IN  INTEGER DEFAULT NULL)
```

Table 2-52 PKG_UTIL.FILE_READLINE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER	IN	No	File handle opened by calling the OPEN function. The file must be opened in read mode. Otherwise, the INVALID_OPERATION exception is thrown.
buffer	TEXT	OUT	No	Buffer used to receive data.
len	INTEGER	IN	Yes	Number of bytes read from a file. The default value is **NULL**. If the default value **NULL** is used, **max_line_size** is used to specify the line size.

- PKG_UTIL.FILE_WRITE

  Writes the data specified in the buffer to a file.

  The prototype of the PKG_UTIL.FILE_WRITE function is as follows:

  ```
  PKG_UTIL.FILE_WRITE(
  file IN INTEGER,
  buffer IN TEXT
  )
  RETURN BOOL
  ```

  Table 2-53 PKG_UTIL.FILE_WRITE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER	IN	No	Opened file handle.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
buffer	TEXT	IN	No	Text data to be written to a file. The maximum buffer size is 32767 bytes. If no value is specified, the default value is 1024 bytes. Before the writing is performed, the buffer occupied by PUT operations cannot exceed 32767 bytes. **NOTE** For the write operation, the system checks the file type. If the ELF file is written, an error will be reported and the system exits.

- PKG_UTIL.FILE_NEWLINE

  Writes a line terminator to an open file. The line terminator is related to the platform.

  The prototype of the PKG_UTIL.FILE_NEWLINE function is as follows:

  ```
  PKG_UTIL.FILE_NEWLINE(
  file IN INTEGER
  )
  RETURN BOOL
  ```

  Table 2-54 PKG_UTIL.FILE_NEWLINE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER	IN	No	Opened file handle.

- PKG_UTIL.FILE_WRITELINE

  Writes a line to a file.

  The prototype of the PKG_UTIL.FILE_WRITELINE function is as follows:

  ```
  PKG_UTIL.FILE_WRITELINE(
  file IN INTEGER,
  buffer IN TEXT
  )
  RETURN BOOL
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-55 PKG_UTIL.FILE_WRITELINE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER	IN	No	Opened file handle.
buffer	TEXT	IN	No	Content to be written.

- PKG_UTIL.FILE_READ_RAW

  Reads binary data of a specified length from an open file handle and returns the read binary data of the RAW type.

  The prototype of the PKG_UTIL.FILE_READ_RAW function is as follows:

```
PKG_UTIL.FILE_READ_RAW(
file      IN INTEGER,
length    IN INTEGER DEFAULT NULL
)
RETURN RAW
```

Table 2-56 PKG_UTIL.FILE_READ_RAW API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER	IN	No	Opened file handle.
length	INTEGER	IN	Yes	Length of the data to be read. The default value is **NULL**. By default, all data in the file is read. The maximum size is 1 GB.

- PKG_UTIL.FILE_WRITE_RAW

  Writes the input binary object of the RAW type to an open file. If the insertion is successful, **true** is returned.

  The prototype of the PKG_UTIL.FILE_WRITE_RAW function is as follows:

```
PKG_UTIL.FILE_WRITE_RAW(
file IN INTEGER,
r    IN RAW
)
RETURN BOOL
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-57 PKG_UTIL.FILE_WRITE_RAW API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER	IN	No	Opened file handle.
r	RAW	IN	No	Data to be written to the file. **NOTE** For the write operation, the system checks the file type. If the ELF file is written, an error will be reported and the system exits.

- PKG_UTIL.FILE_FLUSH

  Adds a line terminator to data in the buffer to write data in a file handle into a physical file. You can flush data if a file can only be read when it is opened. For example, debugging information can be flushed to a file so that it can be read immediately.

  The prototype of the PKG_UTIL.FILE_FLUSH function is as follows:

  ```
  PKG_UTIL.FILE_FLUSH(
  file IN INTEGER
  )
  RETURN VOID
  ```

  Table 2-58 PKG_UTIL.FILE_FLUSH API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER	IN	No	Opened file handle.

- PKG_UTIL.FILE_CLOSE

  Closes a file handle.

  The prototype of the PKG_UTIL.FILE_CLOSE function is as follows:

  ```
  PKG_UTIL.FILE_CLOSE(
  file IN INTEGER
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
)
RETURN BOOL
```

Table 2-59 PKG_UTIL.FILE_CLOSE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER	IN	No	Opened file handle.

- PKG_UTIL.FILE_REMOVE

  Removes a disk file. To perform this operation, you must have required permissions.

  The prototype of the PKG_UTIL.FILE_REMOVE function is as follows:

  ```
  PKG_UTIL.FILE_REMOVE(
  file_name IN TEXT
  )
  RETURN VOID
  ```

  Table 2-60 PKG_UTIL.FILE_REMOVE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file_name	TEXT	IN	No	Name of the file to be removed.

- PKG_UTIL.FILE_RENAME

  Renames a file on the disk, similar to **mv** in Unix.

  The prototype of the PKG_UTIL.FILE_RENAME function is as follows:

  ```
  PKG_UTIL.FILE_RENAME(
  src_dir IN TEXT,
  src_file_name IN TEXT,
  dest_dir IN TEXT,
  dest_file_name IN TEXT,
  overwrite BOOLEAN DEFAULT false)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-61 PKG_UTIL.FILE_RENAME API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
src_dir	TEXT	IN	No	Source file directory (case-sensitive). **NOTE** • Location of the file directory, which needs to be added to system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. • When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to operate files in the file path specified by **safe_data_path**.
src_file_name	TEXT	IN	No	Source file name.
dest_dir	TEXT	IN	No	Target file directory (case-sensitive). **NOTE** • Location of the file directory, which needs to be added to system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. • When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to operate files in the file path specified by **safe_data_path**.
dest_file_name	TEXT	IN	No	Target file name.
overwrite	BOOLEAN	IN	Yes	If a file with the same name exists in the destination directory, the file will not be rewritten. The default value is **false**.

- PKG_UTIL.FILE_SIZE

  Returns the size of a specified file.

  The prototype of the PKG_UTIL.FILE_SIZE function is as follows:

```
bigint PKG_UTIL.FILE_SIZE(
file_name IN TEXT
)return BIGINT
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-62 PKG_UTIL.FILE_SIZE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
file_name	TEXT	IN	No	File name.

- PKG_UTIL.FILE_BLOCK_SIZE

  Returns the number of blocks contained in a specified file.

  The prototype of the PKG_UTIL.FILE_BLOCK_SIZE function is as follows:

  ```
  bigint PKG_UTIL.FILE_BLOCK_SIZE(
  file_name IN TEXT
  )return BIGINT
  ```

  Table 2-63 PKG_UTIL.FILE_BLOCK_SIZE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
file_name	TEXT	IN	No	File name.

- PKG_UTIL.FILE_EXISTS

  Checks whether a file exists.

  The prototype of the PKG_UTIL.FILE_EXISTS function is as follows:

  ```
  PKG_UTIL.FILE_EXISTS(
  file_name IN TEXT
  )
  RETURN BOOL
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-64 PKG_UTIL.FILE_EXISTS API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
file_name	TEXT	IN	No	File name.

- PKG_UTIL.FILE_GETPOS

  Specifies the offset of a returned file, in bytes.

  The prototype of the PKG_UTIL.FILE_GETPOS function is as follows:

```
PKG_UTIL.FILE_GETPOS(
file IN INTEGER
)
RETURN BIGINT
```

Table 2-65 PKG_UTIL.FILE_GETPOS API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
file	INTEGER	IN	No	Opened file handle.

- PKG_UTIL.FILE_SEEK

  Adjusts the position of a file pointer forward or backward based on the specified number of bytes.

  The prototype of the PKG_UTIL.FILE_SEEK function is as follows:

```
void PKG_UTIL.FILE_SEEK(
file IN INTEGER,
start IN BIGINT
)
RETURN VOID
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-66 PKG_UTIL.FILE_SEEK API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER	IN	No	Opened file handle.
start	BIGINT	IN	No	File offset, in bytes.

- PKG_UTIL.FILE_CLOSE_ALL

  Closes all file handles opened in a session. This function does not have input or output parameters.

  The prototype of the PKG_UTIL.FILE_CLOSE_ALL function is as follows:

  ```
  PKG_UTIL.FILE_CLOSE_ALL(
  )
  RETURN VOID
  ```

- PKG_UTIL.EXCEPTION_REPORT_ERROR

  Throws an exception.

  The prototype of the PKG_UTIL.EXCEPTION_REPORT_ERROR function is as follows:

  ```
  PKG_UTIL.EXCEPTION_REPORT_ERROR(
  code INTEGER,
  log TEXT,
  flag BOOLEAN DEFAULT FALSE
  )
  RETURN INTEGER
  ```

Table 2-67 PKG_UTIL.EXCEPTION_REPORT_ERROR API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
code	Integer	IN	No	Error code displayed when a running exception occurs.
log	TEXT	IN	No	Log displayed when a running exception occurs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
flag	BOOLEAN	IN	Yes	Reserved. The default value is **FALSE**.

- PKG_UTIL.APP_READ_CLIENT_INFO

  Reads the client information.

  The prototype of the PKG_UTIL.APP_READ_CLIENT_INFO function is as follows:

  ```
  PKG_UTIL.APP_READ_CLIENT_INFO(
  OUT buffer TEXT
  )return TEXT
  ```

  Table 2-68 PKG_UTIL.APP_READ_CLIENT_INFO API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
buffer	TEXT	OUT	No	Client information returned.

- PKG_UTIL.APP_SET_CLIENT_INFO

  Sets the client information.

  The prototype of the PKG_UTIL.APP_SET_CLIENT_INFO function is as follows:

  ```
  PKG_UTIL.APP_SET_CLIENT_INFO(
  str TEXT
  )
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-69 PKG_UTIL.APP_SET_CLIENT_INFO API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
str	TEXT	IN	No	Client information to be set.

- PKG_UTIL.LOB_CONVERTTOBLOB

  Converts a CLOB to a BLOB. *amount* indicates the conversion length.

  The prototype of the PKG_UTIL.LOB_CONVERTTOBLOB function is as follows:

```
PKG_UTIL.LOB_CONVERTTOBLOB(
dest_lob BLOB,
src_clob CLOB,
amount INTEGER,
dest_offset INTEGER,
src_offset INTEGER
)return RAW
```

Table 2-70 PKG_UTIL.LOB_CONVERTTOBLOB API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
dest_lob	BLOB	IN	No	Target LOB.
src_clob	CLOB	IN	No	CLOB to be converted.
amount	INTEGER	IN	No	Length to be converted. If the length is less than 1 or greater than 32767, an error is reported.
dest_offset	INTEGER	IN	No	Start position of the target LOB. If the offset is less than 1 or greater than 1073741312, an error is reported.
src_offset	INTEGER	IN	No	Start position of the source CLOB. If the offset is less than 1 or greater than 1073741312, an error is reported.

- PKG_UTIL.LOB_CONVERTTOCLOB

  Converts a BLOB to a CLOB. *amount* indicates the conversion length.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The prototype of the PKG_UTIL.LOB_CONVERTTOCLOB function is as follows:

```
PKG_UTIL.LOB_CONVERTTOCLOB(
dest_lob CLOB,
src_blob BLOB,
amount INTEGER,
dest_offset INTEGER,
src_offset INTEGER
)return TEXT
```

Table 2-71 PKG_UTIL.LOB_CONVERTTOCLOB API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
dest_lob	CLOB	IN	No	Target LOB.
src_blob	BLOB	IN	No	BLOB to be converted.
amount	INTEGER	IN	No	Length to be converted. If the length is less than 1 or greater than 32767, an error is reported.
dest_offset	INTEGER	IN	No	Start position of the target LOB. If the offset is less than 1 or greater than 1073741312, an error is reported.
src_offset	INTEGER	IN	No	Start position of the source CLOB. If the offset is less than 1 or greater than 1073741312, an error is reported.

- PKG_UTIL.LOB_TEXTTORAW

  Converts the input CLOB text into a RAW output, automatically prefixing a leading zero for odd-length inputs. The input text must consist of valid hexadecimal digits (0-9, A-F, a-f). If any invalid characters are included, an error will be reported.

  The prototype of the PKG_UTIL.LOB_TEXTTORAW function is as follows:

  ```
  PKG_UTIL.LOB_TEXTTORAW(
  src_lob CLOB
  )
  RETURN RAW
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-72 PKG_UTIL.LOB_TEXTTORAW API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
src_lob	CLOB	IN	No	LOB to be converted.

- PKG_UTIL.LOB_RAWTOTEXT

  Converts the input BLOB into a hexadecimal text output, automatically prefixing a leading zero for odd-length inputs. The input must consist of valid hexadecimal digits (0-9, A-F, a-f). If any invalid characters are included, an error will be reported.

  The prototype of the PKG_UTIL.LOB_RAWTOTEXT function is as follows:

```
PKG_UTIL.LOB_RAWTOTEXT(
src_lob IN BLOB
)
RETURN TEXT
```

Table 2-73 PKG_UTIL.LOB_RAWTOTEXT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
src_lob	BLOB	IN	No	LOB to be converted.

- PKG_UTIL.MATCH_EDIT_DISTANCE_SIMILARITY

  Calculates the difference between two character strings.

  The prototype of the PKG_UTIL.MATCH_EDIT_DISTANCE_SIMILARITY function is as follows:

```
PKG_UTIL.MATCH_EDIT_DISTANCE_SIMILARITY(
str1 TEXT,
str2 TEXT
)
RETURN INTEGER
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-74 PKG_UTIL.MATCH_EDIT_DISTANCE_SIMILARITY API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
str1	TEXT	IN	No	First character string.
str2	TEXT	IN	No	Second character string.

- PKG_UTIL.RAW_CAST_TO_VARCHAR2

  Converts the raw type to the varchar2 type.

  The prototype of the PKG_UTIL.RAW_CAST_TO_VARCHAR2 function is as follows:

```
PKG_UTIL.RAW_CAST_TO_VARCHAR2(
str RAW
)
RETURN VARCHAR2
```

Table 2-75 PKG_UTIL.RAW_CAST_TO_VARCHAR2 API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
str	RAW	IN	No	Hexadecimal string.

- PKG_UTIL.SESSION_CLEAR_CONTEXT

  Clears the session context.

  The prototype of the PKG_UTIL.SESSION_CLEAR_CONTEXT function is as follows:

```
PKG_UTIL.SESSION_CLEAR_CONTEXT(
namespace TEXT,
client_identifier TEXT,
attribute TEXT
)
RETURN INTEGER
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-76 PKG_UTIL.SESSION_CLEAR_CONTEXT API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
namespace	TEXT	IN	No	Namespace of an attribute.
client_identifier	TEXT	IN	Yes	Usually same as the value of **namespace**. If this parameter is set to **null**, all namespaces are modified by default.
attribute	TEXT	IN	No	Attribute value to be cleared.

- PKG_UTIL.SESSION_SEARCH_CONTEXT

  Searches for an attribute value.

  The prototype of the PKG_UTIL.SESSION_SEARCH_CONTEXT function is as follows:

```
PKG_UTIL.SESSION_SEARCH_CONTEXT(
namespace TEXT,
attribute TEXT
)
RETURN INTEGER
```

Table 2-77 PKG_UTIL.SESSION_SEARCH_CONTEXT API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
namespace	TEXT	IN	No	Namespace of an attribute.
attribute	TEXT	IN	No	Attribute value to be searched for.

- PKG_UTIL.SESSION_SET_CONTEXT

  Sets an attribute value.

  The prototype of the PKG_UTIL.SESSION_SET_CONTEXT function is as follows:

```
PKG_UTIL.SESSION_SET_CONTEXT(
namespace TEXT,
attribute TEXT,
value TEXT
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
)
RETURN INTEGER
```

Table 2-78 PKG_UTIL.SESSION_SET_CONTEXT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
namespace	TEXT	IN	No	Namespace of an attribute.
attribute	TEXT	IN	No	Attribute to be set.
value	TEXT	IN	No	Attribute value.

- PKG_UTIL.UTILITY_GET_TIME

  Prints the Unix timestamp.

  The prototype of the PKG_UTIL.UTILITY_GET_TIME function is as follows:

  ```
  PKG_UTIL.UTILITY_GET_TIME()
  RETURN BIGINT
  ```

- PKG_UTIL.UTILITY_FORMAT_ERROR_BACKTRACE

  Displays the error stack of a stored procedure.

  The prototype of the PKG_UTIL.UTILITY_FORMAT_ERROR_BACKTRACE function is as follows:

  ```
  PKG_UTIL.UTILITY_FORMAT_ERROR_BACKTRACE()
  RETURN TEXT
  ```

- PKG_UTIL.UTILITY_FORMAT_ERROR_STACK

  Displays the error information about a stored procedure.

  The prototype of the PKG_UTIL.UTILITY_FORMAT_ERROR_STACK function is as follows:

  ```
  PKG_UTIL.UTILITY_FORMAT_ERROR_STACK()
  RETURN TEXT
  ```

- PKG_UTIL.UTILITY_FORMAT_CALL_STACK

  Displays the call stack of a stored procedure.

  The prototype of the PKG_UTIL.UTILITY_FORMAT_CALL_STACK function is as follows:

  ```
  PKG_UTIL.UTILITY_FORMAT_CALL_STACK()
  RETURN TEXT
  ```

- PKG_UTIL.UTILITY_COMPILE_SCHEMA

  Recompiles packages, functions, and stored procedures in the specified schema.

  The prototype of the PKG_UTIL.UTILITY_COMPILE_SCHEMA function is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
PKG_UTIL.UTILITY_COMPILE_SCHEMA (
schema IN VARCHAR2,
compile_all IN BOOLEAN DEFAULT TRUE,
reuse_settings IN BOOLEAN DEFAULT FALSE
)
RETURNS VOID
```

- PKG_UTIL.GS_COMPILE_SCHEMA

  Recompiles packages, functions, and stored procedures in the specified schema.

  The prototype of the PKG_UTIL.GS_COMPILE_SCHEMA stored procedure is as follows:

  ```
  pkg_util.GS_COMPILE_SCHEMA (
  schema_name IN VARCHAR2 DEFAULT NULL,
  compile_all IN BOOLEAN DEFAULT FALSE,
  retry_times IN INT DEFAULT 10
  )
  ```

  Table 2-79 PKG_UTIL.GS_COMPILE_SCHEMA API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
schema_name	VARCHAR2	IN	Yes	Schema name.
compile_all	Boolean	IN	Yes	Specifies whether all objects are compiled. ● **false**: The packages, functions, and stored procedures whose status is **false** in the pg_object table are compiled. ● **true**: All packages, functions, and stored procedures in the pg_object table are compiled. ● **retry_times**: number of retry times.

- PKG_UTIL.APP_SET_MODULE

  Sets the value of a module.

  The prototype of the PKG_UTIL.APP_SET_MODULE stored procedure is as follows:

  ```
  PKG_UTIL.APP_SET_MODULE (
     str TEXT)
  returns VOID
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-80 PKG_UTIL.APP_SET_MODULE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
text	Text	IN	No	Value of the module to be set.

Example:

```
CALL PKG_UTIL.APP_SET_MODULE('set module');
 app_set_module
----------------

(1 row)
```

- PKG_UTIL.APP_READ_MODULE

  Reads the value of a module.

  The prototype of the PKG_UTIL.APP_READ_MODULE stored procedure is as follows:

```
PKG_UTIL.APP_READ_MODULE(
OUT buffer TEXT
)
```

Table 2-81 PKG_UTIL.APP_READ_MODULE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
buffer	TEXT	OUT	No	Value of the module to be returned.

Example:

```
CALL PKG_UTIL.APP_SET_MODULE('set module');
 app_set_module
----------------

(1 row)

DECLARE
    module VARCHAR2(64);
BEGINPKG_UTIL.APP_READ_MODULE(module);
    dbe_output.print_line(module);
END;
/
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
set module
ANONYMOUS BLOCK EXECUTE
```

- PKG_UTIL.APP_SET_ACTION

  Sets the value of an action.

  The prototype of the PKG_UTIL.APP_SET_ACTION stored procedure is as follows:

```
PKG_UTIL.APP_SET_ACTION (
    str TEXT)
returns VOID
```

Table 2-82 PKG_UTIL.APP_SET_ACTION API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
text	Text	IN	No	Value of the action to be set.

Example:

```
CALL PKG_UTIL.APP_SET_ACTION('set action');
 app_set_action
----------------

(1 row)
```

- PKG_UTIL.APP_READ_ACTION

  Reads the value of an action.

  The prototype of the PKG_UTIL.APP_READ_ACTION stored procedure is as follows:

```
PKG_UTIL.APP_READ_ACTION(
OUT buffer TEXT
)
```

Table 2-83 PKG_UTIL.APP_READ_ACTION API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
buffer	TEXT	OUT	No	Value of the action to be returned.

Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
CALL PKG_UTIL.APP_SET_ACTION('set action');
 app_set_action
----------------

(1 row)

DECLARE
       action VARCHAR2(64);
BEGINPKG_UTIL.APP_READ_ACTION(action);
      dbe_output.print_line(action);
END;
/
set action
ANONYMOUS BLOCK EXECUTE
```

- PKG_UTIL.MODIFY_PACKAGE_STATE

  Modifies the PL/SQL status of the current session.

  The prototype of the PKG_UTIL.MODIFY_PACKAGE_STATE stored procedure is as follows:

```
PKG_UTIL.MODIFY_PACKAGE_STATE (
    flags INT)
returns VOID
```

Table 2-84 PKG_UTIL.MODIFY_PACKAGE_STATE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
flags	INT	IN	No	Bit flag of the operation performed on the PL/SQL. When **flags** is set to **1**, the session cache of the PL/SQL that is currently running in the session is released after the top-level PL/SQL is executed. The current value of any package is cleared globally and the cached cursor is closed. On subsequent use, PL/SQL re-instantiates and re-initializes the package globally. Other bit flags are not supported.

Example:

```
CALL PKG_UTIL.MODIFY_PACKAGE_STATE(1);
 modify_package_state
----------------------

(1 row)
```

2.12.1.3 DBE_XML

Table 2-85 lists all APIs supported by DBE_XML.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-85 DBE_XML API parameters

API	Description
DBE_XML.XML_FREE_PARSER	Frees a parser.
DBE_XML.XML_PARSER_GET_DOC	Obtains the parsed document node.
DBE_XML.XML_GET_VALIDATION_MODE	Obtains the validation attribute.
DBE_XML.XML_NEW_PARSER	Creates a parser instance.
DBE_XML.XML_PARSE_BUFFER	Parses the VARCHAR string.
DBE_XML.XML_PARSE_CLOB	Parses the CLOB string.
DBE_XML.XML_SET_VALIDATION_MODE	Sets the validation attribute.
DBE_XML.XML_DOM_APPEND_CHILD	Adds the newchild node to the end of the parent(n) node and returns the newly added node.
DBE_XML.XML_DOM_CREATE_ELEMENT	Returns the DOMElement object with the specified name.
DBE_XML.XML_DOM_CREATE_ELEMENT_NS	Returns the DOMElement object with the specified name and namespace.
DBE_XML.XML_DOM_CREATE_TEXT_NODE	Creates and returns a DOMText object.
DBE_XML.XML_DOM_FREE_DOCUMENT	Frees a specified XML DOM object.
DBE_XML.XML_DOM_FREE_ELEMENT	Frees a specified XML DOM object.
DBE_XML.XML_DOM_FREE_NODE	Frees a DOMNode node.
DBE_XML.XML_DOM_FREE_NODELIST	Frees a DOMNodeList node.
DBE_XML.XML_DOM_GET_ATTRIBUTE	Obtains the attributes of a specified XML DOM object.
DBE_XML.XML_DOM_GET_ATTRIBUTES	Returns the attribute values of a DOMNode node as a map.
DBE_XML.XML_DOM_GET_CHILD_NODES	Converts several subnodes under a node into a node list.
DBE_XML.XML_DOM_GET_CHILDREN_BY_TAGNAME	Obtains the list of specified subnodes of a specified XML DOM object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_XML.XML_DOM_GET_CHILDREN_BY_TAGNAME_NS	Obtains the list of subnodes in the specified namespace of a specified XML DOM object.
DBE_XML.XML_DOM_GET_DOCUMENT_ELEMENT	Returns the first subnode of the specified document.
DBE_XML.XML_DOM_GET_FIRST_CHILD	Returns the first subnode of a node.
DBE_XML.XML_DOM_GET_LAST_CHILD	Returns the last subnode of a node.
DBE_XML.XML_DOM_GET_LENGTH	Returns the number of nodes based on the content in the node of the specified type.
DBE_XML.XML_DOM_GET_LOCALNAME	Returns the local name of the given object.
DBE_XML.XML_DOM_GET_NAMED_ITEM	Returns the node specified by name.
DBE_XML.XML_DOM_GET_NAMED_ITEM_NS	Returns the node specified by name and namespace.
DBE_XML.XML_DOM_GET_NEXT_SIBLING	Returns the next node of the specified node.
DBE_XML.XML_DOM_GET_NODE_NAME	Returns the name of a node.
DBE_XML.XML_DOM_GET_NODE_TYPE	Returns the type of a node.
DBE_XML.XML_DOM_GET_NODE_VALUE	Returns the value of a node.
DBE_XML.XML_DOM_GET_PARENT_NODE	Returns the parent node of the given node.
DBE_XML.XML_DOM_GET_TAGNAME	Obtains the tag name of a specified XML DOM object.
DBE_XML.XML_DOM_HAS_CHILD_NODES	Checks whether the DOMNode object has any subnode.
DBE_XML.XML_DOM_IMPORT_NODE	Copies a node to another node and mounts the copied node to a specified document.
DBE_XML.XML_DOM_IS_NULL	Checks whether the given object is null.
DBE_XML.XML_DOM_ITEM	Returns the element corresponding to the index in a list or map based on the index.
DBE_XML.XML_DOM_MAKE_ELEMENT	Returns the DOMElement object after conversion.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_XML.XML_DOM_MAKENODE	Converts the given object to the DOMNode type.
DBE_XML.XML_DOM_NEW_DOM_DOCUMENT_EMPTY	Returns a new DOMDocument object.
DBE_XML.XML_DOM_NEW_DOM_DOCUMENT_CLOB	Returns a new DOMDocument instance object created from the specified CLOB type.
DBE_XML.XML_DOM_NEW_DOCUMENT_XMLTYPE	Returns a new DOMDocument instance object created from the specified XMLType type.
DBE_XML.XML_DOM_SET_ATTRIBUTE	Sets the attributes of a specified XML DOM object.
DBE_XML.XML_DOM_SET_CHARSET	Sets the character set for a DOMDocument object.
DBE_XML.XML_DOM_SET_DOCTYPE	Sets the external DTD of a DOMDocument object.
DBE_XML.XML_DOM_SET_NODE_VALUE	Sets the value of a node in the DOMNode object.
DBE_XML.XML_DOM_WRITE_TO_BUFFER_DOC	Writes the given DOMDocument object to the buffer.
DBE_XML.XML_DOM_WRITE_TO_BUFFER_NODE	Writes the given DOMNode object to the buffer.
DBE_XML.XML_DOM_WRITE_TO_CLOB_DOC	Writes the given DOMDocument object to a CLOB.
DBE_XML.XML_DOM_WRITE_TO_CLOB_NODE	Writes the given DOMNode object to a CLOB.
DBE_XML.XML_DOM_WRITE_TO_FILE_DOC	Writes an XML node to a specified file using the database character set.
DBE_XML.XML_DOM_WRITE_TO_FILE_NODE	Writes an XML node to a specified file using the database character set.
DBE_XML.XML_DOM_GET_SESSION_TREE_NUM	Displays the number of DOM trees of all types in the current session.
DBE_XML.XML_DOM_GET_DOC_TREES_INFO	Displays statistics such as the memory usage and number of nodes of the DOM tree of the document type.
DBE_XML.XML_DOM_GET_DETAIL_DOC_TREE_INFO	Displays the number of nodes of each type for a specific document variable.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_XML.XML_FREE_PARSER

  Frees a given parser object.

  The prototype of the DBE_XML.XML_FREE_PARSER stored procedure is as follows:

  ```
  DBE_XML.XML_FREE_PARSER(
  id IN RAW(13))
  returns VOID;
  ```

  Table 2-86 DBE_XML.XML_FREE_PARSER API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	Parser object.

- DBE_XML.XML_PARSER_GET_DOC

  Returns the root node of the DOM tree document constructed by the parser.

  The prototype of the DBE_XML.XML_PARSER_GET_DOC function is as follows:

  ```
  DBE_XML.XML_PARSER_GET_DOC(
  id IN RAW(13))
  returns RAW(13);
  ```

  Table 2-87 DBE_XML.XML_PARSER_GET_DOC API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	Parser object.

  NOTE

  - If the DBE_XML.XML_PARSER_GET_DOC function is empty, **NULL** is returned.
  - If the parser input by the DBE_XML.XML_PARSER_GET_DOC function has not parsed any document, **NULL** is returned.

- DBE_XML.XML_GET_VALIDATION_MODE

  Obtains the parsing validation mode of a specified parser. If DTD validation is enabled, **TRUE** is returned. Otherwise, **FALSE** is returned.

  The prototype of the DBE_XML.XML_GET_VALIDATION_MODE function is as follows:

  ```
  DBE_XML.XML_GET_VALIDATION_MODE(
  id RAW(13))
  returns BOOL;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-88 DBE_XML.XML_GET_VALIDATION_MODE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	Parser object.

- DBE_XML.XML_NEW_PARSER

  Creates a parser object and returns a new parser instance.

  The prototype of the DBE_XML.XML_NEW_PARSER function is as follows:

  ```
  DBE_XML.XML_NEW_PARSER()
  RETURNS RAW(13);
  ```

- DBE_XML.XML_PARSE_BUFFER

  Parses XML documents stored in strings.

  The prototype of the DBE_XML.XML_PARSE_BUFFER stored procedure is as follows:

  ```
  DBE_XML.XML_PARSE_BUFFER(
  id RAW(13),
  xmlstr VARCHAR2)
  RETURNS VOID;
  ```

Table 2-89 DBE_XML.XML_PARSE_BUFFER API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	Parser object.
xmlstr	VARCHAR2	IN	No	A string that stores XML documents.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The maximum length of a character string that can be parsed by the xml_parse_buffer function is 32767. If the length exceeds the maximum, an error is reported.
- Different from the ORA database, this database supports only UTF-8 in terms of character encoding, and **version** can only be set to **1.0**. If versions 1.0 to 1.9 are parsed, the execution is normal but a warning appears. For versions later than 1.9, an error is reported.
- DTD validation differences:
  - **!ATTLIST to type (CHECK|check|Check) "Ch..."** reports an error because the default value **"Ch..."** is not an enumerated value in the brackets. However, the ORA database does not report this error.
  - **<!ENTITY baidu "www.baidu.com">..... &Baidu;&writer** reports an error because the letters are case-sensitive. **Baidu** cannot correspond to **baidu**. However, the ORA database does not report this error.
- Namespace validation difference: Undeclared namespace tags are parsed. However, the ORA database reports an error.
- Difference in parsing XML predefined entities: **'** and **"** are parsed and escaped to ' and ". However, predefined entities in database ORA are not escaped to characters.

- DBE_XML.XML_PARSE_CLOB

  Parses XML documents stored in XML_PARSE_CLOB.

  The prototype of the DBE_XML.XML_PARSE_CLOB stored procedure is as follows:

```
DBE_XML.XML_PARSE_CLOB(
id IN RAW(13),
doc IN CLOB)
returns VOID;
```

Table 2-90 DBE_XML.XML_PARSE_CLOB API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	Parser object.
doc	CLOB	IN	No	A string that stores XML documents.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- xml_parse_clob cannot parse CLOBs larger than 1 GB.
- Different from the ORA database, this database supports only UTF-8 in terms of character encoding, and **version** can only be set to **1.0**. If versions 1.0 to 1.9 are parsed, the execution is normal but a warning appears. For versions later than 1.9, an error is reported.
- DTD validation differences:
  - **!ATTLIST to type (CHECK|check|Check) "Ch..."** reports an error because the default value **"Ch..."** is not an enumerated value in the brackets. However, the ORA database does not report this error.
  - **<!ENTITY baidu "www.baidu.com">..... &Baidu;&writer** reports an error because the letters are case-sensitive. **Baidu** cannot correspond to **baidu**. However, the ORA database does not report this error.
- Namespace validation difference: Undeclared namespace tags are parsed. However, the ORA database reports an error.
- Difference in parsing XML predefined entities: **'** and **"** are parsed and escaped to ' and ". However, predefined entities in database ORA are not escaped to characters.

- DBE_XML.XML_SET_VALIDATION_MODE

  Sets the parsing validation mode of a specified parser.

  The stored procedure prototype of DBE_XML.XML_SET_VALIDATION_MODE is as follows:

```
DBE_XML.XML_SET_VALIDATION_MODE(
id RAW(13),
validate BOOLEAN)
returns VOID;
```

Table 2-91 DBE_XML.XML_SET_VALIDATION_MODE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	Parser object.
validate	BOOLEAN	IN	Yes	Mode to be set: • **TRUE**: DTD validation is enabled. • **FALSE**: DTD validation is disabled.

NOTE

- If the input parameter **validate** of the DBE_XML.XML_SET_VALIDATION_MODE function is null, the parsing validation mode of the parser is not changed.
- By default, the DTD validation is enabled during parser initialization.

- DBE_XML.XML_DOM_APPEND_CHILD

  Adds the newchild node to the end of the parent(n) node and returns the newly added node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The prototype of the DBE_XML.XML_DOM_APPEND_CHILD stored procedure is as follows:

```
DBE_XML.XML_DOM_APPEND_CHILD(
   parentId IN RAW(13),
   childId IN RAW(13)
)
RETURNS RAW(13);
```

Table 2-92 DBE_XML.XML_DOM_APPEND_CHILD API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
parentId	RAW(13)	IN	No	XML DOM object.
childId	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_CREATE_ELEMENT

  Returns the DOMElement object with the specified name.

  The prototype of the DBE_XML.XML_DOM_CREATE_ELEMENT function is as follows:

```
DBE_XML.XML_DOM_CREATE_ELEMENT(
   id IN RAW(13),
   tagname IN VARCHAR2
)
RETURNS RAW(13);
```

Table 2-93 DBE_XML.XML_DOM_CREATE_ELEMENT API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.
tagname	VARCHAR2	IN	No	Name of the new DOMElement object.

- DBE_XML.XML_DOM_CREATE_ELEMENT_NS

  Returns the DOMElement object with the specified name and namespace.

  The prototype of the DBE_XML.XML_DOM_CREATE_ELEMENT_NS function is as follows:

```
DBE_XML.XML_DOM_CREATE_ELEMENT_NS(
   id IN RAW(13),
   tagname IN VARCHAR2,
   ns IN VARCHAR2
)
RETURNS RAW(13);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-94 DBE_XML.XML_DOM_CREATE_ELEMENT_NS API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.
tagname	VARCHAR2	IN	No	Name of the new DOMElement object.
ns	VARCHAR2	IN	No	Namespace.

- DBE_XML.XML_DOM_CREATE_TEXT_NODE

  Creates and returns a DOMText object.

  The prototype of the DBE_XML.XML_DOM_CREATE_TEXT_NODE function is as follows:

```
DBE_XML.XML_DOM_CREATE_TEXT_NODE(
   id IN RAW(13),
   data IN VARCHAR2
)
RETURNS RAW(13);
```

  Table 2-95 DBE_XML.XML_DOM_CREATE_TEXT_NODE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.
data	VARCHAR2	IN	No	Content of the new DOMText node.

- DBE_XML.XML_DOM_FREE_DOCUMENT

  Frees a specified XML DOM object.

  The prototype of the DBE_XML.XML_DOM_FREE_DOCUMENT stored procedure is as follows:

```
DBE_XML.XML_DOM_FREE_DOCUMENT(
   id RAW(13)
)
RETURNS VOID;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-96 DBE_XML.XML_DOM_FREE_DOCUMENT API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_FREE_ELEMENT

  Frees a specified XML DOM object.

  The prototype of the DBE_XML.XML_DOM_FREE_ELEMENT stored procedure is as follows:

```
DBE_XML.XML_DOM_FREE_ELEMENT (
    id RAW(13)
)
RETURNS VOID;
```

Table 2-97 DBE_XML.XML_DOM_FREE_ELEMENT API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_FREE_NODE

  Frees a DOMNode node.

  The prototype of the DBE_XML.XML_DOM_FREE_NODE function is as follows:

```
DBE_XML.XML_DOM_FREE_NODE (
    id RAW(13)
)
RETURNS VOID;
```

Table 2-98 DBE_XML.XML_DOM_FREE_NODE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_FREE_NODELIST

  Frees a DOMNodeList node.

  The prototype of the DBE_XML.XML_DOM_FREE_NODELIST stored procedure is as follows:

```
DBE_XML.XML_DOM_FREE_NODELIST(
    id IN RAW(13)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
)
RETURNS VOID
```

Table 2-99 DBE_XML.XML_DOM_FREE_NODELIST API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_GET_ATTRIBUTE

  Obtains the attributes of a specified XML DOM object.

  The prototype of the DBE_XML.XML_DOM_GET_ATTRIBUTE stored procedure is as follows:

```
DBE_XML.XML_DOM_GET_ATTRIBUTE (
   docid   IN  RAW(13),
   name    IN  VARCHAR2
)
RETURNS VARCHAR2;
```

Table 2-100 DBE_XML.XML_DOM_GET_ATTRIBUTE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
docid	RAW(13)	IN	No	XML DOM object.
name	VARCHAR2	IN	No	A string.

- DBE_XML.XML_DOM_GET_ATTRIBUTES

  Returns the attribute values of a DOMNode node as a map.

  The prototype of the DBE_XML.XML_DOM_GET_ATTRIBUTES function is as follows:

```
DBE_XML.XML_DOM_GET_ATTRIBUTES (
   id RAW(13)
)
RETURNS RAW(13);
```

Table 2-101 DBE_XML.XML_DOM_GET_ATTRIBUTES API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_GET_CHILD_NODES

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Converts several subnodes under a node into a node list.

The prototype of the DBE_XML.XML_DOM_GET_CHILD_NODES function is as follows:

```
DBE_XML.XML_DOM_GET_CHILD_NODES(
   id IN RAW(13)
)
RETURNS RAW(13);
```

Table 2-102 DBE_XML.XML_DOM_GET_CHILD_NODES API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_GET_CHILDREN_BY_TAGNAME

  Obtains the list of specified subnodes of a specified XML DOM object.

  The prototype of the DBE_XML.XML_DOM_GET_CHILDREN_BY_TAGNAME stored procedure is as follows:

  ```
  DBE_XML.XML_DOM_GET_CHILDREN_BY_TAGNAME (
     docid   IN  RAW(13),
     name    IN  VARCHAR2
  )
  RETURNS RAW(13);
  ```

  Table 2-103 DBE_XML.XML_DOM_GET_CHILDREN_BY_TAGNAME API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
docid	RAW(13)	IN	No	XML DOM object.
name	VARCHAR2	IN	No	A string.

- DBE_XML.XML_DOM_GET_CHILDREN_BY_TAGNAME_NS

  Obtains the list of subnodes in the specified namespace of a specified XML DOM object.

  The prototype of the DBE_XML.XML_DOM_GET_CHILDREN_BY_TAGNAME_NS stored procedure is as follows:

  ```
  DBE_XML.XML_DOM_GET_CHILDREN_BY_TAGNAME_NS (
     docid   IN  RAW(13),
     name    IN  VARCHAR2,
     ns      IN  VARCHAR2
  )
  RETURNS RAW(13);
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-104 DBE_XML.XML_DOM_GET_CHILDREN_BY_TAGNAME_NS API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
docid	RAW(13)	IN	No	XML DOM object.
name	VARCHAR2	IN	No	A string.
ns	VARCHAR2	IN	Yes	A string.

- DBE_XML.XML_DOM_GET_DOCUMENT_ELEMENT

  Returns the first subnode of the specified document.

  The prototype of the DBE_XML.XML_DOM_GET_DOCUMENT_ELEMENT stored procedure is as follows:

```
DBE_XML.XML_DOM_GET_DOCUMENT_ELEMENT(
   id  RAW(13)
)
RETURNS RAW(13);
```

  Table 2-105 DBE_XML.XML_DOM_GET_DOCUMENT_ELEMENT API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_GET_FIRST_CHILD

  Returns the first subnode of a node.

  The prototype of the DBE_XML.XML_DOM_GET_FIRST_CHILD function is as follows:

```
DBE_XML.XML_DOM_GET_FIRST_CHILD(
   id IN RAW(13)
)
RETURNS RAW(13);
```

  Table 2-106 DBE_XML.XML_DOM_GET_FIRST_CHILD API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_GET_LAST_CHILD

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Returns the last subnode of a node.

The prototype of the DBE_XML.XML_DOM_GET_LAST_CHILD function is as follows:

```
DBE_XML.XML_DOM_GET_LAST_CHILD(
    id IN RAW(13)
)
RETURNS RAW(13);
```

Table 2-107 DBE_XML.XML_DOM_GET_LAST_CHILD API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_GET_LENGTH

  Returns the number of nodes based on the content in the node of the specified type.

  The prototype of the DBE_XML.XML_DOM_GET_LENGTH stored procedure is as follows:

  ```
  DBE_XML.XML_DOM_GET_LENGTH(
      id RAW(13)
  )
  RETURNS VOID;
  ```

  Table 2-108 DBE_XML.XML_DOM_GET_LENGTH API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	XML DOM object.

- DBE_XML.XML_DOM_GET_LOCALNAME

  Returns the local name of the given object.

  The prototype of the DBE_XML.XML_DOM_GET_LOCALNAME stored procedure is as follows:

  ```
  DBE_XML.XML_DOM_GET_LOCALNAME (
      id  RAW(13)
  )
  RETURNS VARCHAR2;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-109 DBE_XML.XML_DOM_GET_LOCALNAME API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	XML DOM object.

- DBE_XML.XML_DOM_GET_NAMED_ITEM

  Returns the node specified by name.

  The prototype of the DBE_XML.XML_DOM_GET_NAMED_ITEM function is as follows:

```
DBE_XML.XML_DOM_GET_NAMED_ITEM(
   id IN RAW(13),
   nodeName IN VARCHAR2
)
RETURNS RAW(13);
```

Table 2-110 DBE_XML.XML_DOM_GET_NAMED_ITEM API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.
nodeName	VARCHAR2	IN	No	Name of the element to be retrieved.

- DBE_XML.XML_DOM_GET_NAMED_ITEM_NS

  Returns the node specified by name and namespace.

  The prototype of the DBE_XML.XML_DOM_GET_NAMED_ITEM_NS function is as follows:

```
DBE_XML.XML_DOM_GET_NAMED_ITEM_NS(
   id  RAW(13),
   nodeName IN VARCHAR2,
   ns IN VARCHAR2
)
RETURNS RAW(13);
```

Table 2-111 DBE_XML.XML_DOM_GET_NAMED_ITEM_NS API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
nodeName	VARCHAR2	IN	No	Name of the element to be retrieved.
ns	VARCHAR2	IN	Yes	Namespace.

- DBE_XML.XML_DOM_GET_NEXT_SIBLING

  Returns the next node of the specified node.

  The prototype of the DBE_XML.XML_DOM_GET_NEXT_SIBLING function is as follows:

```
DBE_XML.XML_DOM_GET_NEXT_SIBLING(
    id IN RAW(13)
)
RETURNS RAW(13);
```

  Table 2-112 DBE_XML.XML_DOM_GET_NEXT_SIBLING API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_GET_NODE_NAME

  Returns the name of a node.

  The prototype of the DBE_XML.XML_DOM_GET_NODE_NAME function is as follows:

```
DBE_XML.XML_DOM_GET_NODE_NAME(
    id IN RAW(13)
)
RETURNS VARCHAR2;
```

  Table 2-113 DBE_XML.XML_DOM_GET_NODE_NAME API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_GET_NODE_TYPE

  Returns the type of a node.

  The prototype of the DBE_XML.XML_DOM_GET_NODE_TYPE function is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_XML.XML_DOM_GET_NODE_TYPE(
   id IN RAW(13)
)
RETURNS INTEGER;
```

Table 2-114 DBE_XML.XML_DOM_GET_NODE_TYPE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_GET_NODE_VALUE

  Returns the value of a node.

  The prototype of the DBE_XML.XML_DOM_GET_NODE_VALUE stored procedure is as follows:

  ```
  DBE_XML.XML_DOM_GET_NODE_VALUE(
  id IN RAW(13))
  RETURNS VARCHAR2;
  ```

  Table 2-115 DBE_XML.XML_DOM_GET_NODE_VALUE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	XML DOM object.

- DBE_XML.XML_DOM_GET_PARENT_NODE

  Returns the parent node of the given node.

  The prototype of the DBE_XML.XML_DOM_GET_PARENT_NODE stored procedure is as follows:

  ```
  DBE_XML.XML_DOM_GET_PARENT_NODE(
  id IN RAW(13))
  RETURNS RAW(13);
  ```

  Table 2-116 DBE_XML.XML_DOM_GET_PARENT_NODE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	XML DOM object.

- DBE_XML.XML_DOM_GET_TAGNAME

  Obtains the tag name of a specified XML DOM object.

  The prototype of the DBE_XML.XML_DOM_GET_TAGNAME stored procedure is as follows:

```
DBE_XML.XML_DOM_GET_TAGNAME (
   docid RAW(13)
)
RETURNS VARCHAR2;
```

Table 2-117 DBE_XML.XML_DOM_GET_TAGNAME API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
docid	RAW(13)	IN	Yes	XML DOM object.

- DBE_XML.XML_DOM_HAS_CHILD_NODES

  Checks whether the DOMNode object has any subnode.

  The prototype of the DBE_XML.XML_DOM_HAS_CHILD_NODES stored procedure is as follows:

  ```
  DBE_XML.XML_DOM_HAS_CHILD_NODES(
  id IN RAW(13))
  RETURNS BOOLEAN
  ```

  Table 2-118 DBE_XML.XML_DOM_HAS_CHILD_NODES API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	XML DOM object.

- DBE_XML.XML_DOM_IMPORT_NODE

  Copies a node to another node and mounts the copied node to a specified document. If the type of the copied node does not belong to the 12 types specified by constants of XML DOM, an exception indicating that the type is not supported will be thrown.

  The prototype of the DBE_XML.XML_DOM_IMPORT_NODE function is as follows:

  ```
  DBE_XML.XML_DOM_IMPORT_NODE(
     doc_id IN RAW(13),
     node_id IN RAW(13),
     deep IN BOOLEAN
  )
  RETURNS RAW(13);
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-119 DBE_XML.XML_DOM_IMPORT_NODE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
doc_id	RAW(13)	IN	No	Document to which the node is mounted.
node_id	RAW(13)	IN	No	Node to be imported.
deep	BOOLEAN	IN	No	Specifies whether to perform recursive import. • If the value is **TRUE**, the node and all its subnodes are imported. • If the value is **FALSE**, the node itself is imported.

- DBE_XML.XML_DOM_IS_NULL

  Checks whether the given object is null. If it is null, **TRUE** is returned. Otherwise, **FALSE** is returned.

  The prototype of the DBE_XML.XML_DOM_IS_NULL function is as follows:

```
DBE_XML.XML_DOM_IS_NULL (
   id  RAW(13)
)
RETURNS BOOLEAN;
```

Table 2-120 DBE_XML.XML_DOM_IS_NULL API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	XML DOM object.

- DBE_XML.XML_DOM_ITEM

  Returns the element corresponding to the index in a list or map based on the index.

  The prototype of the DBE_XML.XML_DOM_ITEM function is as follows:

```
DBE_XML.XML_DOM_ITEM (
   id IN RAW(13),
   index IN INTEGER
)
RETURNS RAW(13);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-121 DBE_XML.XML_DOM_ITEM API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.
index	INTEGER	IN	No	Index of the element to be retrieved.

- DBE_XML.XML_DOM_MAKE_ELEMENT

  Returns the DOMElement object after conversion.

  The prototype of the DBE_XML.XML_DOM_MAKE_ELEMENT stored procedure is as follows:

```
DBE_XML.XML_DOM_MAKE_ELEMENT(
id IN RAW(13))
RETURNS RAW(13)
```

Table 2-122 DBE_XML.XML_DOM_MAKE_ELEMENT API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

- DBE_XML.XML_DOM_MAKENODE

  Converts the given object to the DOMNode type.

  The prototype of the DBE_XML.XML_DOM_MAKENODE stored procedure is as follows:

```
DBE_XML.XML_DOM_MAKENODE(
  id    RAW(13)
)
RETURNS DOMNODE;
```

Table 2-123 DBE_XML.XML_DOM_MAKENODE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	XML DOM object.

- DBE_XML.XML_DOM_NEW_DOM_DOCUMENT_EMPTY

  Returns a new DOMDocument object.

  The prototype of the DBE_XML.XML_DOM_NEW_DOM_DOCUMENT_EMPTY function is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_XML.XML_DOM_NEW_DOM_DOCUMENT_EMPTY()
RETURNS RAW(13);
```

- DBE_XML.XML_DOM_NEW_DOM_DOCUMENT_CLOB

  Returns a new DOMDocument instance object created from the specified CLOB type.

  The prototype of the DBE_XML.XML_DOM_NEW_DOM_DOCUMENT_CLOB function is:

```
DBE_XML.XML_DOM_NEW_DOM_DOCUMENT_CLOB(
   content IN  CLOB
)
RETURNS RAW(13);
```

  Table 2-124 xml_dom_new_dom_document_clob API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
content	CLOB	IN	No	Specified CLOB type.

- DBE_XML.XML_DOM_NEW_DOCUMENT_XMLTYPE

  Returns a new DOMDocument instance object created from the specified XMLType type.

  The prototype of the DBE_XML.XML_DOM_NEW_DOCUMENT_XMLTYPE function is as follows:

```
DBE_XML.XML_DOM_NEW_DOCUMENT_XMLTYPE(
   content IN  CLOB
)
RETURNS RAW(13);
```

  Table 2-125 DBE_XML.XML_DOM_NEW_DOCUMENT_XMLTYPE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
content	CLOB	IN	No	Specified CLOB type.

- DBE_XML.XML_DOM_SET_ATTRIBUTE

  Sets the attributes of a specified XML DOM object.

  The prototype of the DBE_XML.XML_DOM_SET_ATTRIBUTE stored procedure is as follows:

```
DBE_XML.XML_DOM_SET_ATTRIBUTE(
   docid   IN  RAW(13),
   name    IN  VARCHAR2,
   value   IN  VARCHAR2
)
RETURNS void;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-126 DBE_XML.XML_DOM_SET_ATTRIBUTE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
docid	RAW(13)	IN	No	XML DOM object.
name	VARCHAR2	IN	No	A string.
value	VARCHAR2	IN	No	A string.

- DBE_XML.XML_DOM_SET_CHARSET

  Sets the character set for a DOMDocument object.

  The prototype of the DBE_XML.XML_DOM_SET_CHARSET function is as follows:

```
DBE_XML.XML_DOM_SET_CHARSET(
   id      IN RAW(13),
   charset IN VARCHAR2
)
RETURNS void;
```

Table 2-127 DBE_XML.XML_DOM_SET_CHARSET API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.
charset	VARCHAR2	IN	No	Character set.

- DBE_XML.XML_DOM_SET_DOCTYPE

  Sets the external DTD of a DOMDocument object.

  The prototype of the DBE_XML.XML_DOM_SET_DOCTYPE function is as follows:

```
DBE_XML.XML_DOM_SET_DOCTYPE(
   id          IN  RAW(13),
   dtd_name    IN  VARCHAR2,
   system_id   IN  VARCHAR2,
   public_id   IN  VARCHAR2
)
RETURNS void;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-128 DBE_XML.XML_DOM_SET_DOCTYPE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.
dtd_name	VARCHAR2	IN	No	Name of the DOCType to be initialized.
system_id	VARCHAR2	IN	No	ID of the system whose DOCType needs to be initialized.
public_id	VARCHAR2	IN	No	Public ID of the DOCType to be initialized.

- DBE_XML.XML_DOM_SET_NODE_VALUE

  Sets the value of a node in the DOMNode object.

  The prototype of the DBE_XML.XML_DOM_SET_NODE_VALUE stored procedure is as follows:

  ```
  DBE_XML.XML_DOM_SET_NODE_VALUE(
  id IN RAW(13),
  node_value IN VARCHAR2)
  RETURNS VOID
  ```

  Table 2-129 DBE_XML.XML_DOM_SET_NODE_VALUE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.
node_value	VARCHAR2	IN	No	String to be set in the DOMNode object.

- DBE_XML.XML_DOM_WRITE_TO_BUFFER_DOC

  Writes the given DOMDocument object to the buffer.

  The prototype of the DBE_XML.XML_DOM_WRITE_TO_BUFFER_DOC stored procedure is as follows:

  ```
  DBE_XML.XML_DOM_WRITE_TO_BUFFER_DOC(
  id IN RAW(13))
  RETURNS VARCHAR2;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-130 DBE_XML.XML_DOM_WRITE_TO_BUFFER_DOC API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	XML DOM object.

- DBE_XML.XML_DOM_WRITE_TO_BUFFER_NODE

  Writes the given DOMNode object to the buffer.

  The prototype of the DBE_XML.XML_DOM_WRITE_TO_BUFFER_NODE stored procedure is as follows:

  ```
  DBE_XML.XML_DOM_WRITE_TO_BUFFER_NODE(
  id IN RAW(13))
  RETURNS VARCHAR2;
  ```

  Table 2-131 DBE_XML.XML_DOM_WRITE_TO_BUFFER_NODE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	XML DOM object.

- DBE_XML.XML_DOM_WRITE_TO_CLOB_DOC

  Writes the given DOMDocument object to a CLOB.

  The prototype of the DBE_XML.XML_DOM_WRITE_TO_CLOB_DOC stored procedure is as follows:

  ```
  DBE_XML.XML_DOM_WRITE_TO_CLOB_DOC(
      id IN RAW(13)
  )
  RETURNS VARCHAR2;
  ```

  Table 2-132 DBE_XML.XML_DOM_WRITE_TO_CLOB_DOC API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	XML DOM object.

- DBE_XML.XML_DOM_WRITE_TO_CLOB_NODE

  Writes the given DOMNode object to a CLOB.

  The prototype of the DBE_XML.XML_DOM_WRITE_TO_CLOB_NODE stored procedure is as follows:

  ```
  DBE_XML.XML_DOM_WRITE_TO_CLOB_NODE(
      id          IN  RAW(13)
  )
  RETURNS CLOB;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-133 DBE_XML.XML_DOM_WRITE_TO_CLOB_NODE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	XML DOM object.

- DBE_XML.XML_DOM_WRITE_TO_FILE_DOC

  Writes an XML node to a specified file using the database character set.

  The prototype of the DBE_XML.XML_DOM_WRITE_TO_FILE_DOC stored procedure is as follows:

```
DBE_XML.XML_DOM_WRITE_TO_FILE_DOC(
id IN  RAW(13),
file_dir IN  VARCHAR2)
RETURNS VOID

DBE_XML.XML_DOM_WRITE_TO_FILE_DOC(
id IN  RAW(13),
file_dir IN  VARCHAR2,
charset  IN  VARCHAR2)
RETURNS VOID PACKAGE
```

Table 2-134 DBE_XML.XML_DOM_WRITE_TO_FILE_DOC API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	Yes	XML DOM object.
file_dir	VARCHAR2	IN	No	File to be written.
charset	VARCHAR2	IN	No	Specified character set.

- DBE_XML.XML_DOM_WRITE_TO_FILE_NODE

  Writes an XML node to a specified file using the database character set.

  The prototype of the DBE_XML.XML_DOM_WRITE_TO_FILE_NODE stored procedure is as follows:

```
DBE_XML.XML_DOM_WRITE_TO_FILE_NODE(
id IN  RAW(13),
filename IN  VARCHAR2)
RETURNS VOID
```

Table 2-135 DBE_XML.XML_DOM_WRITE_TO_FILE_NODE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
filename	VARCHAR2	IN	No	Specified file address.

- DBE_XML.XML_DOM_GET_SESSION_TREE_NUM

  Queries the number of DOM trees of all types in the current session.

  The prototype of the DBE_XML.XML_DOM_GET_SESSION_TREE_NUM function is as follows:

  ```
  DBE_XML.XML_DOM_GET_SESSION_TREE_NUM()
  RETURNS INTEGER
  ```

- DBE_XML.XML_DOM_GET_DOC_TREES_INFO

  Queries the DOM tree information of the document type in the current session, such as the memory usage.

  The prototype of the DBE_XML.XML_DOM_GET_DOC_TREES_INFO function is as follows:

  ```
  DBE_XML.XML_DOM_GET_DOC_TREES_INFO()
  RETURNS VARCHAR2
  ```

- DBE_XML.XML_DOM_GET_DETAIL_DOC_TREE_INFO

  Queries the number of subnodes of each type in the specified document.

  The prototype of the DBE_XML.XML_DOM_GET_DETAIL_DOC_TREE_INFO function is as follows:

  ```
  dbe_xml.xml_dom_get_detail_doc_tree_info(
  id IN  RAW(13))
  RETURNS VARCHAR2
  ```

  Table 2-136 DBE_XML.XML_DOM_GET_DETAIL_DOC_TREE_INFO API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	RAW(13)	IN	No	XML DOM object.

2.12.2 Secondary Encapsulation APIs

2.12.2.1 DBE_APPLICATION_INFO

APIs

Table 2-137 provides all APIs supported by the DBE_APPLICATION_INFO package. DBE_APPLICATION_INFO applies to the current session.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-137 DBE_APPLICATION_INFO

API	Description
DBE_APPLICATION_INFO.SET_CLIENT_INFO	Writes client information.
DBE_APPLICATION_INFO.READ_CLIENT_INFO	Reads client information.
DBE_APPLICATION_INFO.SET_MODULE	Sets the name of the currently running module to the new module. You can set the module and action.
DBE_APPLICATION_INFO.READ_MODULE	Reads the values of the module and action columns of the current session.
DBE_APPLICATION_INFO.SET_ACTION	Sets the name of the current action in the current module.

- DBE_APPLICATION_INFO.SET_CLIENT_INFO

  Writes client information.

  The prototype of DBE_APPLICATION_INFO.SET_CLIENT_INFO is as follows:

  ```
  DBE_APPLICATION_INFO.SET_CLIENT_INFO(
      str text
  )returns void;
  ```

  Table 2-138 DBE_APPLICATION_INFO.SET_CLIENT_INFO API parameter

Parameter	Description
str	Client information to be written. The maximum length is 64 bytes. If the length exceeds 64 bytes, it will be truncated.

- DBE_APPLICATION_INFO.READ_CLIENT_INFO

  Reads client information.

  The prototype of DBE_APPLICATION_INFO.READ_CLIENT_INFO is as follows:

  ```
  DBE_APPLICATION_INFO.READ_CLIENT_INFO(
   OUT client_info text);
  ```

  Table 2-139 DBE_APPLICATION_INFO.READ_CLIENT_INFO API parameter

Parameter	Description
client_info	Client information

- DBE_APPLICATION_INFO.SET_MODULE

  Sets the name of the currently running module to the new module.

  The prototype of DBE_APPLICATION_INFO.SET_MODULE is as follows:

  ```
  DBE_APPLICATION_INFO.SET_MODULE(
      IN module_name text,
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    IN action_name text
);
```

Table 2-140 DBE_APPLICATION_INFO.SET_MODULE API parameters

Parameter	Description
module_name	Name of the running module. The maximum length is 64 bytes. If the length exceeds 64 bytes, it will be truncated.
action_name	Name of the current action in the current module. The maximum length is 64 bytes. If the length exceeds 64 bytes, it will be truncated.

Example:

```
CALL dbe_application_info.set_module('module_name','action_name');
 set_module
------------


(1 row)
```

- DBE_APPLICATION_INFO.READ_MODULE

  Reads the values of the module and action columns of the current session.

  The prototype of DBE_APPLICATION_INFO.READ_MODULE is as follows:

```
DBE_APPLICATION_INFO.READ_MODULE(
    OUT module_name text,
    OUT action_name text
);
```

Table 2-141 DBE_APPLICATION_INFO.READ_MODULE API parameters

Parameter	Description
module_name	Name of the running module.
action_name	Name of the current action in the current module.

Example:

```
DECLARE
 module varchar2(64);
 action varchar2(64);
BEGIN
 dbe_application_info.read_module(module,action);
 dbe_output.print_line(module);
 dbe_output.print_line(action);
END;
/
module_name
action_name
ANONYMOUS BLOCK EXECUTE
```

- DBE_APPLICATION_INFO.SET_ACTION

  Sets the name of the current action in the current module.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The prototype of DBE_APPLICATION_INFO.SET_ACTION is as follows:

```
DBE_APPLICATION_INFO.SET_ACTION(
      IN action_name text
);
```

Table 2-142 DBE_APPLICATION_INFO.SET_ACTION API parameters

Parameter	Description
action_name	Name of the current action in the current module. The maximum length is 64 bytes. If the length exceeds 64 bytes, it will be truncated.

Example:

```
CALL dbe_application_info.set_action('action_name');
 set_action
------------


(1 row)
```

2.12.2.2 DBE_COMPRESSION

APIs

Evaluate the sampling compression speed of a specified data object or obtain the compression type of a specified row of data based on the input parameters. **Table 2-143** lists APIs supported by DBE_COMPRESSION.

Table 2-143 DBE_COMPRESSION

API	Description
DBE_COMPRESSION.GET_COMPRESSION_RATIO	Evaluates the sampling compression ratio of a specified data object based on input parameters.
DBE_COMPRESSION.GET_COMPRESSION_TYPE	Obtains the compression type of data in a specified row based on input parameters.
DBE_COMPRESSION.GET_ACTUAL_COMPRESSION_RATIO	Evaluates the actual compression ratio of a specified data object based on input parameters.
DBE_COMPRESSION.GET_HOLE_RATIO	Evaluates the sampling hole rate of a specified cross-page transparent compression data object based on input parameters.
DBE_COMPRESSION.TURBO_COMPRESS_BUFFER_STATS	Outputs the usage of turbo_compress_shared_buffers.
DBE_COMPRESSION.GET_INDEX_COMPRESSION_TYPE	Obtains the compression type of data in a specified index page based on input parameters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

The GET_COMPRESSION_TYPE API supports only DNs in the distributed database.

- DBE_COMPRESSION.GET_COMPRESSION_RATIO

  Evaluates the sampling compression ratio of a specified data object based on input parameters.

  The prototype of DBE_COMPRESSION.GET_COMPRESSION_RATIO is as follows:

```
DBE_COMPRESSION.GET_COMPRESSION_RATIO(
SCRATCHTBSNAME IN VARCHAR2,
OWNNAME        IN VARCHAR2,
OBJNAME        IN VARCHAR2,
SUBOBJNAME     IN VARCHAR2,
COMPTYPE       IN NUMBER,
BLKCNT_CMP     OUT INTEGER,
BLKCNT_UNCMP   OUT INTEGER,
ROW_CMP        OUT INTEGER,
ROW_UNCMP      OUT INTEGER,
CMP_RATIO      OUT NUMBER,
COMPTYPE_STR   OUT VARCHAR2,
SAMPLE_RATIO   IN NUMBER DEFAULT 20,
OBJTYPE        IN INTEGER DEFAULT 1);
```

  NOTE

  - For a cross-page transparently compressed table, the partition name must be passed if it is a partitioned table and the subpartition name must be passed if it is a subpartitioned table; otherwise, the error message "no data found." will be displayed. For an advanced compressed table, the partition name must be passed if it is a partitioned table (or a subpartitioned table); otherwise, the error message "no data found." will be displayed.
  - The compression ratio of cross-page transparent compression cannot be evaluated on an advanced compressed table. If you directly use this API on such a table, the error message "The ilm turbo policy is not supported here." will be displayed.

Table 2-144 DBE_COMPRESSION.GET_COMPRESSION_RATIO API parameters

Parameter	Description
SCRATCHTBSNAME	Tablespace to which a data object belongs.
OWNNAME	Data object owner (schema to which the data object belongs).
OBJNAME	Data object name.
SUBOBJNAME	Data subobject name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
COMPTYPE	Compression type. The options are as follows: ● **1**: uncompressed. ● **2**: advanced compression. ● **3**: cross-page transparent compression (low). ● **4**: cross-page transparent compression (medium). ● **5**: cross-page transparent compression (high). ● **6**: advanced compression (high).
BLKCNT_CMP	Number of blocks occupied by compressed samples.
BLKCNT_UNCMP	Number of blocks occupied by uncompressed samples.
ROW_CMP	Number of rows that can be contained in a single block after samples are compressed. The value is **0** for cross-page transparent compression.
ROW_UNCMP	Number of rows that can be contained in a single data block when samples are not compressed. The value is **0** for cross-page transparent compression.
CMP_RATIO	Compression ratio, that is, **blkcnt_uncmp** divided by **blkcnt_cmp**.
COMPTYPE_STR	Character string that describes the compression type.
SAMPLE_RATIO	Sampling ratio. The value is an integer or floating-point number ranging from 0 to 100, corresponding to the sampling ratio of *N* percent. The default value is **20**, indicating that 20% of the rows are sampled.
OBJTYPE	Object type. The options are as follows: ● **1**: table object.

- DBE_COMPRESSION.GET_COMPRESSION_TYPE

  This API is used to obtain the compression status (**1** for uncompressed and **2** for compressed) of a specified row based on input parameters. It is an O&M API and does not check the visibility. That is, if the input CTID is a deleted row, this API still returns the latest status of the current row on the page. Cross-page transparent compression does not support this advanced package. The default value **1** is returned, which is meaningless.

  The prototype of DBE_COMPRESSION.GET_COMPRESSION_TYPE is as follows:

```
DBE_COMPRESSION.GET_COMPRESSION_TYPE(
OWNNAME    IN VARCHAR2,
TABNAME    IN VARCHAR2,
CTID       IN TEXT,
SUBOBJNAME IN VARCHAR2 DEFAULT NULL,
BUCKETID   IN pg_catalog.int2 DEFAULT NULL))
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-145 DBE_COMPRESSION.GET_COMPRESSION_TYPE API parameters

Parameter	Description
OWNNAME	Data object owner (schema to which the data object belongs).
TABNAME	Data object name.
CTID	CTID of the target row.
SUBOBJNAME	Data subobject name.
BUCKETID	Table bucket ID of the target row.

- DBE_COMPRESSION.GET_ACTUAL_COMPRESSION_RATIO

  Evaluates the actual compression ratio of a specified data object based on input parameters. Only cross-page transparently compressed tables are supported.

  The prototype of DBE_COMPRESSION.GET_ACTUAL_COMPRESSION_RATIO is as follows:

```
DBE_COMPRESSION.GET_ACTUAL_COMPRESSION_RATIO(
OWNER_NAME       IN VARCHAR2,
OBJNAME          IN VARCHAR2,
FILEPATH         OUT VARCHAR2,
IS_COMPRESS      OUT BOOL,
FILE_COUNT       OUT INTEGER,
LOGIC_SIZE       OUT BIGINT,
PHYSIC_SIZE      OUT BIGINT,
COMPRESS_RATIO   OUT VARCHAR2);
```

Table 2-146 DBE_COMPRESSION.GET_ACTUAL_COMPRESSION_RATIO API parameters

Parameter	Description
OWNER_NAME	Data object owner (schema to which the data object belongs).
OBJNAME	Data object name.
FILEPATH	Data object path.
IS_COMPRESS	Compression type.
FILE_COUNT	Number of files, including data files and CMAP files.
LOGIC_SIZE	Logical size.
PHYSIC_SIZE	Physical size.
COMPRESS_RATIO	Compression ratio. In small table scenarios, the compression ratio may be less than 1, which is normal.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

In small table scenarios, the compression ratio may be less than 1, which is normal.

For example, create a **table_test** table, insert a data record, and use an advanced package for query. The output shows that the compression ratio is less than 1.

```
gaussdb=# CREATE TABLE table_test(a int) ILM ADD POLICY ROW STORE COMPRESS TURBO
HIGH UNIT AFTER 1 days OF NO MODIFICATION;
gaussdb=# INSERT INTO table_test VALUES(1);
gaussdb=# CALL DBE_COMPRESSION.GET_ACTUAL_COMPRESSION_RATIO('public', 'table_test',
NULL,NULL,NULL,NULL,NULL,NULL);
```

- DBE_COMPRESSION.GET_HOLE_RATIO

  Evaluates the sampling hole rate of a specified cross-page transparent compression data object based on input parameters. Only cross-page transparent compression tables are supported.

  The prototype of DBE_COMPRESSION.GET_HOLE_RATIO is as follows:

```
DBE_COMPRESSION.GET_HOLE_RATIO(
OWNER_NAME       IN VARCHAR2,
OBJNAME          IN VARCHAR2,
FILE_COUNT       OUT INTEGER,
PHYSIC_SIZE      OUT BIGINT,
HOLE_SIZE        OUT BIGINT,
HOLE_RATIO       OUT VARCHAR2,
SAMPLE_RATIO     IN NUMBER DEFAULT 20);
```

Table 2-147 DBE_COMPRESSION.GET_HOLE_RATIO API parameters

Parameter	Description
OWNER_NAME	Data object owner (schema to which the data object belongs).
OBJNAME	Data object name.
FILE_COUNT	Number of data object files, including only data files.
PHYSIC_SIZE	Physical size.
HOLE_SIZE	Hole size.
HOLE_RATIO	Hole rate.
SAMPLE_RATIO	Sampling rate.

- DBE_COMPRESSION.TURBO_COMPRESS_BUFFER_STATS

  Outputs the usage of turbo_compress_shared_buffers.

  The prototype of DBE_COMPRESSION.TURBO_COMPRESS_BUFFER_STATS is as follows:

```
DBE_COMPRESSION.TURBO_COMPRESS_BUFFER_STATS(
CAPACITY        OUT BIGINT,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
MAIN_LEN        OUT BIGINT,
FREE_LEN        OUT BIGINT,
BUFFER_ACCESS   OUT BIGINT,
BUFFER_NOREADS  OUT BIGINT,
DISK_READS      OUT BIGINT,
RECYCLE_TIMES   OUT BIGINT,
RECYCLE_STEP    OUT BIGINT,
RECYCLE_WAITS   OUT BIGINT,
HIT_RATIO       OUT VARCHAR2);
```

Table 2-148 DBE_COMPRESSION.TURBO_COMPRESS_BUFFER_STATS API parameters

Parameter	Description
CAPACITY	Capacity of turbo_compress_shared_buffers.
MAIN_LEN	Length of main_list in the LRU list.
FREE_LEN	Length of free_list in the LRU list.
BUFFER_ACCESS	Number of logical reads.
BUFFER_NOREADS	Number of new blocks.
DISK_READS	Number of physical reads.
RECYCLE_TIMES	Number of recycling times.
RECYCLE_STEP	Recycling step size.
RECYCLE_WAITS	Number of wait times during the execution of **alloc buf ctrl**.
HIT_RATIO	Hit ratio in the memory.

- DBE_COMPRESSION.GET_INDEX_COMPRESSION_TYPE

  Checks whether the index page has been compressed. This advanced package can only query the status of advanced compression, but cannot query the status of cross-page transparent compression.

  The prototype of DBE_COMPRESSION.GET_INDEX_COMPRESSION_TYPE is as follows:

  ```
  DBE_COMPRESSION.GET_INDEX_COMPRESSION_TYPE(
  IDXOID          IN OID,
  IDXTYPE         IN TEXT,
  BUCKET_ID       IN INT2 DEFAULT NULL,
  BLOCK_ID        IN INT8);
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-149 DBE_COMPRESSION.GET_INDEX_COMPRESSION_TYPE API parameters

Parameter	Description
IDXOID	Index OID.
IDXTYPE	Index type. Ordinary index ('i'), global index ('I'), and partitioned index ('x') are supported.
BUCKET_ID	If the index is stored in a bucket, the corresponding bucket ID is passed. The default value is **NULL**.
BLOCK_ID	Block ID of the index to be queried. If the value is a decimal, the value is rounded off.

2.12.2.3 DBE_DESCRIBE

Data Types

The advanced package DBE_DESCRIBE has two built-in data types. The two data types are created using user-defined types and are used for the return values of the DESCRIBE_PROCEDURE API.

- DBE_DESCRIBE.NUMBER_TABLE

  This type is the TABLE type of NUMBER and is implemented through the TABLE OF syntax.

  The prototype of DBE_DESCRIBE.NUMBER_TABLE is as follows:

  ```
  CREATE TYPE DBE_DESCRIBE.NUMBER_TABLE AS TABLE OF NUMBER INDEX BY INTEGER;
  ```

- DBE_DESCRIBE.VARCHAR2_TABLE

  This type is the TABLE type of VARCHAR2 and is implemented through the TABLE OF syntax.

  The prototype of DBE_DESCRIBE.VARCHAR2_TABLE is as follows:

  ```
  CREATE TYPE DBE_DESCRIBE.VARCHAR2_TABLE AS TABLE OF VARCHAR2(30) INDEX BY INTEGER;
  ```

API

The advanced package DBE_DESCRIBE provides the DBE_DESCRIBE.DESCRIBE_PROCEDURE API. For details about the API description, see **Table 2-150**.

Table 2-150 DBE_DESCRIBE

API	Description
DBE_DESCRIBE.DESCRIBE_PROCEDURE	Displays the parameters of a stored procedure or function.

- DBE_DESCRIBE.DESCRIBE_PROCEDURE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The stored procedure DESCRIBE_PROCEDURE is used to display the parameter information of a stored procedure or function, such as the parameter name, parameter mode, and parameter position. The parameter information of a function or stored procedure is returned in a list.

The prototype of DBE_DESCRIBE.DESCRIBE_PROCEDURE is as follows:

```
DBE_DESCRIBE.DESCRIBE_PROCEDURE(
  object_name               IN  VARCHAR2,
  reserved1                 IN  VARCHAR2,
  reserved2                 IN  VARCHAR2,
  overload                  OUT DBE_DESCRIBE.NUMBER_TABLE,
  dataposition               OUT DBE_DESCRIBE.NUMBER_TABLE,
  datalevel                 OUT DBE_DESCRIBE.NUMBER_TABLE,
  argument_name                OUT DBE_DESCRIBE.VARCHAR2_TABLE,
  datatype                  OUT DBE_DESCRIBE.NUMBER_TABLE,
  default_value              OUT DBE_DESCRIBE.NUMBER_TABLE,
  in_out                    OUT DBE_DESCRIBE.NUMBER_TABLE,
  datalength                 OUT DBE_DESCRIBE.NUMBER_TABLE,
  dataprecision              OUT DBE_DESCRIBE.NUMBER_TABLE,
  scale                     OUT DBE_DESCRIBE.NUMBER_TABLE,
  radix                     OUT DBE_DESCRIBE.NUMBER_TABLE,
  spare                     OUT DBE_DESCRIBE.NUMBER_TABLE,
  include_string_constraints OUT BOOLEAN
);
```

Table 2-151 DBE_DESCRIBE.DESCRIBE_PROCEDURE API parameters

Parameter	Type	Can Be Null	Description
object_name	varchar2	No	Stored procedure name. The syntax format of this parameter is [[*schema*.]*package*.]*function*[@*dblink*], in which: • *schema* (optional): schema name. • *package* (optional): package name. • *function* (required): function or stored procedure name. • *dblink* (optional): remote connection name.
reserved1	varchar2	Yes	Reserved parameter.
reserved2	varchar2	Yes	Reserved parameter.
overload	number_table	Yes	Unique ID assigned to a stored procedure or function. If a stored procedure or function is overloaded, each overloading of the stored procedure or function is counted and numbered from 1 in ascending order. If a stored procedure or function is not overloaded, **overload** is set to **0**.
dataposition	number_table	Yes	Position of a specified parameter in the parameter list.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Can Be Null	Description
datalevel	number_table	Yes	It is set to **0**.
argument_name	varchar2_table	Yes	Name of the parameter associated with the specified stored procedure.
datatype	number_table	Yes	Data type OID of the specified parameter.
default_value	number_table	Yes	If the specified parameter has a default value, the value is **1**. Otherwise, the value is **0**.
in_out	number_table	Yes	Parameter mode. The options are as follows: • **0**: IN. • **1**: OUT. • **2**: IN OUT.
datalength	number_table	Yes	Not supported. It is set to **0** by default.
dataprecision	number_table	Yes	Not supported. It is set to **0** by default.
scale	number_table	Yes	Not supported. It is set to **0** by default.
radix	number_table	Yes	If the value is of the numeric type (such as NUMBER and INTEGER), **10** is returned. Otherwise, **0** is returned. For details about the numeric type, see **Numeric Types**.
spare	number_table	Yes	Reserved parameter. It is set to **0** by default.
include_string_constraints	Boolean	Yes	Reserved parameter. It is set to **FALSE** by default.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The data type of the **datatype** parameter is different from that of the ORA-compatible database. GaussDB returns the OID of the data type, and the ORA-compatible database returns the number of the data type in the ORA-compatible database.
- **include_string_constraints** is a reserved parameter. Its value does not change and does not affect the return values of other parameters.
- The OIDs of the data types created by the CREATE TYPE operation are uncertain. Therefore, do not use these OIDs for sure.
- For the **dataposition** parameter, if a stored procedure is specified, the return value starts from **1**. If a function is specified, the return value starts from **0**. **0** indicates the position sequence number of the return value of the function.
- For the **argument_name** parameter, if a function is specified, the first position of the return value is empty. This position indicates the name of the return value of the described function (that is, the empty name).
- Do not directly specify a package. Otherwise, an error is reported.
- If you do not have the execute permission on a stored procedure, function, or package, the system considers that the stored procedure, function, or package does not exist and reports an error.
- The input parameters **reserved1** and **reserved2** are not involved in internal processing. Entering any character string does not affect the returned result.
- The advanced package cannot specify the stored procedures or functions obtained through database links.
- You are advised to add the schema prefix before the specified stored procedure or function. If the schema prefix is omitted, the advanced package searches for its owning entity in the schema of the current session. (If the owning entity does not exist in the schema of the current session, the mechanism specified by **bind_procedure_searchpath** in **behavior_compat_options** is followed.)
- If the %type operation is used to obtain the data type from a table column, the constraint on the data type (such as NUMBER(3) and VARCHAR2(10)) is not retained.

Example:

```
-- Create a stored procedure to encapsulate the advanced package for printing return values.
gaussdb=# CREATE PROCEDURE PRINT_DESCRIBE (obj_name IN VARCHAR2)
AS
  a_overload      DBE_DESCRIBE.NUMBER_TABLE;
  a_position      DBE_DESCRIBE.NUMBER_TABLE;
  a_level         DBE_DESCRIBE.NUMBER_TABLE;
  a_arg_name      DBE_DESCRIBE.VARCHAR2_TABLE;
  a_dty           DBE_DESCRIBE.NUMBER_TABLE;
  a_def_val       DBE_DESCRIBE.NUMBER_TABLE;
  a_mode          DBE_DESCRIBE.NUMBER_TABLE;
  a_length        DBE_DESCRIBE.NUMBER_TABLE;
  a_precision     DBE_DESCRIBE.NUMBER_TABLE;
  a_scale         DBE_DESCRIBE.NUMBER_TABLE;
  a_radix         DBE_DESCRIBE.NUMBER_TABLE;
  a_spare         DBE_DESCRIBE.NUMBER_TABLE;
  a_include_string_constraints BOOLEAN;
BEGIN
  DBE_DESCRIBE.DESCRIBE_PROCEDURE(
    obj_name,
    null,
    null,
    a_overload,
    a_position,
    a_level,
    a_arg_name,
    a_dty,
    a_def_val,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  a_mode,
  a_length,
  a_precision,
  a_scale,
  a_radix,
  a_spare,
  a_include_string_constraints
);
dbe_output.print('overload                ' || chr(9));
for indx in 1 .. a_overload.count
loop
  dbe_output.print(a_overload(indx) || chr(9));
end loop;

dbe_output.print_line(' ');

dbe_output.print('dataposition            ' || chr(9));
for indx in 1 .. a_position.count
loop
  dbe_output.print(a_position(indx) || chr(9));
end loop;

dbe_output.print_line(' ');

dbe_output.print('datalevel              ' || chr(9));
for indx in 1 .. a_level.count
loop
  dbe_output.print(a_level(indx) || chr(9));
end loop;

dbe_output.print_line(' ');

dbe_output.print('argument_name          ' || chr(9));
for indx in 1 .. a_arg_name.count
loop
  dbe_output.print(a_arg_name(indx) || chr(9));
end loop;

dbe_output.print_line(' ');

dbe_output.print('default_value          ' || chr(9));
for indx in 1 .. a_def_val.count
loop
  dbe_output.print(a_def_val(indx) || chr(9));
end loop;

dbe_output.print_line(' ');

dbe_output.print('in_out                 ' || chr(9));
for indx in 1 .. a_mode.count
loop
  dbe_output.print(a_mode(indx) || chr(9));
end loop;

dbe_output.print_line(' ');

dbe_output.print('length                 ' || chr(9));
for indx in 1 .. a_length.count
loop
  dbe_output.print(a_length(indx) || chr(9));
end loop;

dbe_output.print_line(' ');

dbe_output.print('precision              ' || chr(9));
for indx in 1 .. a_precision.count
loop
  dbe_output.print(a_precision(indx) || chr(9));
end loop;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  dbe_output.print_line(' ');

  dbe_output.print('scale                    ' || chr(9));
  for indx in 1 .. a_scale.count
  loop
    dbe_output.print(a_scale(indx) || chr(9));
  end loop;

  dbe_output.print_line(' ');

  dbe_output.print('radix                    ' || chr(9));
  for indx in 1 .. a_radix.count
  loop
    dbe_output.print(a_radix(indx) || chr(9));
  end loop;

  dbe_output.print_line(' ');
END;
/
CREATE PROCEDURE

-- Create a function overloaded three times.
gaussdb=# CREATE OR REPLACE FUNCTION TEST_FUNC_OVERLOAD (
  param_a IN NUMBER,
  param_b IN VARCHAR2,
  param_c OUT TEXT
)
RETURN VARCHAR2 package
AS
BEGIN
  dbe_output.print_line('This procedure/function test num param.');
  RETURN 'This procedure/function test num param.';
END;
/
CREATE FUNCTION

gaussdb=# CREATE OR REPLACE FUNCTION TEST_FUNC_OVERLOAD (
  param_a IN NUMBER DEFAULT 20,
  param_b VARCHAR2 DEFAULT 'n',
  param_c IN TEXT, param_d OUT DATE,
  param_e INOUT RAW
)
RETURN VARCHAR2 package
AS
BEGIN
  dbe_output.print_line('This procedure/function test num param.');
  RETURN 'This procedure/function test num param.';
END;
/
CREATE FUNCTION

gaussdb=# CREATE OR REPLACE FUNCTION TEST_FUNC_OVERLOAD (
  param_a IN NUMBER DEFAULT 20,
  param_b VARCHAR2 DEFAULT 'n',
  param_c IN TEXT, param_d IN DATE,
  param_e OUT RAW, param_f INOUT INTEGER
)
RETURN VARCHAR2 package
AS
BEGIN
  dbe_output.print_line('This procedure/function test num param.');
  RETURN 'This procedure/function test num param.';
END;
/
CREATE FUNCTION

-- Call the preceding encapsulation.
gaussdb=# BEGIN PRINT_DESCRIBE('TEST_FUNC_OVERLOAD'); END;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
/

-- Execution result.
overload                1   1   1   1   2   2   2   2   2   2   3   3   3   3   3   3   3
dataposition             0   1   2   3   0   1   2   3   4   5   0   1   2   3   4   5   6
datalevel               0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
argument_name                param_a    param_b    param_c       param_a    param_b    param_c
param_d    param_e       param_a    param_b    param_c    param_d    param_e    param_f
default_value            0   0   0   0   0   1   1   0   0   0   0   1   1   0   0   0   0
in_out                  1   0   0   1   1   0   0   0   1   2   1   0   0   0   0   1   2
length                  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
precision               0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
scale                  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
radix                  0   10   0   0   0   10   0   0   0   0   0   10   0   0   0   0   10
ANONYMOUS BLOCK EXECUTE

-- Clean up.
gaussdb=# DROP FUNCTION TEST_FUNC_OVERLOAD (
  param_a IN NUMBER,
  param_b IN VARCHAR2,
  param_c OUT TEXT
);
DROP FUNCTION

gaussdb=# DROP FUNCTION TEST_FUNC_OVERLOAD (param_a IN NUMBER,
  param_b VARCHAR2,
  param_c IN TEXT,
  param_d OUT DATE,
  param_e INOUT RAW
);
DROP FUNCTION

gaussdb=# DROP FUNCTION TEST_FUNC_OVERLOAD (param_a IN NUMBER,
  param_b VARCHAR2,
  param_c IN TEXT,
  param_d IN DATE,
  param_e OUT RAW,
  param_f INOUT INTEGER
);
DROP FUNCTION

gaussdb=# DROP PROCEDURE PRINT_DESCRIBE (obj_name IN VARCHAR2);
DROP PROCEDURE
```

2.12.2.4 DBE_FILE

DBE_FILE provides the capabilities of reading and writing OS text files for stored procedures.

Precautions

- DBE_FILE requires that files opened using DBE_FILE.FOPEN be encoded using the database character set. If the opened files are not encoded using the expected character set, an encoding verification error occurs when DBE_FILE.READ_LINE is used to read files. DBE_FILE requires that files opened using DBE_FILE.FOPEN_NCHAR be encoded using the UTF-8 character set. If the opened files are not encoded using the expected character set, an encoding verification error occurs when DBE_FILE.READ_LINE_NCHAR is used to read files.
- When DBE_OUTPUT.PUT_LINE is used to print the DBE_FILE.READ_LINE_NCHAR API result, ensure that the UTF-8 character set encoding can be converted to the current database character set encoding. If the preceding conditions are met, the result can be properly output. DBE_OUTPUT.PRINT_LINE does not support this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_FILE requires that the character set encoding of the client be the same as that of the database.
- Assume that the database character set is ASCII, the client character set supports Chinese, and the client calls DBE_FILE.WRITE_NCHAR or DBE_FILE.WRITE_LINE_NCHAR to write Chinese content. If the entered content is in UTF-8 encoding format, the written content may not be encoded in UTF-8 format. An error may be reported when the DBE_FILE.READ_LINE_NCHAR is used.

Data Types

DBE_FILE.FILE_TYPE

Defines the representation of files in the DBE_FILE package. The columns in DBE_FILE.FILE_TYPE are private columns of the DBE_FILE package. Do not change the column values in DBE_FILE.FILE_TYPE.

The prototype of DBE_FILE.FILE_TYPE is as follows:

```
CREATE TYPE DBE_FILE.FILE_TYPE AS(
    id INTEGER,
    datatype INTEGER,
    byte_mode BOOLEAN
);
```

Table 2-152 DBE_FILE.FILE_TYPE columns

Parameter	Description
id	File handle.
datatype	File data type (CHAR, NCHAR, or binary). Currently, CHAR and NCHAR files are supported. For a CHAR file, **1** is returned. For an NCHAR file, **2** is returned.
byte_mode	Specifies whether the file is opened in binary mode (**TRUE**) or text mode (**FALSE**).

APIs

Table 2-153 lists APIs supported by the DBE_FILE package.

Table 2-153 DBE_FILE

API	Description
DBE_FILE.OPEN/ DBE_FILE.FOPEN	Opens a file based on the specified directory and file name. The corresponding file handle or the DBE_FILE.FILE_TYPE object encapsulated with the handle is returned.
DBE_FILE.IS_CLOSE	Checks whether a specified file handle is closed.
DBE_FILE.IS_OPEN	Checks whether a specified file handle is opened.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_FILE.READ_LINE	Reads a line of a specified length from an open file handle.
DBE_FILE.WRITE	Writes data to the buffer of an open file.
DBE_FILE.NEW_LINE	Writes one or more line terminators to the buffer of an open file.
DBE_FILE.WRITE_LINE	Writes data to the buffer of an open file and automatically appends a line terminator.
DBE_FILE.FORMAT_WRITE	Writes data in a specified format to the buffer of an open file.
DBE_FILE.GET_RAW	Reads RAW data of a specified number of bytes from an open file.
DBE_FILE.PUT_RAW	Writes RAW data to the buffer of an open file.
DBE_FILE.FLUSH	Writes data in the buffer to a physical file.
DBE_FILE.CLOSE	Closes an open file handle.
DBE_FILE.CLOSE_ALL	Closes all file handles opened in a session.
DBE_FILE.REMOVE	Removes a disk file based on the specified directory and file name. You must have sufficient permissions to perform this operation.
DBE_FILE.RENAME	Renames a disk file. This function is similar to the **mv** command of Unix.
DBE_FILE.COPY	Copies data in a continuous area to a new file. If **start_line** and **end_line** are not specified, the entire file is copied.
DBE_FILE.GET_ATTR	Reads and returns the attributes of a disk file.
DBE_FILE.SEEK	Adjusts the position of a file pointer forward or backward based on the specified number of bytes.
DBE_FILE.GET_POS	Returns the current offset of a file, in bytes.
DBE_FILE.FOPEN_NCHAR	Opens a file of the NCHAR type based on the specified directory and file name.
DBE_FILE.WRITE_NCHAR	Writes data of the NVARCHAR2 type to the buffer of an open file of the NCHAR type.
DBE_FILE.WRITE_LINE_NCHAR	Writes data of the NVARCHAR2 type to the buffer of an open file of the NCHAR type and automatically appends a line terminator.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_FILE.FORMAT_WRITE_NCHAR	Writes data of the NVARCHAR2 type to the buffer of an open file of the NCHAR type in a specified format. It is a DBE_FILE.WRITE_NCHAR API that allows formatting.
DBE_FILE.READ_LINE_NCHAR	Reads a line of a specified length from an open file of the NCHAR type.

- DBE_FILE.OPEN/DBE_FILE.FOPEN

  Opens a file. You can specify the maximum line size. A maximum of 50 files can be opened in a session. This function returns a file handle of the INTEGER type. The function of DBE_FILE.FOPEN is similar to that of DBE_FILE.OPEN. The difference is that DBE_FILE.FOPEN returns a DBE_FILE.FILE_TYPE object.

  The prototypes of DBE_FILE.OPEN and DBE_FILE.FOPEN are as follows:

  ```
  DBE_FILE.OPEN (
  dir             IN    TEXT,
  file_name       IN    TEXT,
  open_mode       IN    TEXT,
  max_line_size   IN    INTEGER DEFAULT 1024)
  RETURN INTEGER;

  DBE_FILE.FOPEN(
     dir          IN TEXT,
     file_name    IN TEXT,
     open_mode    IN TEXT,
     max_line_size IN INTEGER DEFAULT 1024)
  RETURN DBE_FILE.FILE_TYPE;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-154 DBE_FILE.OPEN/DBE_FILE.FOPEN API parameters

Parameter	Type	Input/OutputParameter	CanBeNull	Description
dir	TEXT	IN	No	Directory of a file. The string indicates the name of a directory object. **NOTE** • Location of the file directory, which needs to be added to system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. • When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to read and write files in the file path specified by **safe_data_path**.
file_name	TEXT	IN	No	File name with an extension (file type), excluding the path name. A path contained in a file name is ignored in the OPEN function. In Unix, the file name cannot end with the combination of a slash and a dot (/.).
open_mode	TEXT	IN	No	Open mode of a specified file, including: • **r**: read text. • **w**: write text. • **a**: append text. • **rb**: read byte. • **wb**: write byte. • **ab**: append byte. **NOTE** For the write operation, the system checks the file type. If the ELF file is written, an error is reported and the system exits.
max_line_size	INTEGER	IN	Yes	Maximum number of bytes in each line, including newline characters. The minimum value is **1** and the maximum value is **32767**. If this parameter is not specified, the default value **1024** is used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_FILE.IS_CLOSE

  Checks whether a specified file handle is closed. A Boolean value is returned. If an invalid file handle is detected, the INVALID_FILEHANDLE exception will be thrown.

  The prototype of DBE_FILE.IS_CLOSE is as follows:

```
DBE_FILE.IS_CLOSE (
   file IN INTEGER)
RETURN BOOLEAN;

DBE_FILE.IS_CLOSE(
   file IN DBE_FILE.FILE_TYPE)
RETURN BOOLEAN;
```

  Table 2-155 DBE_FILE.IS_CLOSE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER or DBE_FILE.FILE_TYPE	IN	Yes	File handle or DBE_FILE.FILE_TYPE object to be checked. If it is left empty, the DBE_FILE.IS_CLOSE API returns null.

- DBE_FILE.IS_OPEN

  Checks whether a specified file handle is opened. A Boolean value is returned. If an invalid file handle is detected, the INVALID_FILEHANDLE exception will be thrown.

  The prototype of DBE_FILE.IS_OPEN is as follows:

```
DBE_FILE.IS_OPEN(
   file IN INTEGER)
RETURN BOOLEAN;

DBE_FILE.IS_OPEN(
   file IN DBE_FILE.FILE_TYPE)
RETURN BOOLEAN;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-156 DBE_FILE.IS_OPEN API parameters

Parameter	Type	Input/OutputParameter	Can Be Null	Description
file	INTEGER or DBE_FILE.FILE_TYPE	IN	Yes	File handle or DBE_FILE.FILE_TYPE object to be checked. If it is left empty, the DBE_FILE.IS_OPEN API returns **false**.

- DBE_FILE.READ_LINE

  Reads data from an open file and stores the read result to the buffer. It reads data to the end of each line excluding the line terminator, to the end of the file, or to the size specified by the **len** parameter. The length of the data to be read cannot exceed the value of **max_line_size** specified when the file is opened.

  The prototype of DBE_FILE.READ_LINE is as follows:

```
DBE_FILE.READ_LINE(
   file   IN  INTEGER,
   buffer OUT TEXT,
   len    IN  INTEGER DEFAULT NULL)
RETURN TEXT;

DBE_FILE.READ_LINE(
   file   IN  DBE_FILE.FILE_TYPE,
   buffer OUT TEXT,
   len    IN  INTEGER DEFAULT NULL)
RETURN TEXT;
```

Table 2-157 DBE_FILE.READ_LINE API parameters

Parameter	Type	Input/OutputParameter	Can Be Null	Description
file	INTEGER or DBE_FILE.FILE_TYPE	IN	No	File handle opened using OPEN or object of the DBE_FILE.FILE_TYPE type opened using FOPEN. The file must be opened in read mode. Otherwise, the INVALID_OPERATION exception is thrown.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
buffer	TEXT	OUT	No	Buffer for receiving data.
len	INTEGER	IN	Yes	Number of bytes read from a file. The default value is **NULL**. If the default value **NULL** is used, **max_linesize** is used to specify the line size.

- DBE_FILE.WRITE

  Writes buffer data to the buffer corresponding to a file. The file must be opened in write mode. This operation does not write a line terminator.

  The prototype of DBE_FILE.WRITE is as follows:

```
DBE_FILE.WRITE(
   file   IN INTEGER,
   buffer IN TEXT)
RETURN BOOLEAN;

DBE_FILE.WRITE(
   file   IN DBE_FILE.FILE_TYPE,
   buffer IN TEXT)
RETURN VOID;
```

Table 2-158 DBE_FILE.WRITE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER or DBE_FILE.FILE_TYPE	IN	No	File handle opened using OPEN or object of the DBE_FILE.FILE_TYPE type opened using FOPEN. The file must be opened in write mode. This operation does not write line terminators.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
buffer	TEXT	IN	Yes	Text data to be written to the file. The accumulated write length of each line cannot be greater than or equal to the value of **max_line_size** specified when OPEN or FOPEN is used. Otherwise, an error is reported when the file is flushed. If this parameter is left empty, the API directly returns an empty value. **NOTE** For the write operation, the system checks the file type. If the ELF file is written, an error is reported and the system exits.

- DBE_FILE.NEW_LINE

  Writes one or more line terminators to the buffer corresponding to a file. The line terminators are related to the platform used.

  The prototype of DBE_FILE.NEW_LINE is as follows:

```
DBE_FILE.NEW_LINE(
   file      IN INTEGER,
   line_nums IN INTEGER DEFAULT 1)
RETURN BOOLEAN;

DBE_FILE.NEW_LINE(
   file      IN DBE_FILE.FILE_TYPE,
   line_nums IN INTEGER DEFAULT 1)
RETURN VOID;
```

Table 2-159 DBE_FILE.NEW_LINE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER or DBE_FILE.FILE_TYPE	IN	No	File handle opened using OPEN or object of the DBE_FILE.FILE_TYPE type opened using FOPEN. The file must be opened in write mode. This operation does not write line terminators.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
line_nums	INTEGER	IN	Yes	Number of line terminators written to a file. The default value is **1**. If this parameter is left blank, line terminators are not written.

- DBE_FILE.WRITE_LINE

  Writes buffer data to the buffer corresponding to a file. The file must be opened in write mode. This operation automatically adds a line terminator.

  The prototype of DBE_FILE.WRITE_LINE is as follows:

```
DBE_FILE.WRITE_LINE(
   file   IN INTEGER,
   buffer IN TEXT,
   flush  IN BOOLEAN DEFAULT FALSE)
RETURN BOOLEAN;

DBE_FILE.WRITE_LINE(
   file   IN DBE_FILE.FILE_TYPE,
   buffer IN TEXT,
   flush  IN BOOLEAN DEFAULT FALSE)
RETURN VOID;
```

Table 2-160 DBE_FILE.WRITE_LINE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER or DBE_FILE.FILE_TYPE	IN	No	File handle opened using OPEN or object of the DBE_FILE.FILE_TYPE type opened using FOPEN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
buffer	TEXT	IN	Yes	Text data to be written to the file. The length of each line (including the newline character) cannot be greater than the value of **max_line_size** specified when OPEN or FOPEN is executed or the default value. Otherwise, an error is reported when the file is flushed. **NOTE** For the write operation, the system checks the file type. If the ELF file is written, an error is reported and the system exits.
flush	BOOLEAN	IN	Yes	Specifies whether to flush data to disks after WRITE_LINE. The default value is **FALSE**.

- DBE_FILE.FORMAT_WRITE

  The DBE_FILE.FORMAT_WRITE function writes formatted data to the buffer corresponding to an open file. It is a DBE_FILE.WRITE API that allows formatting.

  The prototype of DBE_FILE.FORMAT_WRITE is as follows:

```
DBE_FILE.FORMAT_WRITE(
   file   IN INTEGER,
   format IN TEXT,
   arg1   IN TEXT DEFAULT NULL,
   . . .
   arg6   IN TEXT DEFAULT NULL)
RETURN BOOLEAN;

DBE_FILE.FORMAT_WRITE(
   file   IN DBE_FILE.FILE_TYPE,
   format IN TEXT,
   arg1   IN TEXT DEFAULT NULL,
   . . .
   arg6   IN TEXT DEFAULT NULL)
RETURN VOID;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-161 DBE_FILE.FORMAT_WRITE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER or DBE_FILE.FILE_TYPE	IN	No	File handle opened using OPEN or object of the DBE_FILE.FILE_TYPE type opened using FOPEN.
format	TEXT	IN	Yes	Formatted string, containing the text and format characters \n and %s. If this parameter is left blank, no data is written.
[arg1. . .arg6]	TEXT	IN	Yes	Six optional parameters. The parameters and the positions of characters to be formatted are in one-to-one correspondence. If the parameter corresponding to a character to be formatted is not provided or the parameter is empty, an empty string is used to replace %s.

- DBE_FILE.GET_RAW

  Reads RAW data from an open file, stores the read result in the buffer, and returns the result from *r*.

  The prototype of DBE_FILE.GET_RAW is as follows:

```
DBE_FILE.GET_RAW(
   file   IN  INTEGER,
   r      OUT RAW,
   length IN  INTEGER DEFAULT NULL)
RETURN RAW;

DBE_FILE.GET_RAW(
   file   IN  DBE_FILE.FILE_TYPE,
   r      OUT RAW,
   length IN  INTEGER DEFAULT NULL)
RETURN RAW;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-162 DBE_FILE.GET_RAW API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER or DBE_FILE.FILE_TYPE	IN	No	File handle opened using OPEN or object of the DBE_FILE.FILE_TYPE type opened using FOPEN.
r	RAW	OUT	No	Buffer for receiving RAW data.
length	INTEGER	IN	Yes	Number of bytes read from the file. The default value is **NULL**. If the value is **NULL**, the maximum length of the RAW type is used to specify the size.

- DBE_FILE.PUT_RAW

  Writes raw data to the buffer corresponding to a file.

  The prototype of DBE_FILE.PUT_RAW is as follows:

```
DBE_FILE.PUT_RAW (
   file  IN INTEGER,
   r     IN RAW,
   flush IN BOOLEAN DEFAULT FALSE)
RETURN BOOLEAN;

DBE_FILE.PUT_RAW (
   file  IN DBE_FILE.FILE_TYPE,
   r     IN RAW,
   flush IN BOOLEAN DEFAULT FALSE)
RETURN VOID;
```

Table 2-163 DBE_FILE.PUT_RAW API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER or DBE_FILE.FILE_TYPE	IN	No	File handle opened using OPEN or object of the DBE_FILE.FILE_TYPE type opened using FOPEN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
r	RAW	IN	Yes	RAW data to be written to a file. If this parameter is set to **NULL**, no data is written. **NOTE** For the write operation, the system checks the file type. If the ELF file is written, an error is reported and the system exits.
flush	BOOLEAN	IN	Yes	Determines whether to flush data to disks after PUT_RAW. The default value is **FALSE**.

- DBE_FILE.FLUSH

  Writes cached data to a physical file. The cached data must have a line terminator.

  The prototype of DBE_FILE.FLUSH is as follows:

```
DBE_FILE.FLUSH(
   file IN INTEGER)
RETURN VOID;

DBE_FILE.FLUSH(
   file IN DBE_FILE.FILE_TYPE)
RETURN VOID;
```

  Table 2-164 DBE_FILE.FLUSH API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER or DBE_FILE.FILE_TYPE	IN	No	File handle opened using OPEN or object of the DBE_FILE.FILE_TYPE type opened using FOPEN.

- DBE_FILE.CLOSE

  Closes an open file. When this function is called, if there is cached data to be written, exception information may be received and the return value is always **TRUE**.

  The prototype of DBE_FILE.CLOSE is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_FILE.CLOSE(
   file IN INTEGER)
RETURN BOOLEAN;

DBE_FILE.CLOSE(
   file IN DBE_FILE.FILE_TYPE)
RETURN BOOLEAN;
```

Table 2-165 DBE_FILE.CLOSE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER or DBE_FILE.FILE_TYPE	IN	No	File handle opened using OPEN or object of the DBE_FILE.FILE_TYPE type opened using FOPEN.

- DBE_FILE.CLOSE_ALL

  Closes all file handles opened in a session. This function can be used for emergency cleanup.

  The prototype of DBE_FILE.CLOSE_ALL is as follows:

  ```
  DBE_FILE.CLOSE_ALL()
  RETURN VOID;
  ```

- DBE_FILE.REMOVE

  Removes a disk file. Only users with the required permission can execute this function.

  The prototype of DBE_FILE.REMOVE is as follows:

  ```
  DBE_FILE.REMOVE(
     dir       IN TEXT,
     file_name IN TEXT)
  RETURN VOID;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-166 DBE_FILE.REMOVE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
dir	TEXT	IN	No	Directory of a file. It is a string, indicating an object name. **NOTE** • Location of the file directory, which needs to be added to system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. • When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to operate files in the file path specified by **safe_data_path**.
file_name	TEXT	IN	No	File name.

- DBE_FILE.RENAME

  Renames a disk file. This function is similar to the **mv** command of Unix.

  The prototype of DBE_FILE.RENAME is as follows:

```
DBE_FILE.RENAME(
   dir        IN TEXT,
   src_file_name  IN TEXT,
   dest_dir       IN TEXT,
   dest_file_name IN TEXT,
   overwrite      IN BOOLEAN DEFAULT FALSE)
RETURN VOID;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-167 DBE_FILE.RENAME API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
dir	TEXT	IN	No	Directory of the source file (case-sensitive). **NOTE** • Location of the file directory, which needs to be added to system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. • When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to operate files in the file path specified by **safe_data_path**.
src_file_name	TEXT	IN	No	Source file to be renamed.
dest_dir	TEXT	IN	No	Target directory (case-sensitive). **NOTE** • Location of the file directory, which needs to be added to system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. • When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to operate files in the file path specified by **safe_data_path**.
dest_file_name	TEXT	IN	No	New file name.
overwrite	BOOLEAN	IN	Yes	Specifies whether to overwrite the file. If the parameter is left empty or not specified, the parameter is not overwritten. If no overwriting is performed and a file with the same name already exists in the destination directory, an error is reported.

- DBE_FILE.COPY

  Copies data in a continuous area to a new file. If **start_line** and **end_line** are omitted, the entire file is copied.

  The prototype of DBE_FILE.COPY is as follows:

```
DBE_FILE.COPY (
src_dir         IN     TEXT,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
src_file_name     IN     TEXT,
dest_dir          IN     TEXT,
dest_file_name    IN     TEXT,
start_line        IN     INTEGER DEFAULT 1,
end_line          IN     INTEGER DEFAULT NULL)
RETURN VOID;
```

Table 2-168 DBE_FILE.COPY API parameters

Parameter	Type	Input/OutputParameter	CanBeNull	Description
src_dir	TEXT	IN	No	Directory of the source file. **NOTE** • Location of the file directory, which needs to be added to system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. • When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to operate files in the file path specified by **safe_data_path**.
src_file_name	TEXT	IN	No	Name of the source file to be copied.
dest_dir	TEXT	IN	No	Directory of the target file. **NOTE** • Location of the file directory, which needs to be added to system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. • When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to operate files in the file path specified by **safe_data_path**.
dest_file_name	TEXT	IN	No	Target file name. **NOTE** For the write operation, the system checks the file type. If the ELF file is written, an error is reported and the system exits.
start_line	INTEGER	IN	No	Number of the line where the copy starts. The default value is **1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/OutputParameter	CanBeNull	Description
end_line	INTEGER	IN	Yes	Number of the line where the copy ends. The default value is **NULL**, indicating the end of the file.

- DBE_FILE.GET_ATTR

  Reads and returns the attributes of a disk file.

  The prototype of DBE_FILE.GET_ATTR is as follows:

```
DBE_FILE.GET_ATTR(
   location    IN  TEXT,
   filename    IN  TEXT,
   fexists     OUT BOOLEAN,
   file_length OUT BIGINT,
   block_size  OUT INTEGER)
RETURN RECORD;
```

Table 2-169 DBE_FILE.GET_ATTR API parameters

Parameter	Type	Input/OutputParameter	CanBeNull	Description
location	TEXT	IN	No	File directory. **NOTE** • Location of the file directory, which needs to be added to system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. • When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to operate files in the file path specified by **safe_data_path**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/OutputParameter	CanBeNull	Description
filename	TEXT	IN	No	File name.
fexists	BOOLEAN	OUT	No	Specifies whether the file exists.
file_length	BIGINT	OUT	No	File length (unit: bytes). If the file does not exist, **NULL** is returned.
block_size	INTEGER	OUT	No	Block size of the file system (unit: bytes). If the file does not exist, **NULL** is returned.

- DBE_FILE.SEEK

  Adjusts the position of a file pointer forward or backward based on the specified number of bytes.

  The prototype of DBE_FILE.SEEK is as follows:

```
DBE_FILE.SEEK(
   file          IN INTEGER,
   absolute_start IN BIGINT DEFAULT NULL,
   relative_start IN BIGINT DEFAULT NULL)
RETURN VOID;

DBE_FILE.SEEK(
   file          IN DBE_FILE.FILE_TYPE,
   absolute_start IN INTEGER DEFAULT NULL,
   relative_start IN INTEGER DEFAULT NULL)
RETURN VOID;
```

Table 2-170 DBE_FILE.SEEK API parameters

Parameter	Type	Input/OutputParameter	Can Be Null	Description
file	INTEGER or DBE_FILE.FILE_TYPE	IN	No	File handle opened using OPEN or object of the DBE_FILE.FILE_TYPE type opened using FOPEN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
absolute_start	BIGINT or INTEGER	IN	Yes	Absolute offset of a file. The default value is **NULL**.
relative_start	BIGINT or INTEGER	IN	Yes	Relative offset of a file. A positive number indicates forward offset and a negative number indicates backward offset. The default value is **NULL**. If both **absolute_start** and this parameter are specified, the **absolute_start** parameter is used.

- DBE_FILE.GET_POS

  Returns the current offset of a file, in bytes.

  The prototype of DBE_FILE.GET_POS is as follows:

```
DBE_FILE.GET_POS(
    file IN INTEGER)
RETURN BIGINT;

DBE_FILE.GET_POS(
    file IN DBE_FILE.FILE_TYPE)
RETURN BIGINT;
```

  Table 2-171 DBE_FILE.GET_POS API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	INTEGER or DBE_FILE.FILE_TYPE	IN	No	File handle opened using OPEN or object of the DBE_FILE.FILE_TYPE type opened using FOPEN.

- DBE_FILE.FOPEN_NCHAR

  Opens a file. You can specify the maximum line size. A maximum of 50 files can be opened in a session. This function returns a DBE_FILE.FILE_TYPE type object that encapsulates a file handle. This function opens a file in national character set mode for input or output.

  The prototype of DBE_FILE.FOPEN_NCHAR is as follows:

```
DBE_FILE.FOPEN_NCHAR(
    dir         IN TEXT,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  file_name     IN TEXT,
  open_mode     IN TEXT,
  max_line_size IN INTEGER DEFAULT 1024)
RETURN DBE_FILE.FILE_TYPE;
```

Table 2-172 DBE_FILE.FOPEN_NCHAR API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
dir	TEXT	IN	No	Directory of a file. The string indicates the name of a directory object. **NOTE** • Location of the file directory, which needs to be added to system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. • When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to read and write files in the file path specified by **safe_data_path**.
file_name	TEXT	IN	No	File name with an extension (file type), excluding the path name. A path contained in a file name is ignored in the FOPEN_NCHAR function. In Unix, the file name cannot end with the combination of a slash and a dot (/.).
open_mode	TEXT	IN	No	Open mode of a specified file, including: • **r**: read text. • **w**: write text. • **a**: append text. • **rb**: read byte. • **wb**: write byte. • **ab**: append byte. **NOTE** For the write operation, the system checks the file type. If the ELF file is written, an error is reported and the system exits.
max_line_size	INTEGER	IN	Yes	Maximum number of bytes in each line, including newline characters. The minimum value is **1** and the maximum value is **32767**. If this parameter is not specified, the default value **1024** is used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_FILE.WRITE_NCHAR

  Writes buffer data to the buffer of a file. The file must be opened in national character set or write mode. This operation does not write a line terminator. The text string is written in the UTF-8 character set format.

  The prototype of DBE_FILE.WRITE_NCHAR is as follows:

```
DBE_FILE.WRITE_NCHAR(
   file   IN DBE_FILE.FILE_TYPE,
   buffer IN NVARCHAR2)
RETURN VOID;
```

  Table 2-173 DBE_FILE.WRITE_NCHAR API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	DBE_FILE.FILE_TYPE	IN	No	Object of the DBE_FILE.FILE_TYPE type opened using FOPEN_NCHAR. The file must be opened in write mode. This operation does not write line terminators.
buffer	VARCHAR2	IN	Yes	Text data to be written to the file. The accumulated write length of each line cannot be greater than or equal to the value of **max_line_size** specified by **FOPEN_NCHAR**. Otherwise, an error is reported when the file is flushed. **NOTE** For the write operation, the system checks the file type. If the ELF file is written, an error is reported and the system exits.

- DBE_FILE.WRITE_LINE_NCHAR

  Writes buffer data to the buffer of a file. The file must be opened in national character set or write mode. This operation automatically adds a line terminator. The text string is written in the UTF-8 character set format.

  The prototype of DBE_FILE.WRITE_LINE_NCHAR is as follows:

```
DBE_FILE.WRITE_LINE_NCHAR(
   file   IN DBE_FILE.FILE_TYPE,
   buffer IN NVARCHAR2)
RETURN VOID;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-174 DBE_FILE.WRITE_LINE_NCHAR API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	DBE_FILE.FILE_TYPE	IN	No	Object of the DBE_FILE.FILE_TYPE type opened using FOPEN_NCHAR.
buffer	VARCHAR2	IN	Yes	Text data to be written to the file. The length of each line (including the newline character) cannot be greater than the value of **max_line_size** specified by **FOPEN_NCHAR** or the default value. Otherwise, an error is reported when the file is flushed. **NOTE** For the write operation, the system checks the file type. If the ELF file is written, an error is reported and the system exits.

- DBE_FILE.FORMAT_WRITE_NCHAR

  Writes formatted data to the buffer of an open file. It is a DBE_FILE.WRITE_NCHAR API that allows formatting.

  The prototype of DBE_FILE.FORMAT_WRITE_NCHAR is as follows:

```
DBE_FILE.FORMAT_WRITE_NCHAR(
   file   IN DBE_FILE.FILE_TYPE,
   format IN NVARCHAR2,
   arg1   IN NVARCHAR2 DEFAULT NULL,
   . . .
   arg5   IN NVARCHAR2 DEFAULT NULL)
RETURN VOID;
```

Table 2-175 DBE_FILE.FORMAT_WRITE_NCHAR API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	DBE_FILE.FILE_TYPE	IN	No	Object of the DBE_FILE.FILE_TYPE type opened using FOPEN_NCHAR.
format	VARCHAR2	IN	Yes	Formatted string, containing the text and format characters \n and %s.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
[arg1. . .arg5]	VARCHAR2	IN	Yes	Five optional parameters. The parameters and the positions of characters to be formatted are in one-to-one correspondence. If the parameter corresponding to a character to be formatted is not provided, an empty string is used to replace %s.

- DBE_FILE.READ_LINE_NCHAR

  Reads data from an open file and stores the read result to the buffer. It reads data to the end of each line excluding the line terminator, to the end of the file, or to the size specified by the **len** parameter. The length of the data to be read cannot exceed the value of **max_line_size** specified by **FOPEN_NCHAR**.

  The prototype of DBE_FILE.READ_LINE_NCHAR is as follows:

```
DBE_FILE.READ_LINE_NCHAR(
   file   IN  DBE_FILE.FILE_TYPE,
   buffer OUT NVARCHAR2,
   len    IN  INTEGER DEFAULT NULL)
RETURN NVARCHAR2;
```

Table 2-176 DBE_FILE.READ_LINE_NCHAR API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
file	DBE_FILE.FILE_TYPE	IN	No	Object of the DBE_FILE.FILE_TYPE type opened using FOPEN_NCHAR. The file must be opened in read mode. Otherwise, the INVALID_OPERATION exception is thrown.
buffer	VARCHAR2	OUT	No	Buffer for receiving data.
len	INTEGER	IN	Yes	Number of bytes read from the file. The default value is **NULL**. If the default value **NULL** is used, **max_line_size** is used to specify the line size.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Examples

```
-- Add the /tmp/ directory to the PG_DIRECTORY system catalog as a system administrator.
CREATE OR REPLACE DIRECTORY dir AS '/tmp/';
-- Execution result.
CREATE DIRECTORY
-- Use the DBE_FILE advanced package.
DECLARE
    f INTEGER;
    buffer VARCHAR2;
    raw_buffer RAW;

    f1 DBE_FILE.FILE_TYPE;
    f2 DBE_FILE.FILE_TYPE;

    fexists BOOLEAN;
    file_length BIGINT;
    block_size INTEGER;
    pos BIGINT;

    nvarchar_buffer nvarchar2;
    f_nchar DBE_FILE.FILE_TYPE;
BEGIN
    -- Open a file.
    f := DBE_FILE.OPEN('dir', 'sample.txt', 'w');

    IF DBE_FILE.IS_OPEN(f) = true THEN
        DBE_OUTPUT.PRINT_LINE('file opened');
    END IF;

    -- Close the file.
    DBE_FILE.CLOSE(f);

    IF DBE_FILE.IS_CLOSE(f) = true THEN
        DBE_OUTPUT.PRINT_LINE('file closed');
    END IF;

    f := DBE_FILE.OPEN('dir', 'sample.txt', 'w');

    -- Write a file.
    DBE_FILE.WRITE(f, 'A');
    DBE_FILE.NEW_LINE(f);
    DBE_FILE.WRITE(f, 'B');
    DBE_FILE.WRITE(f, 'C');
    DBE_FILE.NEW_LINE(f, 2);
    DBE_FILE.WRITE_LINE(f, 'ABC');
    DBE_FILE.FORMAT_WRITE(f, '[1 -> %s, 2 -> %s]\n', 'GaussDB', 'DBE_FILE');
    DBE_FILE.PUT_RAW(f, '414243');
    DBE_FILE.NEW_LINE(f);
    DBE_FILE.CLOSE(f);

    -- Create sample_copy.txt and copy the content of sample.txt.
    DBE_FILE.COPY('dir', 'sample.txt', 'dir', 'sample_copy.txt');

    -- Open a file in read mode.
    f := DBE_FILE.OPEN('dir', 'sample_copy.txt', 'r');
    -- Read a file.
    DBE_FILE.READ_LINE(f, buffer); -- A
    DBE_OUTPUT.PRINT_LINE(buffer);
    DBE_FILE.READ_LINE(f, buffer); -- BC
    DBE_OUTPUT.PRINT_LINE(buffer);
    DBE_FILE.READ_LINE(f, buffer);
    DBE_FILE.READ_LINE(f, buffer); -- ABC
    DBE_OUTPUT.PRINT_LINE(buffer);
    DBE_FILE.READ_LINE(f, buffer); -- [1 -> GaussDB, 2 -> DBE_FILE]
    DBE_OUTPUT.PRINT_LINE(buffer);
    DBE_FILE.READ_LINE(f, buffer); -- RAW 414243 --> ABC
    DBE_OUTPUT.PRINT_LINE(buffer);

    -- Close files.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  DBE_FILE.CLOSE(f);

  f1 := DBE_FILE.FOPEN('dir', 'sample1.txt', 'w');
  f2 := DBE_FILE.FOPEN('dir', 'sample2.txt', 'w');
  DBE_FILE.CLOSE_ALL();

  IF DBE_FILE.IS_CLOSE(f1) = true and DBE_FILE.IS_CLOSE(f2) = true THEN
     DBE_OUTPUT.PRINT_LINE('f1 and f2 all closed');
  END IF;

  -- Remove files.
  DBE_FILE.REMOVE('dir', 'sample1.txt');
  DBE_FILE.REMOVE('dir', 'sample2.txt');
  DBE_FILE.REMOVE('dir', 'sample_copy.txt');

  -- Open the sample.txt file and clear its data.
  f := DBE_FILE.OPEN('dir', 'sample.txt', 'w');
  DBE_FILE.WRITE_LINE(f, 'ABC');
  DBE_FILE.CLOSE(f);

  f := DBE_FILE.OPEN('dir', 'sample.txt', 'r');
  -- GET_RAW
  DBE_FILE.GET_RAW(f, raw_buffer); -- 0A of 4142430A is a newline character.
  DBE_OUTPUT.PRINT_LINE(raw_buffer);
  DBE_FILE.CLOSE(f);

  -- Obtain file attributes.
  DBE_FILE.GET_ATTR('dir', 'sample.txt', fexists, file_length, block_size);

  IF fexists = true THEN
     DBE_OUTPUT.PRINT_LINE('file length: ' || file_length);
   END IF;

  -- Rename a file.
  DBE_FILE.RENAME('dir', 'sample.txt', 'dir', 'sample_rename.txt', true);
  f1 := DBE_FILE.FOPEN('dir', 'sample_rename.txt', 'r');
  DBE_FILE.SEEK(f1, 1, null);
  pos := DBE_FILE.GET_POS(f1);
  DBE_OUTPUT.PRINT_LINE('position is: ' || pos);
  DBE_FILE.READ_LINE(f1, buffer); -- BC
  DBE_OUTPUT.PRINT_LINE(buffer);
  DBE_FILE.CLOSE(f1);

  -- FLUSH
  f1 := DBE_FILE.FOPEN('dir', 'sample_rename.txt', 'w');
  DBE_FILE.WRITE_LINE(f1, 'ABCEFG');
  DBE_FILE.FLUSH(f1);

  f2 := DBE_FILE.FOPEN('dir', 'sample_rename.txt', 'r');
  DBE_FILE.READ_LINE(f2, buffer); -- ABCEFG
  DBE_OUTPUT.PRINT_LINE(buffer);
  DBE_FILE.CLOSE(f1);
  DBE_FILE.CLOSE(f2);
  DBE_FILE.REMOVE('dir', 'sample_rename.txt');

  -- NCHAR function.
  f_nchar := DBE_FILE.FOPEN_NCHAR('dir', 'sample_nchar.txt', 'w');
  DBE_FILE.WRITE_NCHAR(f_nchar, 'ABCDE');
  DBE_FILE.WRITE_LINE_NCHAR(f_nchar, 'ABCDE');
  DBE_FILE.FORMAT_WRITE_NCHAR(f_nchar, '%s, %s', 'hello', 'world');
  DBE_FILE.CLOSE(f_nchar);
  f_nchar := DBE_FILE.FOPEN_NCHAR('dir', 'sample_nchar.txt', 'r');
  DBE_FILE.READ_LINE_NCHAR(f_nchar, nvarchar_buffer); -- ABCDEABCDE
  DBE_OUTPUT.PRINT_LINE(nvarchar_buffer);
  DBE_FILE.READ_LINE_NCHAR(f_nchar, nvarchar_buffer); -- hello, world
  DBE_OUTPUT.PRINT_LINE(nvarchar_buffer);
  DBE_FILE.CLOSE(f_nchar);
  DBE_FILE.REMOVE('dir', 'sample_nchar.txt');
END;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
/

-- Execution result.
file opened
file closed
A
BC
ABC
[1 -> GaussDB, 2 -> DBE_FILE]
ABC
f1 and f2 all closed
4142430A
file length: 4
position is: 1
BC
ABCEFG
ABCDEABCDE
hello, world
ANONYMOUS BLOCK EXECUTE
```

2.12.2.5 DBE_HEAT_MAP

API

DBE_HEAT_MAP returns information such as the last modification time of a row in the target data block based on the input parameters. The information is used to browse the basis for determining cold and hot rows. This API is an O&M API and has no visibility check. That is, if the input CTID is a deleted row, this API still returns the latest status of the current row on the page. **Table 2-177** lists APIs supported by DBE_HEAT_MAP. Cross-page transparent compression does not support this advanced package. In the query result of a cross-page transparently compressed table, the last modification time of rows is empty.

Table 2-177 DBE_HEAT_MAP

API	Description
DBE_HEAT_MAP.ROW_HEAT_MAP	Obtains information such as the last modification time of a row based on the schema to which the object belongs, data object name, data object partition name, and CTID.

- DBE_HEAT_MAP.ROW_HEAT_MAP

  Obtains information such as the last modification time of a row based on the schema to which the object belongs, data object name, data object partition name, and CTID.

  The prototype of DBE_HEAT_MAP.ROW_HEAT_MAP is as follows:

```
BE_HEAT_MAP.ROW_HEAT_MAP(
OWNER           IN VARCHAR2,
SEGMENT_NAME    IN VARCHAR2,
PARTITION_NAME  IN VARCHAR2     DEFAULT NULL,
CTID            IN TEXT,
V_DEBUG         IN BOOL         DEFAULT FALSE,
BUCKETID        IN pg_catalog.int2 DEFAULT NULL))
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-178 DBE_HEAT_MAP.ROW_HEAT_MAP API parameters

Parameter	Description
OWNER	Schema to which a data object belongs.
SEGMENT_NAME	Data object name.
PARTITION_NAME	Data object partition name. This parameter is optional. The default value is **NULL**.
CTID	**ctid** of the target row, that is, **block_id** or **row_id**.
V_DEBUG	Debugging. Debug logs are recorded.
BUCKETID	Table bucket ID of the target row.

Table 2-179 Output parameters of DBE_HEAT_MAP.ROW_HEAT_MAP

Parameter	Description
OWNER	Data object owner.
SEGMENT_NAME	Data object name.
PARTITION_NAME	Data object partition name. This parameter is optional.
TABLESPACE_NAME	Name of the tablespace to which data belongs.
FILE_ID	ID of the absolute file to which a row belongs.
RELATIVE_FNO	ID of the relative file to which a row belongs. (GaussDB does not have this logic. Therefore, the value is the same as the preceding value.)
CTID	**ctid** of a row, that is, **block_id** or **row_id**.
WRITETIME	Last modification time of a row.

NOTE

The DBE_HEAT_MAP.ROW_HEAT_MAP API supports only DNs in the distributed database.

Example:

```
gaussdb=# ALTER DATABASE set ilm = on;
gaussdb=# CREATE SCHEMA HEAT_MAP_DATA;
gaussdb=# SET current_schema=HEAT_MAP_DATA;

gaussdb=# CREATE TABLESPACE example1 RELATIVE LOCATION 'tablespace1';
gaussdb=# CREATE TABLE HEAT_MAP_DATA.heat_map_table(id INT, value TEXT) TABLESPACE
example1;
gaussdb=# INSERT INTO HEAT_MAP_DATA.heat_map_table VALUES (1, 'test_data_row_1');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT * FROM DBE_HEAT_MAP.ROW_HEAT_MAP(
    owner          =>  'heat_map_data',
    segment_name   =>  'heat_map_table',
    partition_name  => NULL,
    ctid           =>  '(0,1)');
     owner     |  segment_name  | partition_name | tablespace_name | file_id | relative_fno | ctid  |
writetime
---------------+----------------+----------------+-----------------+---------+--------------+-------+-----------
 heat_map_data | heat_map_table |                | example1        |   17291 |        17291 | (0,1) |
(1 row)
```

2.12.2.6 DBE_ILM

APIs

Implements ILM policies and APIs for manually triggering and stopping ADO tasks. **Table 2-180** lists APIs supported by DBE_ILM.

Table 2-180 DBE_ILM

API	Description
DBE_ILM.EXECUTE_ILM	Evaluates the specified data and ILM policy based on parameters. If the evaluation is passed, the corresponding compression job is generated.
DBE_ILM.STOP_ILM	Stops a compression job that is being executed based on parameters.

- DBE_ILM.EXECUTE_ILM

  Evaluates the specified data and ILM policy based on parameters. If the evaluation is passed, the corresponding compression job is generated.

  The prototype of DBE_ILM.EXECUTE_ILM is as follows:

```
DBE_ILM.EXECUTE_ILM(
OWNER          IN VARCHAR2,
OBJECT_NAME    IN VARCHAR2,
TASK_ID        IN OUT NUMBER,
SUBOBJECT_NAME IN VARCHAR2 DEFAULT NULL,
POLICY_NAME    IN VARCHAR2 DEFAULT 'ALL POLICIES',
EXECUTION_MODE IN NUMBER DEFAULT 2,
TURBO_FORCE    IN BOOLEAN DEFAULT FALSE,
TURBO_BLKSTART IN INTEGER DEFAULT -1,
TURBO_BLKEND   IN INTEGER DEFAULT -1)
```

Table 2-181 DBE_ILM.EXECUTE_ILM API parameters

Parameter	Description
OWNER	Schema to which an object belongs.
OBJECT_NAME	Object name.
TASK_ID	Descriptor ID of the generated ADO task.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
SUBOBJECT_NAME	Subobject name.
POLICY_NAME	Policy name. You can query the **GS_ADM_ILMOBJECTS** view to obtain the policy name. The default value of **DBE_ILM.ILM_ALL_POLICIES** is **'ALL POLICIES'**, indicating all policies on the object. Lowercase letters are not supported.
EXECUTION_MOD	Execution mode. This parameter is not supported in the current version and is reserved.
TURBO_FORCE	Specifies whether to enable forcible compression that ignores cold and hot data management. This function takes effect only for cross-page transparent compression. The value can be **TRUE** (enabled) or **FALSE** (disabled).
TURBO_BLKSTART	Number of the block from which forcible compression starts. This parameter takes effect only for cross-page transparent compression.
TURBO_BLKEND	Number of the block at which forcible compression stops. This parameter takes effect only for cross-page transparent compression.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- DBE_ILM.EXECUTE_ILM generates a job name in the format of *Database name_Table name_Partition name_Number* based on the current database name and table name, and then creates a job with the corresponding name by using DBE_SCHEDULER.Create_job. If the table name contains special characters, an error message "Invalid program name" will be displayed when this API is called.
- When DBE_ILM.EXECUTE_ILM and DBE_ILM.STOP_ILM are executed concurrently, there is a low probability that the job fails, and "tuple concurrently updated" is displayed in the **comments** column in the gs_adm_ilmresults view.
- When performing compression operations (either automatically or manually), you are advised not to perform DDL operations on the tables that are being compressed. In particular, executing DDL operations in a long transaction may cause deadlocks or lock waiting. If DDL operations are necessary, you are advised to call the DBE_ILM.STOP_ILM API to stop the compression jobs of the related tables, and then perform the DDL operations.
- To execute DBE_ILM.EXECUTE_ILM, you must have the CREATE JOB permission. You can grant the permissions using DBE_SCHEDULER.GRANT_USER_AUTHORIZATION. If you are not a SYSADMIN user, you must have the table owner permission.
- To compress a subpartition, the **SUBOBJECT_NAME** column cannot be set to the name of the partition. Instead, the name of the subpartition must be passed. (Alternatively, pass **NULL** to compress all subpartitions.)
- If this API is called within a transaction, the compression operation takes effect only after the transaction ends or is committed. If the DBE_ILM.EXECUTE_ILM API is called by multiple transactions in sequence and the first transaction is not committed, the subsequent transactions may encounter lock timeout.
- The value of the **TURBO_FORCE** parameter influences the gs_adm_ilmresults view as follows:
  - If this parameter is set to **FALSE**, the **statistics** field in the gs_adm_ilmresults view is meaningless for cross-page transparent compression. Therefore, the default value is displayed for cross-page transparent compression objects in this field.
  - If this parameter is set to **TRUE**, the current compression task runs independently of the original task division mechanism of ILM and does not collect statistics on the object. Therefore, the **statistics** field in the gs_adm_ilmresults view is empty.

- DBE_ILM.STOP_ILM

  Stops an ILM policy that is being executed based on parameters.

  The prototype of DBE_ILM.STOP_ILM is as follows:

```
DBE_ILM.STOP_ILM (
TASK_ID               IN NUMBER DEFAULT -1,
P_DROP_RUNNING_JOBS     IN BOOLEAN DEFAULT FALSE,
P_JOBNAME               IN VARCHAR2 DEFAULT NULL);
```

Table 2-182 DBE_ILM.STOP_ILM API parameters

Parameter	Description
TASK_ID	Specifies the descriptor ID of the ADO task to be stopped.
P_DROP_RUNNING_JOBS	Determines whether to stop a task that is being executed. The value **TRUE** indicates that the task is forcibly stopped, and the value **FALSE** indicates that the task is not stopped.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
P_JOBNAME	Specifies the name of the job to be stopped, which can be queried in the **GS_MY_ILMEVALUATIONDETAILS** view.

> **NOTE**
>
> When there are a large number of concurrent requests, the system may display the following message when you execute DBE_ILM.STOP_ILM: "Resources are busy, please try again later." In this case, try again later.

Examples

```
gaussdb=# CREATE DATABASE ilmtabledb with dbcompatibility = 'ORA';
gaussdb=# \c ilmtabledb
gaussdb=# ALTER DATABASE set ilm = on;
gaussdb=# CREATE Schema ILM_DATA;
gaussdb=# SET current_schema=ILM_DATA;
gaussdb=# CREATE SEQUENCE ILM_DATA.ORDER_TABLE_SE_ORDER_ID MINVALUE 1;
gaussdb=# CREATE OR REPLACE PROCEDURE ILM_DATA.ORDER_TABLE_CREATE_DATA(NUM INTEGER) IS
BEGIN
    FOR X IN 1..NUM
        LOOP
            INSERT INTO ORDER_TABLE VALUES(ORDER_TABLE_SE_ORDER_ID.nextval, 'Snack gift basket A',
NOW());
        END LOOP;
    COMMIT;
END;
/
gaussdb=# CREATE TABLE ILM_DATA.ORDER_TABLE (ORDER_ID INT, GOODS_NAME TEXT, CREATE_TIME
TIMESTAMP) WITH (STORAGE_TYPE=ASTORE) ILM ADD POLICY ROW STORE COMPRESS ADVANCED ROW
AFTER 1 DAYS OF NO MODIFICATION;
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'order_id' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
BEGIN
    ILM_DATA.ORDER_TABLE_CREATE_DATA(100);
    PERFORM PG_SLEEP(2);
END;
/
DECLARE
    v_taskid number;
BEGIN
    DBE_ILM.EXECUTE_ILM(OWNER        => 'ilm_data',
                        OBJECT_NAME    => 'order_table',
                        TASK_ID        => v_taskid,
                        SUBOBJECT_NAME => NULL,
                        POLICY_NAME    => 'ALL POLICIES',
                        EXECUTION_MODE => 2);
    RAISE INFO 'Task ID is:%', v_taskid;
END;
/
INFO:  Task ID is:1
SELECT * FROM pg_sleep(3);
 pg_sleep
----------

(1 row)

gaussdb=# EXECUTE DIRECT ON DATANODES 'SELECT A.DBNAME, A.JOB_STATUS, A.ENABLE,
A.FAILURE_MSG FROM PG_JOB A WHERE A.DBNAME = ''ilmtabledb'' AND A.JOB_NAME LIKE ''ilmjob$_%''
ORDER BY A.JOB_NAME DESC LIMIT 1';
  dbname  | job_status | enable | failure_msg
----------+------------+--------+-------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 dbeilmdb | s         | f     |
 dbeilmdb | s         | f     |
 dbeilmdb | s         | f     |
 dbeilmdb | s         | f     |
 dbeilmdb | s         | f     |
 dbeilmdb | s         | f     |
(6 rows)

gaussdb=# CALL DBE_ILM.STOP_ILM(-1, true, NULL);
 stop_ilm
----------

(1 row)
```

2.12.2.7 DBE_ILM_ADMIN

APIs

Implement ILM policies, and control ADO background scheduling and concurrency control parameters. **Table 2-183** lists APIs supported by DBE_ILM_ADMIN.

Table 2-183 DBE_ILM_ADMIN

API	Description
DBE_ILM_ADMIN.CUSTOMIZE_ILM	Customizes ILM policy attributes based on input parameters.
DBE_ILM_ADMIN.DISABLE_ILM	Disables background scheduling.
DBE_ILM_ADMIN.ENABLE_ILM	Enables background scheduling.
DBE_ILM_ADMIN.CREATE_ILM_DB_POLICY	Creates a database-level compression policy.
DBE_ILM_ADMIN.DELETE_ILM_DB_POLICY	Deletes a database-level compression policy.

NOTE

When there are a large number of concurrent requests and you run DBE_ILM_ADMIN.DISABLE_ILM or DBE_ILM_ADMIN.ENABLE_ILM, the system may display the message "Resources are busy, please try again later." In this case, try again later.

- DBE_ILM_ADMIN.CUSTOMIZE_ILM

  Customizes ILM policy attributes based on input parameters.

  The prototype of DBE_ILM_ADMIN.CUSTOMIZE_ILM is as follows:

  ```
  DBE_ILM_ADMIN.CUSTOMIZE_ILM(
  PARAM    IN    NUMBER,
  VAL      IN    NUMBER);
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-184 DBE_ILM_ADMIN.CUSTOMIZE_ILM API parameters

Parameter	Description
PARAM	Sequence number of a parameter.
VAL	Parameter value.

NOTE

In MYSQL- or M-compatible mode, if the input **VAL** is a non-numeric character when DBE_ILM_ADMIN.CUSTOMIZE_ILM() is called, for example, DBE_ILM_ADMIN.CUSTOMIZE_ILM(13, '*'), the input **VAL** is set to **0** by default.

Table 2-185 DBE_ILM_ADMIN.CUSTOMIZE_ILM API parameters

No.	Parameter Value	Description
1	EXECUTION_INTERVAL	Specifies the frequency of executing an ADO task, in minutes. The default value is **15**. The value is an integer or floating-point number greater than or equal to 1 and less than or equal to 2147483647. The value is rounded down.
2	RETENTION_TIME	Specifies the retention period of ADO-related history records, in days. The default value is **30**. The value is an integer or floating-point number greater than or equal to 1 and less than or equal to 2147483647. The value is rounded down.
7	ENABLE	Status of background scheduling. This parameter cannot be modified using DBE_ILM_ADMIN.CUSTOMIZE_ILM. Use DBE_ILM_ADMIN.DISABLE_ILM or DBE_ILM_ADMIN.ENABLE_ILM to modify it.
11	POLICY_TIME	Specifies whether the time unit of ADO is day or second. The time unit second is used only for testing. Possible values are: ● **0**: ILM_POLICY_IN_DAYS (default value) ● **1**: ILM_POLICY_IN_SECONDS
12	ABS_JOBLIMIT	Maximum number of ADO jobs generated by an ADO task. The value is an integer or floating-point number greater than or equal to 0 and less than or equal to 2147483647. The value is rounded down.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Parameter Value	Description
13	JOB_SIZELIMIT	Maximum number of bytes that can be processed by a single ADO job. The unit is MB. The value is an integer or floating-point number greater than or equal to 0 and less than or equal to 2147483647. The value is rounded down. When the value of **JOB_SIZELIMIT** is **0**, a warning is reported and no compression is performed when a compression task is manually executed. In addition, the automatic background scheduling job does not perform compression.
14	WIND_DURATION	Duration of the maintenance window, in minutes. The default value is **240** (4 hours). The value is an integer greater than or equal to 0 and less than 1440 (24 hours).
15	BLOCK_LIMITS	Upper limit of the instance-level row-store compression speed. The default value is **40**. The unit is blocks/ms, indicating the maximum number of blocks that can be compressed per millisecond. The value ranges from 0 to 10000. The value **0** indicates no limit. Cross-page transparent compression does not support this parameter.
16	ENABLE_META_COMPRESSION	Specifies whether to enable header compression. The default value is **0**. The value can be **0** (disabled) or **1** (enabled). Cross-page transparent compression does not support this parameter. **NOTE** If this parameter is set to **1**, the compression ratio is improved for tables with short rows (less data in a single row), but the performance of accessing compressed rows greatly deteriorates. If most tables in the database have long rows, you are advised not to enable this parameter.
17	SAMPLE_MIN	Minimum sampling step for constant encoding and equivalent encoding. The default value is **10**. The value range is [1,100]. Decimals can be entered and will be automatically rounded down. Cross-page transparent compression does not support this parameter.
18	SAMPLE_MAX	Maximum sampling step for constant encoding and equivalent encoding. The default value is **10**. The value range is [1,100]. Decimals can be entered and will be automatically rounded down. Cross-page transparent compression does not support this parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Parameter Value	Description
19	CONST_PRIO	Constant encoding priority. The default value is **40**. The value range is [0,100]. The value **100** indicates that constant encoding is disabled. Decimals can be entered and will be automatically rounded down. Cross-page transparent compression does not support this parameter.
20	CONST_THRESHOLD	Constant encoding threshold. The default value is **90**. The value range is [1,100], indicating that constant encoding is performed when the proportion of constant values in a column exceeds the threshold. Decimals can be entered and will be automatically rounded down. Cross-page transparent compression does not support this parameter.
21	EQVALUE_PRIO	Equivalent encoding priority. The default value is **60**. The value range is [0,100]. The value **100** indicates that equivalent encoding is disabled. Decimals can be entered and will be automatically rounded down. Cross-page transparent compression does not support this parameter.
22	EQVALUE_THRESHOLD	Equivalent encoding threshold. The default value is **80**. The value range is [1,100], indicating that equivalent encoding is performed when the ratio of equal values in two columns exceeds the threshold. Decimals can be entered and will be automatically rounded down. Cross-page transparent compression does not support this parameter.
23	ENABLE_DELTA_ENCODE_SWITCH	Specifies whether to enable difference encoding. The default value is **1**. Value range: [0, 1]. The value **0** indicates that the function is disabled and **1** indicates that the function is enabled. Decimals are supported and will be automatically rounded down. Cross-page transparent compression does not support this parameter.
24	LZ4_COMPRESSION_LEVEL	LZ4 compression level. The default value is **0**. The value range is [0,16]. Decimals can be entered and will be automatically rounded down. Cross-page transparent compression does not support this parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Parameter Value	Description
25	ENABLE_LZ4_PARTIAL_DECOMPRESSION	Specifies whether to enable partial decompression. The default value is **1**. Value range: [0,1]. The value **0** indicates that the function is disabled and **1** indicates that the function is enabled. Decimals are supported and will be automatically rounded down. Cross-page transparent compression does not support this parameter.
26	INDEXJOB_SIZELIMIT	Maximum number of bytes that can be processed by a single ADO job during index compression. The unit is MB. The default value is **1024**. The value is an integer or floating-point number greater than or equal to 0 and less than or equal to 2147483647. The value is rounded down. When the value of this parameter is **0**, a warning is reported and no compression is performed when a compression task is manually executed. In addition, the automatic background scheduling job does not perform compression.
27	ENABLE_INDEX_COMPRESS	Specifies whether to enable index compression. The value **0** (default value) indicates that index compression is disabled, and the value **1** indicates that index compression is enabled.

Example:

```
gaussdb=# CALL DBE_ILM_ADMIN.CUSTOMIZE_ILM(1, 15);
 customize_ilm
---------------


(1 row)
gaussdb=# SELECT * FROM gs_adm_ilmparameters;
              name               | value
---------------------------------+-------
 EXECUTION_INTERVAL              |    15
 RETENTION_TIME                  |    30
 ENABLED                         |     1
 POLICY_TIME                     |     0
 ABS_JOBLIMIT                    |    10
 JOB_SIZELIMIT                   |  1024
 WIND_DURATION                   |   240
 BLOCK_LIMITS                    |    40
 ENABLE_META_COMPRESSION         |     0
 SAMPLE_MIN                      |    10
 SAMPLE_MAX                      |    10
 CONST_PRIO                      |    40
 CONST_THRESHOLD                 |    90
 EQVALUE_PRIO                    |    60
 EQVALUE_THRESHOLD               |    80
 ENABLE_DELTA_ENCODE_SWITCH      |     1
 LZ4_COMPRESSION_LEVEL           |     0
 ENABLE_LZ4_PARTIAL_DECOMPRESSION |     1
 INDEXJOB_SIZELIMIT              |  1024
 ENABLE_INDEX_COMPRESS           |     0
(20 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_ILM_ADMIN.DISABLE_ILM

  Disables background scheduling.

  The prototype of DBE_ILM_ADMIN.DISABLE_ILM is as follows:

  ```
  CALL DBE_ILM_ADMIN.DISABLE_ILM();
  ```

- DBE_ILM_ADMIN.ENABLE_ILM

  Enables background scheduling.

  The prototype of DBE_ILM_ADMIN.ENABLE_ILM is as follows:

  ```
  CALL DBE_ILM_ADMIN.ENABLE_ILM();
  ```

  NOTE

  To make background scheduling take effect, you need to set the GUC parameter **enable_ilm** to **on** on the database O&M platform.

- DBE_ILM_ADMIN.CREATE_ILM_DB_POLICY

  Creates a database-level compression policy based on input parameters. The policy performs the following operations:

  – Automatically overwriting existing tables: This policy is added to all user tables that are not bound to a compression policy in the current database.

  – Default inheritance for new tables: If no compression policy is explicitly specified for a new table, the table automatically inherits the database-level policy.

  The prototype of DBE_ILM_ADMIN.CREATE_ILM_DB_POLICY is as follows:

  ```
  DBE_ILM_ADMIN.CREATE_ILM_DB_POLICY(
  DAYS IN INT,
  COMPRESSION_LEVEL IN TEXT DEFAULT 'MEDIUM');
  ```

  Table 2-186 DBE_ILM_ADMIN.CREATE_ILM_DB_POLICY API parameters

Parameter	Description
DAYS	Number of days of the compression policy.
COMPRESSION_LEVEL	Compression policy level. Only **HIGH** and **MEDIUM** are supported. The default value is **MEDIUM**.

- DBE_ILM_ADMIN.DELETE_ILM_DB_POLICY

  Deletes a database-level compression policy and the compression policies inherited from the database-level compression policy on all tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- Only advanced database-level compression policies are supported.
- Database-level policies do not support expressions.
- When using gs_dump to export the database-level policy definition, ensure that the entire database is exported and you have the system administrator permission. If the preceding conditions are not met, the related policies are exported as independent policies at the table, partition, or subpartition level instead of being processed as database-level policies.
- If the DBE_ILM_ADMIN.CREATE_ILM_DB_POLICY, DBE_ILM_ADMIN.DELETE_ILM_DB_POLICY, and DBE_ILM.EXECUTE_ILM APIs are called by multiple transactions concurrently and the first transaction is not committed, the subsequent transactions may encounter lock timeout.

Example of a database-level policy:

- Create a compression policy that takes effect for 10 days.

```
gaussdb=# CALL DBE_ILM_ADMIN.CREATE_ILM_DB_POLICY(10);
 create_ilm_db_policy
----------------------


(1 row)
```

- Delete a database-level policy.

```
gaussdb=# CALL DBE_ILM_ADMIN.DELETE_ILM_DB_POLICY();
 delete_ilm_db_policy
----------------------


(1 row)
```

- Create a high-level compression policy that takes effect for 10 days.

```
gaussdb=# CALL DBE_ILM_ADMIN.CREATE_ILM_DB_POLICY(10, 'high');
 create_ilm_db_policy
----------------------


(1 row)
```

2.12.2.8 DBE_LICENSE

NOTE

Only SYSADMIN or OPRADMIN can call the APIs in the advanced package.

APIs

The advanced package DBE_LICENSE is used to activate, view, and revoke licenses. For details about the supported APIs, see **Table DBE_LICENSE**.

Table 2-187 DBE_LICENSE

API	Description
DBE_LICENSE.ACTIVATE	Activates a license.
DBE_LICENSE.DEACTIVATE	Revokes the current license.
DBE_LICENSE.GET_ESN	Queries the ESN of a node.
DBE_LICENSE.GET_LICENSE_INFO	Queries the license status of the current node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_LICENSE.GET_ALL_NODE_LICENSE_ACTIVATE	Queries the license activation status of the database cluster.

- DBE_LICENSE.ACTIVATE

  Activates a license based on the license type and the corresponding file.

  Currently, only the token file or ESDP license file can be activated.

  The prototype of DBE_LICENSE.ACTIVATE is as follows:

```
DBE_LICENSE.ACTIVATE(
    license_type IN VARCHAR2,
    token_file IN VARCHAR2)
RETURN BOOLEAN;

DBE_LICENSE.ACTIVATE(
    license_type IN VARCHAR2,
    license_file IN VARCHAR2,
    key_file IN VARCHAR2)
RETURN BOOLEAN;
```

  Table 2-188 DBE_LICENSE.ACTIVATE API parameters

Parameter	Description
license_type	Type of the license to be activated. The value can be **OnCloud** or **OffCloud**. • **OnCloud**: The license is activated using a token file. • **OffCloud**: The license is activated using an ESDP license file.
token_file	Name of the token file. The following requirements must be met: • The name cannot be **license.json** or **license.xml**. • The name cannot contain slashes (/) or backslashes (\).
license_file	Name of the license file in ESDP format.
key_file	Public key file name.

- DBE_LICENSE.DEACTIVATE

  Deactivates the license of the current node.

  If the input parameter is **false**, the function throws an error when the license is deactivated. If the input parameter is **true**, the function returns **true** when the license is deactivated.

  If no parameter is passed, the default value **false** is used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- After the license is deactivated, it becomes invalid. The system will generate a new ESN for the current database node and the ESN is not activated. To protect your rights and interests, exercise caution when performing this operation.
- When you use an ESDP license file to activate the license, the revocation code file **RevoTicket.txt** is generated in the **gs_esn** folder in the data directory. You can use this file to update the ESN of the license on the ESDP platform and activate the node license again.

The prototype of DBE_LICENSE.DEACTIVATE is as follows:

```
DBE_LICENSE.DEACTIVATE(
    idempotent    IN BOOLEAN DEFAULT FALSE)
RETURN BOOLEAN;
```

- DBE_LICENSE.GET_ESN

  Queries the ESN of the current database node. The ESN is the unique identifier of the database node. The database ensures that the ESN of each node is unique.

  The prototype of DBE_LICENSE.GET_ESN is as follows:

  ```
  DBE_LICENSE.GET_ESN()
  RETURN VARCHAR2;
  ```

- DBE_LICENSE.GET_LICENSE_INFO

  Queries the license information of a node.

  The prototype of DBE_LICENSE.GET_LICENSE_INFO is as follows:

  ```
  DBE_LICENSE.GET_LICENSE_INFO()
  ```

  Return type: setof record

  Examples:

  ```
  gaussdb=# SELECT * FROM  dbe_license.get_license_info();
       create_time       |     activate_time      |      expire_time       |  type   | license_version | customer |
  state   |       lsn       |                                 feature_list
                                  | dn_capacity | cn_capacity
  ------------------------+------------------------+------------------------+---------+-----------------+----------
  +-----------+-----------------+-------------------------------------------------------------
  ----------------------------------+-------------+-------------
   2024-10-20 14:30:45+08 | 2026-01-22 12:02:17+08 | 2026-01-22 12:03:45+08 | OnCloud |               1 |
  Test     | Activated | lsntesttesttest | enable_ilm,enable_tde,enable_ledger,enable
  _security_policy |           1 |           1
  (1 row)
  ```

Table 2-189 DBE_LICENSE.GET_LICENSE_INFO API return columns

Name	Type	Description
create_time	timestamptz(0)	Time when a license is created.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
activate_time	timestamptz(0)	License activation time. • For an activated commercial license, the activation time is the time when the license is activated by calling the API. • For a trial license, the activation time is the time when the trial mode is enabled.
expire_time	timestamptz(0)	Time when the license expires. • For an activated license, the expiration time is the time when the token file or ESDP license file expires. • For a trial license, the expiration time is the 90th day after the trial mode is enabled.
type	text	License deployment type. • **OnCloud**: with a console deployed. • **OffCloud**: without a console deployed. • Others: illegal deployment mode.
license_version	bigint	License version. • In the case of deployment with a console, this parameter is used to distinguish the signature algorithm and license structure. • **0** for deployment without a console.
customer	text	Purchaser of a license.
state	text	License status. • **Activated**: The license has been activated. • **Deactivated**: The license has been deactivated. • **Expired**: The license has expired. • **Trial**: The license is in trial state. • **Error**: The license is abnormal.
lsn	text	License identifier.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
feature_list	text	List of advanced features contained in the license, which is a list of GUC parameter names separated by commas (,). For example, "enable_tde,enable_ledger".
dn_capacity	bigint	Reserved.
cn_capacity	bigint	Reserved.

- DBE_LICENSE.GET_ALL_NODE_LICENSE_ACTIVATE

  Queries the license activation status of the current database cluster. If the licenses of all CNs and primary DNs in the cluster are activated or in trial state, **true** is returned. Otherwise, **false** is returned.

  The prototype of DBE_LICENSE.GET_ALL_NODE_LICENSE_ACTIVATE is as follows:

```
DBE_LICENSE.GET_ALL_NODE_LICENSE_ACTIVATE()
RETURN BOOLEAN;
```

2.12.2.9 DBE_LOB

APIs

Table 2-190 lists APIs supported by the DBE_LOB package.

NOTE

- In database ORA, the byte content of the space is 00. However, in GaussDB, the byte content corresponding to the space is ASCII code 32.
- In a distributed environment, the maximum size of a CLOB, BLOB, or BFILE is 1 GB.
- LOBMAXSIZE supports a maximum of 1073741771 bytes.
- LOB update APIs (such as LOB_WRITE, LOB_COPY, CONVERTTOBLOB, LOB_CONVERTTOBLOB, and LOADBLOBFROMBFILE) in the DBE_LOB package update only LOB variables in memory and do not automatically update the associated physical table data. Even if the target row has been locked using SELECT ... FOR UPDATE, LOB data changes are not automatically persisted to the database table. To save the update result, you must explicitly execute the UPDATE statement.

Table 2-190 DBE_LOB

API	Description
DBE_LOB.GET_LENGTH	Obtains and returns the length of a specified LOB. The object cannot be greater than 2 GB.
DBE_LOB.LOB_GET_LENGTH	Obtains and returns the length of a specified LOB or BFILE.
DBE_LOB.OPEN	Opens a LOB and returns a LOB descriptor.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_LOB.READ	Loads a part of LOB content to the buffer based on the specified length and initial position offset.
DBE_LOB.LOB_READ	Reads a part of LOB (including BFILE) content to the buffer based on the specified length and initial position offset.
DBE_LOB.WRITE	Copies content in the buffer to a LOB based on the specified length and initial position offset.
DBE_LOB.WRITE_APPEND	Copies content in the buffer to the end part of a LOB based on the specified length.
DBE_LOB.LOB_WRITE_APPEND	Copies content in the buffer to the end part of a LOB based on the specified length.
DBE_LOB.COPY	Copies content in a LOB to another LOB based on the specified length and initial position offset.
DBE_LOB.LOB_COPY	Copies content in a LOB to another LOB based on the specified length and initial position offset.
DBE_LOB.ERASE	Deletes content in a LOB (less than or equal to 1 GB) based on the specified length and initial position offset.
DBE_LOB.LOB_ERASE	Deletes content in a LOB based on the specified length and initial position offset.
DBE_LOB.CLOSE	Closes a LOB descriptor.
DBE_LOB.MATCH	Returns the position of the *N*th occurrence of a character string in a LOB.
DBE_LOB.COMPARE	Compares two LOBs (including BFILE objects) or a certain part of two LOBs.
DBE_LOB.SUBSTR	Reads a LOB substring and returns the read substring.
DBE_LOB.LOB_SUBSTR	Reads a LOB or BFILE substring and returns the read substring.
DBE_LOB.STRIP	Truncates the LOB of a specified length. After the execution is complete, the length of the LOB is set to the length specified by the **newlen** parameter.
DBE_LOB.LOB_STRIP	Truncates the LOB of a specified length. After the execution is complete, the length of the LOB is set to the length specified by the **newlen** parameter.
DBE_LOB.CREATE_TEMPORARY	Creates a temporary BLOB or CLOB.
DBE_LOB.APPEND	Adds the content of a LOB to another LOB.
DBE_LOB.LOB_APPEND	Adds the content of a LOB to another LOB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_LOB.FREETEMPORARY	Deletes a temporary BLOB or CLOB.
DBE_LOB.FILEOPEN	Opens a database BFILE and returns its file descriptor.
DBE_LOB.FILECLOSE	Closes a BFILE that is opened by FILEOPEN.
DBE_LOB.BFILEOPEN	Opens a database BFILE.
DBE_LOB.BFILECLOSE	Closes a BFILE that is opened by BFILEOPEN.
DBE_LOB.LOADFROMFILE	Loads the database BFILE of the specified length in the specified location to the BLOB in the specified location.
DBE_LOB.LOADFROMBFILE	Loads the database BFILE of the specified length in the specified location to the LOB in the specified location.
DBE_LOB.LOADBLOBFROMFILE	Loads an external database file of the specified length in the specified location to a BLOB (less than or equal to 1 GB) in a specified location.
DBE_LOB.LOADBLOBFROMBFILE	Loads the database BFILE of the specified length in the specified location to the BLOB in the specified location.
DBE_LOB.LOADCLOBFROMFILE	Loads an external database file of the specified length in the specified location to a CLOB (less than or equal to 1 GB) in a specified location.
DBE_LOB.LOADCLOBFROMBFILE	Loads the database BFILE of the specified length in the specified location to the CLOB in the specified location.
DBE_LOB.CONVERTTOBLOB	Converts a CLOB to a BLOB (less than or equal to 1 GB).
DBE_LOB.CONVERTTOCLOB	Converts a BLOB to a CLOB (less than or equal to 1 GB).
DBE_LOB.LOB_CONVERTTOBLOB	Converts a CLOB to a BLOB.
DBE_LOB.LOB_CONVERTTOCLOB	Converts a BLOB to a CLOB.
DBE_LOB.GETCHUNKSIZE	Obtains the maximum size of LOB data that can be stored in the chunk structure in the database.
DBE_LOB.LOB_WRITE	Reads the specified length of the source object from the start position, writes the content to the specified offset position of the target LOB, overrides the original content, and returns the target LOB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_LOB.BFILENAME	Constructs and returns the DBE_LOB.BFILE based on the directory and file name.

- DBE_LOB.GET_LENGTH

  Obtains and returns the size of a specified LOB. The maximum size of the object is 2 GB.

  The prototype of DBE_LOB.GET_LENGTH is as follows:

```
DBE_LOB.GET_LENGTH (
   blob_obj IN BLOB)
RETURN INTEGER;

DBE_LOB.GET_LENGTH (
   clob_obj IN CLOB)
RETURN INTEGER;
```

  Table 2-191 DBE_LOB.GET_LENGTH API parameters

Parameter	Description
blob_obj/ clob_obj	BLOB/CLOB whose length is to be obtained.

- DBE_LOB.LOB_GET_LENGTH

  Obtains and returns the length of a specified LOB or BFILE file. The maximum size of the object is 32 TB.

  The prototype of DBE_LOB.LOB_GET_LENGTH is as follows:

```
DBE_LOB.LOB_GET_LENGTH (
   blob_obj IN BLOB)
RETURN BIGINT;

DBE_LOB.LOB_GET_LENGTH (
   clob_obj IN CLOB)
RETURN BIGINT;

DBE_LOB.LOB_GET_LENGTH (
   bfile IN DBE_LOB.BFILE)
RETURN BIGINT;
```

  Table 2-192 DBE_LOB.LOB_GET_LENGTH API parameters

Parameter	Description
blob_obj/ clob_obj/bfile	BLOB/CLOB/BFILE whose length is to be obtained.

- DBE_LOB.OPEN

  The stored procedure opens a LOB and returns a LOB descriptor. This procedure is meaningless and is used only for compatibility.

  The prototype of DBE_LOB.OPEN is as follows:

```
DBE_LOB.OPEN (
   lob INOUT BLOB);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_LOB.OPEN (
   lob INOUT CLOB);

DBE_LOB.OPEN (
   bfile     INOUT DBE_LOB.BFILE,
   open_mode IN    TEXT DEFAULT 'null');
```

Table 2-193 DBE_LOB.OPEN API parameters

Parameter	Description
lob/bfile	Opened BLOB/CLOB/BFILE.
open_mode	Operation mode. Currently, the range is [R,W,A,RB,WB,AB].

- DBE_LOB.READ

  Reads a part of LOB to the output buffer based on the specified length and initial position offset.

  The prototype of DBE_LOB.READ is as follows:

```
DBE_LOB.READ (
   blob_obj IN  BLOB,
   amount   IN  INTEGER,
   off_set  IN  INTEGER,
   out_put  OUT RAW);

DBE_LOB.READ (
   clob_obj IN  CLOB,
   amount   IN  INTEGER,
   off_set  IN  INTEGER,
   out_put  OUT VARCHAR2);
```

Table 2-194 DBE_LOB.READ API parameters

Parameter	Description
blob_obj/ clob_obj	BLOB/CLOB to be read.
amount	Length of read content. **NOTE** If the length to read is less than 1 or greater than 32767, an error is reported.
off_set	Start position for reading the LOB, that is, the offset bytes to initial position of the LOB content. If the offset is less than 1 or greater than the LOB length, an error is reported. The initial position is 1.
out_put	Target buffer for storing the read LOB.

- DBE_LOB.LOB_READ

  Reads a part of LOB/BFILE to the output buffer based on the specified length and initial position offset.

  The prototype of DBE_LOB.LOB_READ is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_LOB.LOB_READ(
   blob_obj IN    BLOB,
   amount   INOUT BIGINT,
   off_set  IN    BIGINT,
   out_put  OUT   RAW);

DBE_LOB.LOB_READ(
   clob_obj IN    CLOB,
   amount   INOUT BIGINT,
   off_set  IN    BIGINT,
   out_put  OUT   VARCHAR2);

DBE_LOB.LOB_READ(
   bfile   IN    DBE_LOB.BFILE,
   amount  INOUT BIGINT,
   off_set IN    BIGINT,
   out_put OUT   RAW);
```

Table 2-195 DBE_LOB.LOB_READ API parameters

Parameter	Description
blob_obj/ clob_obj/bfile	BLOB/CLOB/BFILE to be read (which can be greater than 1 GB).
amount	Length to read as the IN parameter, or actual read length as the OUT parameter. **NOTE** If the length to read is less than 1 or greater than 32767, an error is reported.
off_set	Start position for reading the LOB/BFILE, that is, the offset bytes to initial position of the LOB content. If the offset is less than 1 or greater than the LOB/BFILE length, an error is reported. The initial position is 1.
out_put	Target buffer for storing the read LOB/BFILE.

- DBE_LOB.WRITE

  Writes content in the source to a LOB based on the specified length and initial position.

  The prototype of DBE_LOB.WRITE is as follows:

```
DBE_LOB.WRITE (
   blob_obj INOUT BLOB,
   amount   IN    INTEGER,
   off_set  IN    INTEGER,
   source   IN    RAW);

DBE_LOB.WRITE (
   clob_obj INOUT CLOB,
   amount   IN    INTEGER,
   off_set  IN    INTEGER,
   source   IN    VARCHAR2);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-196 DBE_LOB.WRITE API parameters

Parameter	Description
blob_obj/ clob_obj	BLOB/CLOB to be written.
amount	Write size. A maximum of 32767 bytes are supported. **NOTE** If the length of written content is shorter than 1 or longer than the length of content to be written, an error is reported.
off_set	Start position for writing content to the target BLOB/ CLOB, that is, the offset bytes to the initial position of LOB content. **NOTE** If the offset is less than 1 or greater than the value of **LOBMAXSIZE**, an error is reported. The initial position is 1, and the maximum value is the maximum LOB length.
source	Content to be written.

- DBE_LOB.WRITE_APPEND

  Writes content in the source object to the end part of a LOB based on the specified length.

  The prototype of DBE_LOB.WRITE_APPEND is as follows:

```
DBE_LOB.WRITE_APPEND (
   blob_obj   INOUT BLOB,
   amount     IN    INTEGER,
   source_obj IN    RAW);

DBE_LOB.WRITE_APPEND (
   clob_obj   INOUT CLOB,
   amount     IN    INTEGER,
   source_obj IN    VARCHAR2);
```

Table 2-197 DBE_LOB.WRITE_APPEND API parameters

Parameter	Description
blob_obj/ clob_obj	BLOB/CLOB to be written.
amount	Write size. A maximum of 32767 bytes are supported. **NOTE** If the length of written content is shorter than 1 or longer than the length of content to be written, an error is reported.
source_obj	Content to be written.

- DBE_LOB.LOB_WRITE_APPEND

  Writes content in the source object to the end part of a LOB based on the specified length.

  The prototype of DBE_LOB.LOB_WRITE_APPEND is as follows:

```
DBE_LOB.LOB_WRITE_APPEND(
   blob_obj    INOUT BLOB,
```

```
  amount      IN    INTEGER,
  source_obj  IN    RAW);

DBE_LOB.LOB_WRITE_APPEND (
  clob_obj    INOUT CLOB,
  amount      IN    INTEGER,
  source_obj  IN    VARCHAR2);
```

Table 2-198 DBE_LOB.LOB_WRITE_APPEND API parameters

Parameter	Description
blob_obj/ clob_obj	BLOB/CLOB to be written.
amount	Write size. A maximum of 32767 bytes are supported. **NOTE** If the length of written content is shorter than 1 or longer than the length of content to be written, an error is reported.
source_obj	Content to be written.

- DBE_LOB.COPY

  Copies content in a LOB to another LOB based on the specified length and initial position offset.

  The prototype of DBE_LOB.COPY is as follows:

```
DBE_LOB.COPY (
  dest_lob   INOUT BLOB,
  src_lob    IN    BLOB,
  len        IN    INTEGER,
  dest_start IN    INTEGER DEFAULT 1,
  src_start  IN    INTEGER DEFAULT 1);
```

Table 2-199 DBE_LOB.COPY API parameters

Parameter	Description
dest_lob	LOB to which the buffer content is to be pasted.
src_lob	LOB from which the buffer content is to be copied.
len	Length of the copied data.
dest_start	Start position for pasting the buffer content to the target LOB (**dest_lob**), that is, the offset bytes to the initial position of LOB content.
src_start	Start position for copying the buffer content from the source LOB (**src_lob**), that is, the offset bytes to the initial position of LOB content.

- DBE_LOB.LOB_COPY

  Copies content in a LOB to another LOB based on the specified length and initial position offset.

  The prototype of DBE_LOB.LOB_COPY is as follows:

```
DBE_LOB.LOB_COPY(
  blob_obj    INOUT BLOB,
```

```
  source_obj  IN    BLOB,
  amount      IN    BIGINT,
  dest_offset IN    BIGINT DEFAULT 1,
  src_offset  IN    BIGINT DEFAULT 1);

DBE_LOB.LOB_COPY(
  clob_obj    INOUT CLOB,
  source_obj  IN    CLOB,
  amount      IN    BIGINT,
  dest_offset IN    BIGINT DEFAULT 1,
  src_offset  IN    BIGINT DEFAULT 1);
```

Table 2-200 DBE_LOB.LOB_COPY API parameters

Parameter	Description
blob_obj/ clob_obj	LOB to which the buffer content is to be pasted.
source_obj	LOB from which the buffer content is to be copied.
amount	Length of the copied data. **NOTE** If the length to copy is less than 1 or greater than the value of **LOBMAXSIZE**, an error is reported.
dest_offset	Start position for pasting the buffer content to the target LOB, that is, the offset bytes/characters to the initial position of LOB content. The unit is byte for BLOB and character for CLOB. **NOTE** If the offset is less than 1 or greater than the value of **LOBMAXSIZE**, an error is reported.
src_offset	Start position for copying the content from the source object, that is, the offset bytes to the initial position of LOB content. The unit is byte for BLOB and character for CLOB. **NOTE** If the offset is less than 1, an error is reported.

- DBE_LOB.ERASE

  Deletes the content in a BLOB (not greater than 1 GB) based on the specified length and initial position offset. The bytes of the deleted part in the BLOB are filled with 0.

  The prototype of DBE_LOB.ERASE is as follows:

```
DBE_LOB.ERASE (
  blob_obj INOUT BLOB,
  amount   INOUT INTEGER,
  off_set  IN    INTEGER DEFAULT 1);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-201 DBE_LOB.ERASE API parameters

Parameter	Description
blob_obj	LOB whose content is to be deleted as the IN parameter, or LOB with specified content deleted as the OUT parameter. If this parameter is left empty, an error is reported.
amount	Length (in bytes for BLOBs) to delete as the IN parameter, or actual length deleted as the OUT parameter. **NOTE** If the length to delete is less than 1 or this parameter is left empty, an error is reported.
off_set	Start position from which LOB is to be deleted, that is, the number of offset bytes to the initial position of LOB content. The value cannot be greater than 1 GB. **NOTE** If the offset is less than 1 or this parameter is left empty, an error is reported.

- DBE_LOB.LOB_ERASE

  Deletes the content in the LOB based on the specified length and initial position offset. The bytes of the deleted part in the BLOB are filled with 0, and the characters of the deleted part in the CLOB are filled with spaces. The LOB can be greater than 1 GB and the maximum size is 32 TB.

  The prototype of DBE_LOB.LOB_ERASE is as follows:

```
DBE_LOB.LOB_ERASE (
   blob_obj INOUT BLOB,
   amount   INOUT BIGINT,
   off_set  IN    BIGINT DEFAULT 1);

DBE_LOB.LOB_ERASE (
   clob_obj INOUT CLOB,
   amount   INOUT BIGINT,
   off_set  IN    BIGINT DEFAULT 1);
```

Table 2-202 DBE_LOB.LOB_ERASE API parameters

Parameter	Description
blob_obj/ clob_obj	LOB whose content is to be deleted as the IN parameter, or LOB with specified content deleted as the OUT parameter. If this parameter is left empty, an error is reported.
amount	Length (BLOB in bytes and CLOB in characters) to delete as the IN parameter, or actual length deleted as the OUT parameter. **NOTE** If the length to delete is less than 1 or this parameter is left empty, an error is reported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
off_set	Start position from which the LOB content is to be deleted, that is, the number of bytes relative to the start position of the BLOB content or the number of characters relative to the start position of the CLOB content. **NOTE** If the offset is less than 1 or this parameter is left empty, an error is reported.

- DBE_LOB.CLOSE

  Closes the LOB descriptor that has been opened.

  The prototype of DBE_LOB.CLOSE is as follows:

```
DBE_LOB.CLOSE(
   lob IN BLOB);

DBE_LOB.CLOSE (
   lob IN CLOB);

DBE_LOB.CLOSE (
   file IN INTEGER);
```

  Table 2-203 DBE_LOB.CLOSE API parameters

Parameter	Description
lob/file	BLOB/CLOB/BFILE to be closed.

- DBE_LOB.MATCH

  The function returns the *N*th occurrence position of a string in a LOB or BFILE. **NULL** is returned for an invalid input. The LOB or BFILE can be greater than 1 GB, up to 32 TB.

  The prototype of DBE_LOB.MATCH is as follows:

```
DBE_LOB.MATCH(
   blob_obj    IN BLOB,
   blob_obj2   IN RAW,
   beg_index   IN BIGINT DEFAULT 1,
   occur_index IN BIGINT DEFAULT 1)
RETURN BIGINT;

DBE_LOB.MATCH(
   clob_obj    IN CLOB,
   clob_obj2   IN VARCHAR2,
   beg_index   IN BIGINT DEFAULT 1,
   occur_index IN BIGINT DEFAULT 1)
RETURN BIGINT;

DBE_LOB.MATCH(
   bfile       IN DBE_LOB.BFILE,
   blob_obj2     IN RAW,
   beg_index   IN BIGINT DEFAULT 1,
   occur_index IN BIGINT DEFAULT 1)
RETURN BIGINT;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-204 DBE_LOB.MATCH API parameters

Parameter	Description
blob_obj/ clob_obj/ bfile	BLOB/CLOB descriptor to be searched for, or BFILE (which must be opened by DBE_LOB.BFILEOPEN first). If this parameter is left blank, **NULL** is returned.
blob_obj2/ clob_obj2	Pattern to match. It is RAW for BLOB/BFILE objects and VARCHAR2 for CLOBs. If this parameter is left empty, **NULL** is returned.
beg_index	Absolute offset (in bytes) for BLOB/BFILE objects, or offset (in characters) for CLOBs. The start position for matching is 1. **NOTE** The value ranges from 1 to *LOBMAXSIZE*. If a value out of the range is input, **NULL** is returned.
occur_index	Number of pattern matching times. The minimum value is 1. **NOTE** If the value is greater than the maximum number of times that the pattern string can be matched in the LOB, **0** is returned. If the value is out of the range from 1 to *LOBMAXSIZE*, **NULL** is returned.

- DBE_LOB.COMPARE

  Compares some or all LOBs or BFILEs.

  - If the two parts are equal, **0** is returned. Otherwise, a non-zero value is returned.
  - If the first LOB is smaller than the second, **-1** is returned. If the first LOB is larger than the second, **1** is returned.
  - If any of the *len*, *start1*, and *start2* parameters is invalid, **NULL** is returned. The valid offset range is 1 to *LOBMAXSIZE*.
  - If both the values of **start_pos1** and **start_pos2** exceed the LOB/BFILE length, **0** is returned.

  The prototype of DBE_LOB.COMPARE is as follows:

```
DBE_LOB.COMPARE (
   lob1       IN BLOB,
   lob2       IN BLOB,
   len        IN BIGINT DEFAULT 1073741312,
   start_pos1 IN BIGINT DEFAULT 1,
   start_pos2 IN BIGINT DEFAULT 1)
RETURN INTEGER;

DBE_LOB.COMPARE (
   lob1       IN CLOB,
   lob2       IN CLOB,
   len        IN BIGINT DEFAULT 1073741312,
   start_pos1 IN BIGINT DEFAULT 1,
   start_pos2 IN BIGINT DEFAULT 1)
RETURN INTEGER;

DBE_LOB.COMPARE (
   file1      IN DBE_LOB.BFILE,
   file2      IN DBE_LOB.BFILE,
   len        IN BIGINT DEFAULT 1073741312,
   start_pos1 IN BIGINT DEFAULT 1,
   start_pos2 IN BIGINT DEFAULT 1)
RETURN INTEGER;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-205 DBE_LOB.COMPARE API parameters

Parameter	Description
lob1/file1	First BLOB/CLOB/BFILE to be compared (which must be opened by DBE_LOB.BFILEOPEN first.)
lob2/file2	Second BLOB/CLOB/BFILE to be compared (which must be opened by DBE_LOB.BFILEOPEN first.)
len	Number of characters or bytes to be compared. The default value is **1073741312**.
start_pos1	Offset of the first LOB descriptor. The initial position is 1, and the maximum value is the maximum LOB length.
start_pos2	Offset of the second LOB descriptor. The initial position is 1, and the maximum value is the maximum LOB length.

- DBE_LOB.SUBSTR

  Reads a LOB substring and returns the read substring.

  The prototype of DBE_LOB.SUBSTR is as follows:

```
DBE_LOB.SUBSTR(
   lob_loc IN BLOB,
   amount  IN INTEGER DEFAULT 32767,
   off_set IN INTEGER  DEFAULT 1)
RETURN RAW;

DBE_LOB.SUBSTR(
   lob_loc IN CLOB,
   amount  IN INTEGER DEFAULT 32767,
   off_set IN INTEGER  DEFAULT 1)
RETURN VARCHAR2;
```

Table 2-206 DBE_LOB.SUBSTR API parameters

Parameter	Description
lob_loc	LOB descriptor whose substring is to be read. For BLOBs, the return value is of the RAW type. For CLOBs, the return value is of the VARCHAR2 type.
amount	Number of bytes or characters to be read. **NOTE** The value ranges from 1 to 32767. If the value exceeds the range, **NULL** is returned.
off_set	Number of characters or bytes offset from the start position. **NOTE** The value ranges from 1 to *LOBMAXSIZE*. If the value exceeds the range, **NULL** is returned.

- DBE_LOB.LOB_SUBSTR

  Reads a LOB or BFILE substring and returns the read substring. The LOB or BFILE can be up to 32 TB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The prototype of DBE_LOB.LOB_SUBSTR is as follows:

```
DBE_LOB.LOB_SUBSTR(
   lob_loc IN BLOB,
   amount  IN INTEGER DEFAULT 32767,
   off_set IN BIGINT  DEFAULT 1)
RETURN RAW;

DBE_LOB.LOB_SUBSTR(
   lob_loc IN CLOB,
   amount  IN INTEGER DEFAULT 32767,
   off_set IN BIGINT  DEFAULT 1)
RETURN VARCHAR2;

DBE_LOB.LOB_SUBSTR(
   bfile   IN DBE_LOB.BFILE,
   amount  IN INTEGER DEFAULT 32767,
   off_set IN BIGINT  DEFAULT 1)
RETURN RAW;
```

Table 2-207 DBE_LOB.LOB_SUBSTR API parameters

Parameter	Description
lob_loc/ bfile	LOB descriptor or BFILE whose substring is to be read. The file must be opened by DBE_LOB.BFILEOPEN first. For BLOBs/ BFILE files, the return value is of the RAW type. For CLOBs, the return value is of the VARCHAR2 type.
amount	Number of bytes or characters to be read. **NOTE** The value ranges from 1 to 32767. If the value exceeds the range, **NULL** is returned.
off_set	Number of characters or bytes offset from the start position. **NOTE** The value ranges from 1 to *LOBMAXSIZE*. If the value exceeds the range, **NULL** is returned.

- DBE_LOB.STRIP

  Truncates the LOB of a specified length. After STRIP is executed, the LOB length is set to the length specified by **newlen**.

  The prototype of DBE_LOB.STRIP is as follows:

```
DBE_LOB.STRIP(
   lob_loc INOUT BLOB,
   newlen  IN INTEGER);

DBE_LOB.STRIP(
   lob_loc INOUT CLOB,
   newlen  IN INTEGER);
```

Table 2-208 DBE_LOB.STRIP API parameters

Parameter	Description
lob_loc	LOB to read as the IN parameter, or truncated object as the OUT parameter. If this parameter is left empty, an error is reported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
newlen	New length after truncation, in bytes for BLOBs or in characters for CLOBs.

- DBE_LOB.LOB_STRIP

  Truncates the LOB of a specified length. After LOB_STRIP is executed, the LOB length is set to the length specified by **newlen**. The LOB can be greater than 1 GB, up to 32 TB.

  The prototype of DBE_LOB.LOB_STRIP is as follows:

```
DBE_LOB.LOB_STRIP(
   lob_loc INOUT BLOB,
   newlen  IN    BIGINT);

DBE_LOB.LOB_STRIP(
   lob_loc INOUT CLOB,
   newlen  IN    BIGINT);
```

  Table 2-209 DBE_LOB.LOB_STRIP API parameters

Parameter	Description
lob_loc	LOB to read as the IN parameter, or truncated object as the OUT parameter. If this parameter is left empty, an error is reported.
newlen	New length after truncation, in bytes for BLOBs or in characters for CLOBs. **NOTE** If the value is less than 1, **NULL** is returned. If the value is greater than the LOB length, an error is reported.

- DBE_LOB.CREATE_TEMPORARY

  Creates a temporary BLOB or CLOB. CREATE_TEMPORARY is used only for syntax compatibility.

  The prototype of DBE_LOB.CREATE_TEMPORARY is as follows:

```
DBE_LOB.CREATE_TEMPORARY (
   lob_loc INOUT BLOB,
   cache   IN    BOOLEAN,
   dur     IN    INTEGER DEFAULT 10);

DBE_LOB.CREATE_TEMPORARY (
   lob_loc INOUT CLOB,
   cache   IN    BOOLEAN,
   dur     IN    INTEGER DEFAULT 10);
```

  Table 2-210 DBE_LOB.CREATE_TEMPORARY API parameters

Parameter	Description
lob_loc	LOB descriptor.
cache	Used only for syntax compatibility.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
dur	Used only for syntax compatibility.

- DBE_LOB.APPEND

  Appends **source_obj** to the end of the target LOB.

  The prototype of DBE_LOB.APPEND is as follows:

```
DBE_LOB.APPEND (
   blob_obj INOUT BLOB,
   source_obj  IN    BLOB);

DBE_LOB.APPEND (
   clob_obj INOUT CLOB,
   source_obj  IN    CLOB);
```

  Table 2-211 DBE_LOB.APPEND API parameters

Parameter	Description
blob_obj/ clob_obj	BLOB/CLOB to be written.
source_obj	BLOB/CLOB to be read.

- DBE_LOB.LOB_APPEND

  Appends **source_obj** to the end of the target LOB.

  The prototype of DBE_LOB.LOB_APPEND is as follows:

```
DBE_LOB.LOB_APPEND(
   blob_obj   INOUT BLOB,
   source_obj IN    BLOB);

DBE_LOB.LOB_APPEND(
   clob_obj   INOUT CLOB,
   source_obj IN    CLOB);
```

  Table 2-212 DBE_LOB.LOB_APPEND API parameters

Parameter	Description
blob_obj/ clob_obj	BLOB/CLOB to be written.
source_obj	BLOB/CLOB to be read.

- DBE_LOB.FREETEMPORARY

  Frees LOBs created by CREATE_TEMPORARY.

  The prototype of DBE_LOB.FREETEMPORARY is as follows:

```
DBE_LOB.FREETEMPORARY (
   blob INOUT BLOB);

DBE_LOB.FREETEMPORARY (
   clob INOUT CLOB);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-213 DBE_LOB.FREETEMPORARY API parameters

Parameter	Description
blob/clob	BLOB/CLOB to be released.

- DBE_LOB.FILEOPEN

  Opens an external database BFILE and returns its file descriptor. A maximum of 10 BFILEs can be opened in a session.

  The BFILE type is defined as follows:

```
DBE_LOB.BFILE (
   directory TEXT,
   filename  TEXT,
   fd        INTEGER);
```

  The prototype of DBE_LOB.FILEOPEN is as follows:

```
DBE_LOB.FILEOPEN (
   bfile     IN DBE_LOB.BFILE,
   open_mode IN TEXT)
RETURN INTEGER;
```

Table 2-214 DBE_LOB.FILEOPEN API parameters

Parameter	Description
bfile	External database file to be opened. For a BFILE, it specifies the file path, file name, and file descriptor. **NOTE** The variable contains the file directory and the file name. • Location of the file directory, which needs to be added to system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. • When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to read and write files in the file path specified by **safe_data_path**. • File name with an extension (file type), excluding the path name. The path contained in the file name will be ignored by OPEN. In Unix, the file name cannot end with the combination of a slash and a dot (/.).
open_mode	File open mode, which can only be **r** (that is, read). An error is reported in other modes.

- DBE_LOB.FILECLOSE

  Closes an external BFILE.

  The prototype of DBE_LOB.FILECLOSE is as follows:

```
DBE_LOB.FILECLOSE (
   file IN INTEGER);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-215 DBE_LOB.FILECLOSE API parameters

Parameter	Description
file	External file to be closed (the file descriptor is returned by FILEOPEN).

- DBE_LOB.BFILEOPEN

  Opens an external BFILE. A maximum of 10 BFILE files can be opened in a session.

  The prototype of DBE_LOB.BFILEOPEN is as follows:

```
DBE_LOB.BFILEOPEN (
    bfile     INOUT DBE_LOB.BFILE,
    open_mode IN    TEXT DEFAULT 'R');
```

Table 2-216 DBE_LOB.BFILEOPEN API parameters

Parameter	Description
bfile	Opened database BFILE file as the INOUT parameter. **NOTE** The variable contains the file directory and the file name. • Location of the file directory, which needs to be added to system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. • When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to read and write files in the file path specified by **safe_data_path**. • File name with an extension (file type), excluding the path name. The path contained in the file name will be ignored by OPEN. In Unix, the file name cannot end with the combination of a slash and a dot (/.).
open_mode	File open mode, which can only be **r** (that is, read). An error is reported in other modes.

  Example:

```
-- Obtain the substring of the BFILE (the file content is ABCD).
DECLARE
bfile dbe_lob.bfile;
BEGIN
bfile = DBE_LOB.BFILENAME(dir_name, file_name); -- Obtain the corresponding BFILE.
DBE_LOB.bfileopen(bfile, 'r'); -- Open the BFILE.
RAISE NOTICE 'res:%', DBE_LOB.lob_substr(bfile, 10, 1); -- Obtain and print the substring.
DBE_LOB.bfileclose(bfile);-- Close the BFILE.
END;
/
NOTICE:  res:41
ANONYMOUS BLOCK EXECUTE
```

- DBE_LOB.BFILECLOSE

  Closes an external BFILE.

  The prototype of DBE_LOB.BFILECLOSE is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_LOB.BFILECLOSE (
   bfile INOUT DBE_LOB.BFILE);
```

Table 2-217 DBE_LOB.BFILECLOSE API parameters

Parameter	Description
bfile	Closed BFILE file as the INOUT parameter.

- DBE_LOB.LOADFROMFILE

  Loads an external BFILE to a BLOB and returns the object of the RAW type.

  The prototype of DBE_LOB.LOADFROMFILE is as follows:

```
DBE_LOB.LOADFROMFILE (
   dest_lob    IN BLOB,
   src_file    IN INTEGER,
   amount      IN INTEGER,
   dest_offset IN INTEGER,
   src_offset  IN INTEGER)
RETURN RAW;
```

Table 2-218 DBE_LOB.LOADFROMFILE API parameters

Parameter	Description
dest_lob	Target BLOB. The BFILE will be loaded to the specified offset position of the BLOB.
src_bfile	Source BFILE file to be read.
amount	Length of the content to be read from the BFILE file. **NOTE** If the length is less than 1 or greater than 32767, an error is reported.
dest_offset	Offset length of the BLOB. **NOTE** If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.
src_offset	Offset length of the BFILE file. **NOTE** • If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported. • If the sum of *amount* and *src_offset* is greater than the length of *src_bfile* plus 1, an error is reported.

- DBE_LOB.LOADFROMBFILE

  Loads an external BFILE to a LOB.

  The prototype of DBE_LOB.LOADFROMBFILE is as follows:

```
DBE_LOB.LOADFROMBFILE (
   dest_lob    INOUT BLOB,
   src_file    IN    DBE_LOB.BFILE,
   amount      IN    BIGINT,
   dest_offset IN    BIGINT DEFAULT 1,
   src_offset  IN    BIGINT DEFAULT 1)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
RETURN BLOB;

DBE_LOB.LOADFROMBFILE (
   dest_lob    INOUT CLOB,
   src_file    IN    DBE_LOB.BFILE,
   amount      IN    BIGINT,
   dest_offset IN    BIGINT DEFAULT 1,
   src_offset  IN    BIGINT DEFAULT 1)
RETURN CLOB;
```

Table 2-219 DBE_LOB.LOADFROMBFILE API parameters

Parameter	Description
dest_lob	Target LOB as the INOUT parameter, to which the BFILE will be loaded. The file must be opened by DBE_LOB.BFILEOPEN first. The LOB can be greater than 1 GB, up to 32 TB.
src_file	Source BFILE file to be read. The BFILE file can be greater than 1 GB, up to 32 TB.
amount	Length of the content to be read from the BFILE and to be written to the LOB. **NOTE** If the length is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.
dest_offset	Offset length of the LOB. **NOTE** If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.
src_offset	Offset length of the BFILE file. **NOTE** • If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported. • If the sum of *amount* and *src_offset* is greater than the length of *src_bfile* plus 1, an error is reported.

- DBE_LOB.LOADBLOBFROMFILE

  Loads an external BFILE to a BLOB and returns the object of the RAW type.

  The prototype of DBE_LOB.LOADBLOBFROMFILE is as follows:

```
DBE_LOB.LOADBLOBFROMFILE (
   dest_lob    IN BLOB,
   src_file    IN INTEGER,
   amount      IN INTEGER,
   dest_offset IN INTEGER,
   src_offset  IN INTEGER)
RETURN RAW;
```

Table 2-220 DBE_LOB.LOADBLOBFROMFILE API parameters

Parameter	Description
dest_lob	Target BLOB, to which the BFILE will be loaded.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
src_file	Source BFILE file to be read.
amount	Length of the target BLOB. If the length of a file exceeds this threshold, the file will not be saved to the BLOB. **NOTE** If the length is less than 1 or greater than 32767, an error is reported.
dest_offset	Offset length of the BLOB. **NOTE** If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.
src_offset	Offset length of the BFILE file. **NOTE** • If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported. • If the sum of *amount* and *src_offset* is greater than the length of *src_bfile* plus 1, an error is reported.

- DBE_LOB.LOADBLOBFROMBFILE

  Loads an external BFILE to a BLOB.

  The prototype of DBE_LOB.LOADBLOBFROMBFILE is as follows:

```
DBE_LOB.LOADBLOBFROMFILE (
   dest_lob    INOUT BLOB,
   src_file    IN    DBE_LOB.BFILE,
   amount      IN    BIGINT,
   dest_offset INOUT BIGINT,
   src_offset  INOUT BIGINT)
```

Table 2-221 DBE_LOB.LOADBLOBFROMBFILE API parameters

Parameter	Description
dest_lob	Target BLOB as the INOUT parameter, to which the BFILE (which must be opened by DBE_LOB.BFILEOPEN first) is loaded. The BFILE file can be greater than 1 GB, up to 32 TB.
src_file	Source BFILE file to be read. The BFILE file can be greater than 1 GB, up to 32 TB.
amount	Length of the target BLOB. If the length of a file exceeds this threshold, the file will not be saved to the BLOB. **NOTE** If the length is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.
dest_offset	Offset length of the BLOB. **NOTE** If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
src_offset	Offset length of the BFILE file. **NOTE** • If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported. • If the sum of *amount* and *src_offset* is greater than the length of *src_bfile* plus 1, an error is reported.

- DBE_LOB.LOADCLOBFROMFILE

  Loads an external BFILE to a CLOB and returns the object of the RAW type.

  The prototype of DBE_LOB.LOADCLOBFROMFILE is as follows:

```
DBE_LOB.LOADCLOBFROMFILE (
   dest_lob    IN CLOB,
   src_file    IN INTEGER,
   amount      IN INTEGER,
   dest_offset IN INTEGER,
   src_offset  IN INTEGER)
RETURN RAW;
```

  Table 2-222 DBE_LOB.LOADCLOBFROMFILE API parameters

Parameter	Description
dest_lob	Target CLOB, to which the BFILE will be loaded.
src_file	Source BFILE file to be read.
amount	Length of the CLOB. **NOTE** If the length is less than 1 or greater than 32767, an error is reported.
dest_offset	Offset length of the CLOB. **NOTE** If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.
src_offset	Offset length of the BFILE file. **NOTE** • If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported. • If the sum of *amount* and *src_offset* is greater than the length of *src_bfile* plus 1, an error is reported.

- DBE_LOB.LOADCLOBFROMBFILE

  Loads an external BFILE to a CLOB.

  The prototype of DBE_LOB.LOADCLOBFROMBFILE is as follows:

```
DBE_LOB.LOADCLOBFROMBFILE (
   dest_lob    INOUT CLOB,
   src_file    IN    DBE_LOB.BFILE,
   amount      IN    BIGINT,
   dest_offset INOUT BIGINT,
   src_offset  INOUT BIGINT)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-223 DBE_LOB.LOADCLOBFROMBFILE API parameters

Parameter	Description
dest_lob	Target CLOB as the INOUT parameter, to which the BFILE (which must be opened by DBE_LOB.BFILEOPEN first) is loaded. The BFILE file can be greater than 1 GB, up to 32 TB.
src_file	Source BFILE file to be read. The BFILE file can be greater than 1 GB, up to 32 TB.
amount	Length of the target CLOB. If the length of a file exceeds this threshold, the file will not be saved to the CLOB. **NOTE** If the length is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.
dest_offset	Offset length of the CLOB. **NOTE** If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.
src_offset	Offset length of the BFILE file. **NOTE** • If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported. • If the sum of *amount* and *src_offset* is greater than the length of *src_bfile* plus 1, an error is reported.

- DBE_LOB.CONVERTTOBLOB

  Converts a CLOB to a BLOB. The CLOB cannot be greater than 1 GB.

  The prototype of DBE_LOB.CONVERTTOBLOB is as follows:

```
DBE_LOB.CONVERTTOBLOB(
   dest_blob   IN BLOB,
   src_clob    IN CLOB,
   amount      IN INTEGER DEFAULT 32767,
   dest_offset IN INTEGER DEFAULT 1,
   src_offset  IN INTEGER DEFAULT 1)
RETURN RAW;
```

Table 2-224 DBE_LOB.CONVERTTOBLOB API parameters

Parameter	Description
dest_blob	Target BLOB, which is converted from a CLOB.
src_clob	Source CLOB to be read.
amount	Length of the target CLOB. If the length of a file exceeds this threshold, the file will not be saved to the BLOB. If the length is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
dest_offset	Offset length of the BLOB. If **dest_offset** is set to **1**, data is loaded from the start position of the file. The rest may be deduced by analogy.
src_offset	Offset length of the CLOB. If **src_offset** is set to **1**, data is read from the start position of the file. The rest may be deduced by analogy.

- DBE_LOB.LOB_CONVERTTOBLOB

  Converts a CLOB to a BLOB. The size of a LOB can be greater than 1 GB.

  The prototype of DBE_LOB.LOB_CONVERTTOBLOB is as follows:

```
DBE_LOB.LOB_CONVERTTOBLOB(
   dest_blob   INOUT BLOB,
   src_clob    IN    CLOB,
   amount      IN    BIGINT,
   dest_offset INOUT BIGINT,
   src_offset  INOUT BIGINT)
```

  Table 2-225 DBE_LOB.LOB_CONVERTTOBLOB API parameters

Parameter	Description
dest_blob	Target BLOB, which is converted from a CLOB.
src_clob	Source CLOB to be read.
amount	Length of the target CLOB. If the length of a file exceeds this threshold, the file will not be saved to the BLOB. If the length is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.
dest_offset	Offset length of the BLOB. If **dest_offset** is set to **1**, data is loaded from the start position of the file. The rest may be deduced by analogy. If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.
src_offset	Offset length of the CLOB. If **src_offset** is set to **1**, data is read from the start position of the file. The rest may be deduced by analogy. If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.

- DBE_LOB.CONVERTTOCLOB

  Converts a BLOB to a CLOB. The BLOB cannot be greater than 1 GB.

  The prototype of DBE_LOB.CONVERTTOCLOB is as follows:

```
DBE_LOB.CONVERTTOCLOB(
   dest_clob   IN CLOB,
   src_blob    IN BLOB,
   amount      IN INTEGER DEFAULT 32767,
   dest_offset IN INTEGER DEFAULT 1,
   src_offset  IN INTEGER DEFAULT 1)
RETURN text;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-226 DBE_LOB.CONVERTTOCLOB API parameters

Parameter	Description
dest_clob	Target CLOB, which is converted from a BLOB.
src_blob	Source BLOB to be read.
amount	Length of the target BLOB. If the length of a file exceeds this threshold, the file will not be saved to the CLOB.
dest_offset	Offset length of the CLOB. If **dest_offset** is set to **1**, data is loaded from the start position of the file. The rest may be deduced by analogy.
src_offset	Offset length of the BLOB. If **src_offset** is set to **1**, data is read from the start position of the file. The rest may be deduced by analogy.

- DBE_LOB.LOB_CONVERTTOCLOB

  Converts a BLOB to a CLOB. The size of a LOB can be greater than 1 GB.

  The prototype of DBE_LOB.LOB_CONVERTTOCLOB is as follows:

```
DBE_LOB.LOB_CONVERTTOCLOB(
   dest_clob    INOUT CLOB,
   src_blob    IN    BLOB,
   amount      IN    BIGINT,
   dest_offset INOUT BIGINT,
   src_offset  INOUT BIGINT)
```

Table 2-227 DBE_LOB.LOB_CONVERTTOCLOB API parameters

Parameter	Description
dest_clob	Target CLOB, which is converted from a BLOB.
src_blob	Source BLOB to be read.
amount	Length of the target BLOB. If the length of a file exceeds this threshold, the file will not be saved to the CLOB.
dest_offset	Offset length of the CLOB. If **dest_offset** is set to **1**, data is loaded from the start position of the file. The rest may be deduced by analogy. If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.
src_offset	Offset length of the BLOB. If **src_offset** is set to **1**, data is read from the start position of the file. The rest may be deduced by analogy. If the offset is less than 1 or greater than the value of *LOBMAXSIZE*, an error is reported.

- DBE_LOB.GETCHUNKSIZE

  Returns *TOAST_MAX_CHUNK_SIZE*. When LOB is stored in the database, TOAST is used internally.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The prototype of DBE_LOB.GETCHUNKSIZE is as follows:

```
DBE_LOB.GETCHUNKSIZE(
   lob_loc IN CLOB
)RETURN INTEGER

DBE_LOB.GETCHUNKSIZE(
   lob_loc IN BLOB
)RETURN INTEGER
```

Table 2-228 DBE_LOB.GETCHUNKSIZE API parameters

Parameter	Description
lob_loc	Target CLOB/BLOB.

- DBE_LOB.LOB_WRITE

  Reads the specified length of the source object from the start position, writes the content to the specified offset position of the target LOB, overrides the original content, and returns the target LOB.

  The prototype of DBE_LOB.LOB_WRITE is as follows:

```
DBE_LOB.LOB_WRITE(
   clob_obj INOUT CLOB,
   amount   IN    INTEGER,
   off_set  IN    BIGINT,
   source   IN    VARCHAR2
)
RETURN CLOB;

DBE_LOB.LOB_WRITE(
   blob_obj INOUT BLOB,
   amount   IN    INTEGER,
   off_set  IN    BIGINT,
   source   IN    RAW
)
RETURN BLOB;
```

Table 2-229 DBE_LOB.LOB_WRITE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
blob_obj / clob_obj	BLOB/ CLOB	INOUT	No	Target LOB as the INOUT parameter, to which the content is to be written.
amount	INTEGER	IN	No	Length of the data to be written (in bytes for BLOBs or in characters for CLOBs).
off_set	BIGINT	IN	No	Offset position for writing data to *blob_obj/clob_obj*

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
source	RAW/VARCHAR2	IN	No	Source object

- DBE_LOB.BFILENAME

  Constructs a BFILE based on the directory and file name.

  The prototype of DBE_LOB.BFILENAME is as follows:

```
DBE_LOB.BFILENAME(
    directory IN TEXT,
    filename  IN TEXT)
RETURN DBE_LOB.BFILE;
```

Table 2-230 DBE_LOB.BFILENAME API parameters

Parameter	Description
directory	File directory. **NOTE** Location of the file directory, which needs to be added to system catalog **PG_DIRECTORY**. If the input path does not match the path in **PG_DIRECTORY**, an error indicating that the path does not exist will be reported. • When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to read and write files in the file path specified by **safe_data_path**. • File name with an extension (file type), excluding the path name. The path contained in the file name will be ignored by OPEN. In Unix, the file name cannot end with the combination of a slash and a dot (/.).
filename	File name.

Examples

```
-- Obtain the length of a string.
SELECT DBE_LOB.GET_LENGTH('12345678');
 get_length
------------
          8
(1 row)

-- DBE_LOB.READ API.
DECLARE
myraw  RAW(100);
amount INTEGER :=2;
buffer INTEGER :=1;
begin
DBE_LOB.READ('123456789012345',amount,buffer,myraw);
dbe_output.print_line(myraw);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
end;
/
0123
ANONYMOUS BLOCK EXECUTE

CREATE TABLE blob_Table (t1 blob) DISTRIBUTE BY REPLICATION;
CREATE TABLE blob_Table_bak (t2 blob) DISTRIBUTE BY REPLICATION;
INSERT INTO blob_Table VALUES('abcdef');
INSERT INTO blob_Table_bak VALUES('22222');

-- Multiple DBE_LOB APIs.
DECLARE
str varchar2(100) := 'abcdef';
source raw(100);
dest blob;
copyto blob;
amount int;
PSV_SQL varchar2(100);
PSV_SQL1 varchar2(100);
a int :=1;
len int;
BEGIN
source := dbe_raw.cast_from_varchar2_to_raw(str);
amount := dbe_raw.get_length(source);

PSV_SQL :='select * from blob_Table for update';
PSV_SQL1 := 'select * from blob_Table_bak for update';

EXECUTE IMMEDIATE PSV_SQL into dest;
EXECUTE IMMEDIATE PSV_SQL1 into copyto;

DBE_LOB.WRITE(dest, amount, 1, source);
DBE_LOB.WRITE_APPEND(dest, amount, source);

DBE_LOB.ERASE(dest, a, 1);
DBE_OUTPUT.PRINT_LINE(a);
DBE_LOB.COPY(copyto, dest, amount, 10, 1);
perform DBE_LOB.CLOSE(dest);
RETURN;
END;
/
1
ANONYMOUS BLOCK EXECUTE

-- Drop the table.
DROP TABLE blob_Table;
DROP TABLE blob_Table_bak;
```

2.12.2.10 DBE_MATCH

APIs

Table 2-231 lists APIs supported by the DBE_MATCH package.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-231 DBE_MATCH

API	Description
DBE_MATCH.EDIT_DISTANCE_SIMILARITY	Compares the difference between two character strings (minimum steps of deletion, addition, and conversion) and normalizes the difference to a value ranging from 0 to 100. The value **100** indicates that the two character strings are the same, and the value **0** indicates that the two character strings are different.

- DBE_MATCH.EDIT_DISTANCE_SIMILARITY

  Compares the difference between two character strings (minimum steps of deletion, addition, and conversion) and normalizes the difference to a value ranging from 0 to 100. The value **100** indicates that the two character strings are the same, and the value **0** indicates that the two character strings are different.

  The prototype of DBE_MATCH.EDIT_DISTANCE_SIMILARITY is as follows:

```
DBE_MATCH.EDIT_DISTANCE_SIMILARITY(
    str1 IN text,
    str2 IN text
)returns integer ;
```

Table 2-232 DBE_MATCH.EDIT_DISTANCE_SIMILARITY API parameters

Parameter	Description
str1	First character string. If the value is **null**, **0** is returned.
str2	Second character string. If the value is **null**, **0** is returned.

2.12.2.11 DBE_OBFUSCATION_TOOLKIT

APIs

NOTICE

The DES, DES3, and MD5 hash algorithms have security risks. Exercise caution when using DBE_OBFUSCATION_TOOLKIT for encryption and decryption.

DBE_OBFUSCATION_TOOLKIT is a compatibility function package provided by GaussDB to support APIs of the DES, DES3, and MD5 hash algorithms, ensuring compatibility with the original service logic. **Table 2-233** lists APIs supported by DBE_OBFUSCATION_TOOLKIT.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-233 DBE_OBFUSCATION_TOOLKIT

API	Description
DBE_OBFUSCATION_TOOLKIT.DESGETKEY	Generates a key using the DES algorithm.
DBE_OBFUSCATION_TOOLKIT.DESENCRYPT	Encrypts data using the DES algorithm.
DBE_OBFUSCATION_TOOLKIT.DESDECRYPT	Decrypts data using the DES algorithm.
DBE_OBFUSCATION_TOOLKIT.DES3GETKEY	Generates a key using the DES3 algorithm.
DBE_OBFUSCATION_TOOLKIT.DES3ENCRYPT	Encrypts data using the DES3 algorithm.
DBE_OBFUSCATION_TOOLKIT.DES3DECRYPT	Decrypts data using the DES3 algorithm.
DBE_OBFUSCATION_TOOLKIT.MD5	Calculates the MD5 hash value of data.

- DBE_OBFUSCATION_TOOLKIT.DESGETKEY

  Description: Generates a key using the DES algorithm.

  DBE_OBFUSCATION_TOOLKIT.DESGETKEY has two prototypes:

```
-- Prototype 1: The input parameter is of the VARCHAR2 type.
DBE_OBFUSCATION_TOOLKIT.DESGETKEY(
seed_string IN VARCHAR2
) RETURN VARCHAR2;

-- Prototype 2: The input parameter is of the RAW type.
DBE_OBFUSCATION_TOOLKIT.DESGETKEY(
seed IN RAW
) RETURN RAW;
```

  Table 2-234 DBE_OBFUSCATION_TOOLKIT.DESGETKEY API parameters

Parameter	Type	Can Be Null	Description
seed_string	VARCHAR2	No	Seed value used to generate a key. The value must contain at least 80 bytes.
seed	RAW	No	

  Example:

```
-- Create a database in ORA-compatible mode.
gaussdb=# CREATE DATABASE ora_db dbcompatibility = 'ORA' encoding = 'utf8';
gaussdb=# \c ora_db
-- Generate a random key.
ora_db=# SELECT
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
rawtohex(dbe_raw.cast_from_varchar2_to_raw(dbe_obfuscation_toolkit.desgetkey(seed_string =>
'0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000')));
WARNING:  The DES algorithm is an insecure algorithm, please use it with caution.
CONTEXT:  referenced column: rawtohex
    rawtohex
------------------
 A7F0B24CF66DAECF
(1 row)

ora_db=# SELECT rawtohex(dbe_obfuscation_toolkit.desgetkey(seed =>
dbe_raw.cast_from_varchar2_to_raw('00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000')));
WARNING:  The DES algorithm is an insecure algorithm, please use it with caution.
CONTEXT:  referenced column: rawtohex
    rawtohex
------------------
 2A5984CA3AB67344
(1 row)
```

NOTE

The key generated by DBE_OBFUSCATION_TOOLKIT.DESGETKEY is a random key.

- DBE_OBFUSCATION_TOOLKIT.DESENCRYPT

  Description: Encrypts data using the DES algorithm.

  DBE_OBFUSCATION_TOOLKIT.DESENCRYPT has two prototypes:

```
-- Prototype 1: The input parameter is of the VARCHAR2 type.
DBE_OBFUSCATION_TOOLKIT.DESENCRYPT(
input_string IN VARCHAR2,
key_string   IN VARCHAR2
) RETURN VARCHAR2;

-- Prototype 2: The input parameter is of the RAW type.
DBE_OBFUSCATION_TOOLKIT.DESENCRYPT(
input IN RAW,
key   IN RAW
) RETURN RAW;
```

Table 2-235 DBE_OBFUSCATION_TOOLKIT.DESENCRYPT API parameters

Parameter	Type	Can Be Null	Description
input_string	VARCHAR2	No	Data to be encrypted. The length must be a multiple of 8 bytes.
input	RAW	No	
key_string	VARCHAR2	No	Encryption key. The length must be at least 8 bytes. If the length exceeds 8 bytes, the encryption result is not affected.
key	RAW	No	

Example:

```
ora_db=# SELECT
rawtohex(dbe_raw.cast_from_varchar2_to_raw(dbe_obfuscation_toolkit.desencrypt(input_string =>
'12345678', key_string => '12345678')));
WARNING:  The DES algorithm is an insecure algorithm, please use it with caution.
CONTEXT:  referenced column: rawtohex
    rawtohex
------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 96D0028878D58C89
(1 row)

ora_db=# SELECT rawtohex(dbe_obfuscation_toolkit.desencrypt(input =>
dbe_raw.cast_from_varchar2_to_raw('12345678'), key =>
dbe_raw.cast_from_varchar2_to_raw('12345678')));
WARNING:  The DES algorithm is an insecure algorithm, please use it with caution.
CONTEXT:  referenced column: rawtohex
    rawtohex
------------------
 96D0028878D58C89
(1 row)
```

NOTE

A password is required for executing this function. For security purposes, the gsql tool does not record the SQL statements containing this function in the execution history. That is, the execution history of this function cannot be found with gsql by paging up and down.

- DBE_OBFUSCATION_TOOLKIT.DESDECRYPT

  Description: Decrypts data using the DES algorithm.

  DBE_OBFUSCATION_TOOLKIT.DESDECRYPT has two prototypes:

```
-- Prototype 1: The input parameter is of the VARCHAR2 type.
DBE_OBFUSCATION_TOOLKIT.DESDECRYPT(
input_string IN VARCHAR2,
key_string   IN VARCHAR2
) RETURN VARCHAR2;

-- Prototype 2: The input parameter is of the RAW type.
DBE_OBFUSCATION_TOOLKIT.DESDECRYPT(
input IN RAW,
key   IN RAW
) RETURN RAW;
```

Table 2-236 DBE_OBFUSCATION_TOOLKIT.DESDECRYPT API parameters

Parameter	Type	Can Be Null	Description
input_string	VARCHAR2	No	Data to be decrypted. The length of the data must be a multiple of 8 bytes.
input	RAW	No	
key_string	VARCHAR2	No	Decryption key. The length of the key must be at least 8 bytes. If the length exceeds 8 bytes, the decryption result is not affected.
key	RAW	No	

Example:

```
ora_db=# SELECT
rawtohex(dbe_raw.cast_from_varchar2_to_raw(dbe_obfuscation_toolkit.desdecrypt(input_string =>
'12345678', key_string => '12345678')));
WARNING:  The DES algorithm is an insecure algorithm, please use it with caution.
CONTEXT:  referenced column: rawtohex
    rawtohex
------------------
 F5E8E9EB81F28B73
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ora_db=# SELECT rawtohex(dbe_obfuscation_toolkit.desdecrypt(input =>
dbe_raw.cast_from_varchar2_to_raw('12345678'), key =>
dbe_raw.cast_from_varchar2_to_raw('12345678')));
WARNING:  The DES algorithm is an insecure algorithm, please use it with caution.
CONTEXT:  referenced column: rawtohex
    rawtohex
------------------
 F5E8E9EB81F28B73
(1 row)
```

NOTE

A password is required for executing this function. For security purposes, the gsql tool does not record the SQL statements containing this function in the execution history. That is, the execution history of this function cannot be found with gsql by paging up and down.

- DBE_OBFUSCATION_TOOLKIT.DES3GETKEY

  Description: Generates a key using the DES3 algorithm.

  DBE_OBFUSCATION_TOOLKIT.DES3GETKEY has two prototypes:

```
-- Prototype 1: The input parameter is of the VARCHAR2 type.
DBE_OBFUSCATION_TOOLKIT.DES3GETKEY(
which       IN INTEGER DEFAULT 0,
seed_string IN VARCHAR2
) RETURN VARCHAR2;

-- Prototype 2: The input parameter is of the RAW type.
DBE_OBFUSCATION_TOOLKIT.DES3GETKEY(
which IN INTEGER DEFAULT 0,
seed  IN RAW
) RETURN RAW;
```

Table 2-237 DBE_OBFUSCATION_TOOLKIT.DES3GETKEY API parameters

Parameter	Type	Can Be Null	Description
which	INTEGER	No	Key mode. Value range: **0** or **1**. • **0**: A 16-byte key (that is, two keys) is returned. • **1**: A 24-byte key (that is, three keys) is returned. Default value: **0**.
seed_string	VARCHAR2	No	Seed value used to generate a key. The value must contain at least 80 bytes.
seed	RAW	No	

Example:

```
-- Generate a random key.
ora_db=# SELECT
rawtohex(dbe_raw.cast_from_varchar2_to_raw(dbe_obfuscation_toolkit.des3getkey(seed_string =>
'00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000')));
WARNING:  The DES3 algorithm is an insecure algorithm, please use it with caution.
CONTEXT:  referenced column: rawtohex
            rawtohex
```

```
----------------------------------
 9A64E470BC16D4969E782F3F5A8491A6
(1 row)

ora_db=# SELECT rawtohex(dbe_obfuscation_toolkit.des3getkey(which => 1, seed =>
dbe_raw.cast_from_varchar2_to_raw('00000000000
```

Parameter	Type	Can Be Null	Description
which	INTEGER	No	Key mode. Value range: **0** or **1**. ● **0**: A 16-byte key (that is, two keys) is used for encryption. ● **1**: A 24-byte key (that is, three keys) is used for encryption. Default value: **0**.
iv_string	VARCHAR2	Yes	Initial vector. When this parameter is not set to **NULL**, the length of the initial vector must be a multiple of 8 bytes. If the length of the initialization vector exceeds 8 bytes, the excess part will not affect the encryption result. Default value: **NULL**.
iv	RAW	Yes	

Example:

```
ora_db=# SELECT
rawtohex(dbe_raw.cast_from_varchar2_to_raw(dbe_obfuscation_toolkit.des3encrypt(input_string =>
'12345678', key_string => '1234567823456789', iv_string => '12345678')));
WARNING:  The DES3 algorithm is an insecure algorithm, please use it with caution.
CONTEXT:  referenced column: rawtohex
    rawtohex
------------------
 EFDD1DFBCEE3B0CA
(1 row)

ora_db=# SELECT rawtohex(dbe_obfuscation_toolkit.des3encrypt(input =>
dbe_raw.cast_from_varchar2_to_raw('12345678'), key =>
dbe_raw.cast_from_varchar2_to_raw('123456781234567887654321'), which => 1));
WARNING:  The DES3 algorithm is an insecure algorithm, please use it with caution.
CONTEXT:  referenced column: rawtohex
    rawtohex
------------------
 1A1546D3DEAEB8BE
(1 row)
```

NOTE

A password is required for executing this function. For security purposes, the gsql tool does not record the SQL statements containing this function in the execution history. That is, the execution history of this function cannot be found with gsql by paging up and down.

- DBE_OBFUSCATION_TOOLKIT.DES3DECRYPT

  Description: Decrypts data using the DES3 algorithm.

  DBE_OBFUSCATION_TOOLKIT.DES3DECRYPT has two prototypes:

```
-- Prototype 1: The input parameter is of the VARCHAR2 type.
DBE_OBFUSCATION_TOOLKIT.DES3DECRYPT(
input_string IN VARCHAR2,
key_string   IN VARCHAR2,
which        IN INTEGER DEFAIULT 0,
iv_string    IN VARCHAR2 DEFAULT NULL
) RETURN VARCHAR2;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Prototype 2: The input parameter is of the RAW type.
DBE_OBFUSCATION_TOOLKIT.DES3DECRYPT(
input IN RAW,
key   IN RAW,
which IN INTEGER DEFAIULT 0,
iv    IN RAW DEFAULT NULL
) RETURN RAW;
```

Table 2-239 DBE_OBFUSCATION_TOOLKIT.DES3DECRYPT API parameters

Parameter	Type	Can Be Null	Description
input_string	VARCHAR2	No	Data to be decrypted. The length of the data must be a multiple of 8 bytes.
input	RAW	No	
key_string	VARCHAR2	No	Decryption key. The length of the key depends on the value of **which**. • If the value of **which** is **0**, the length of the decryption key must be at least 16 bytes. If the length exceeds 16 bytes, the decryption result is not affected. • If the value of **which** is **1**, the length of the decryption key must be at least 24 bytes. If the length exceeds 24 bytes, the decryption result is not affected.
key	RAW	No	
which	INTEGER	No	Key mode. Value range: **0** or **1**. • **0**: A 16-byte key (that is, two keys) is used for decryption. • **1**: A 24-byte key (that is, three keys) is used for decryption. Default value: **0**.
iv_string	VARCHAR2	Yes	Initial vector. When this parameter is not set to **NULL**, the length of the initial vector must be a multiple of 8 bytes. The excess part will not affect the decryption result. Default value: **NULL**.
iv	RAW	Yes	

Example:

```
ora_db=# SELECT
rawtohex(dbe_raw.cast_from_varchar2_to_raw(dbe_obfuscation_toolkit.des3decrypt(input_string =>
'12345678', key_string => '1234567823456789', iv_string => '12345678')));
WARNING:  The DES3 algorithm is an insecure algorithm, please use it with caution.
CONTEXT:  referenced column: rawtohex
    rawtohex
------------------
 8C2ED115CCA35582
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ora_db=# SELECT rawtohex(dbe_obfuscation_toolkit.des3decrypt(input =>
dbe_raw.cast_from_varchar2_to_raw('12345678'), key =>
dbe_raw.cast_from_varchar2_to_raw('123456781234567887654321 0'), which => 1, iv =>
dbe_raw.cast_from_varchar2_to_raw('12345678')));
WARNING:  The DES3 algorithm is an insecure algorithm, please use it with caution.
CONTEXT:  referenced column: rawtohex
    rawtohex
------------------
 A6FF1AA4A875B3E8
(1 row)
```

NOTE

A password is required for executing this function. For security purposes, the gsql tool does not record the SQL statements containing this function in the execution history. That is, the execution history of this function cannot be found with gsql by paging up and down.

- DBE_OBFUSCATION_TOOLKIT.MD5

  Description: Calculates the MD5 hash value of data.

  DBE_OBFUSCATION_TOOLKIT.MD5 has two prototypes:

```
-- Prototype 1: The input parameter is of the VARCHAR2 type.
DBE_OBFUSCATION_TOOLKIT.MD5(
input_string IN VARCHAR2
) RETURN VARCHAR2;

-- Prototype 2: The input parameter is of the RAW type.
DBE_OBFUSCATION_TOOLKIT.MD5(
input IN RAW
) RETURN RAW;
```

Table 2-240 DBE_OBFUSCATION_TOOLKIT.MD5 API parameters

Parameter	Type	Can Be Null	Description
input_string	VARCHAR2	No	Data to be hashed.
input	RAW	No	

Example:

```
ora_db=# SELECT
rawtohex(dbe_raw.cast_from_varchar2_to_raw(dbe_obfuscation_toolkit.md5(input_string =>
'1234567890')));
WARNING:  The MD5 algorithm is an insecure algorithm, please use it with caution.
CONTEXT:  referenced column: rawtohex
            rawtohex
----------------------------------
 E807F1FCF82D132F9BB018CA6738A19F
(1 row)

ora_db=# SELECT
rawtohex(dbe_obfuscation_toolkit.md5(dbe_raw.cast_from_varchar2_to_raw('1234567890')));
WARNING:  The MD5 algorithm is an insecure algorithm, please use it with caution.
CONTEXT:  referenced column: rawtohex
            rawtohex
----------------------------------
 E807F1FCF82D132F9BB018CA6738A19F
(1 row)

-- Drop the ora_db database.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ora_db=# \c postgres
gaussdb=# DROP DATABASE ora_db;
```

2.12.2.12 DBE_OUTPUT

NOTE

When DBE_OUTPUT.PUT_LINE is used to print the result obtained by the DBE_FILE.READ_LINE_NCHAR API, ensure that the UTF-8 character set encoding can be converted to the current database character set encoding. If the preceding conditions are met, the result can be properly output. DBE_OUTPUT.PRINT_LINE does not support this function.

APIs

Table 2-241 lists the APIs supported by the DBE_OUTPUT package.

Table 2-241 DBE_OUTPUT

API	Description
DBE_OUTPUT.PRINT_LINE	Outputs the specified text with newline characters.
DBE_OUTPUT.PRINT	Outputs the specified text without newline characters.
DBE_OUTPUT.SET_BUFFER_SIZE	Sets the size of the output buffer. If the size is not specified, the buffer can contain a maximum of 20000 bytes. If the size is set to a value less than or equal to 2000 bytes, the buffer can contain a maximum of 2000 bytes.
DBE_OUTPUT.DISABLE	Disables the calling of PUT, PUT_LINE, NEW_LINE, GET_LINE, and GET_LINES, and clears the output buffer.
DBE_OUTPUT.ENABLE	Enables the buffer, allows the calling of PUT, PUT_LINE, NEW_LINE, GET_LINE, and GET_LINES, and sets the buffer size.
DBE_OUTPUT.GET_LINE	Obtains a line of data from the buffer with a newline character as the boundary. The obtained data is not output to the client.
DBE_OUTPUT.GET_LINES	Obtains the character string of a specified number of lines in the buffer as a VARCHAR array. The obtained content is cleared from the buffer and is not output to the client.
DBE_OUTPUT.NEW_LINE	Places a line at the end of the buffer, places an end-of-line marker, and leaves a new line empty.
DBE_OUTPUT.PUT	Places an input string in the buffer without a newline character at the end. When the stored procedure ends, the line ending with the newline character is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_OUTPUT.PUT_LINE	Places an input string in the buffer with a newline character at the end. When the stored procedure ends, the line ending with the newline character is displayed.

- DBE_OUTPUT.PRINT_LINE

  Outputs the specified text and adds newline characters.

  The prototype of DBE_OUTPUT.PRINT_LINE is as follows:

  ```
  DBE_OUTPUT.PRINT_LINE (
  format IN VARCHAR2);
  ```

  Table 2-242 DBE_OUTPUT.PRINT_LINE API parameters

Parameter	Description
format	Output text.

- DBE_OUTPUT.PRINT

  Outputs the specified text without adding newline characters.

  The prototype of DBE_OUTPUT.PRINT is as follows:

  ```
  DBE_OUTPUT.PRINT (
  format IN VARCHAR2);
  ```

  Table 2-243 DBE_OUTPUT.PRINT API parameters

Parameter	Description
format	Output text.

- DBE_OUTPUT.SET_BUFFER_SIZE

  Sets the output buffer size. If the size is not specified, it contains a maximum of 20000 bytes.

  The prototype of DBE_OUTPUT.SET_BUFFER_SIZE is as follows:

  ```
  DBE_OUTPUT.SET_BUFFER_SIZE (
  size IN INTEGER default 20000);
  ```

  Table 2-244 DBE_OUTPUT.SET_BUFFER_SIZE API parameters

Parameter	Description
size	Sets the output buffer size. The value ranges from 2000 to 1000000 bytes. If the passed value is less than 2000, it is automatically adjusted to 2000. If the passed value is greater than 1000000, it is automatically adjusted to 1000000.

- DBE_OUTPUT.DISABLE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Disables the calling of PUT, PUT_LINE, NEW_LINE, GET_LINE, and GET_LINES, and clears the output buffer.

The prototype of DBE_OUTPUT.DISABLE is as follows:

```
DBE_OUTPUT.DISABLE;
```

- DBE_OUTPUT.ENABLE

  Enables the buffer, allows the calling of PUT, PUT_LINE, NEW_LINE, GET_LINE, and GET_LINES, and sets the buffer size. If the buffer size is not specified, 20,000 bytes are allowed by default.

  The prototype of DBE_OUTPUT.ENABLE is as follows:

  ```
  DBE_OUTPUT.ENABLE (
      buffer_size IN INTEGER DEFAULT 20000
  );
  ```

  Table 2-245 DBE_OUTPUT.ENABLE API parameters

Parameter	Description
buffer_size	Upper limit of the buffer size, in bytes. If **buffer_size** is set to **NULL**, the default value (20000 bytes) is used.

- DBE_OUTPUT.GET_LINE

  Obtains a line of data from the buffer with a newline character as the boundary. The obtained data is not output to the client.

  The prototype of DBE_OUTPUT.GET_LINE is as follows:

  ```
  DBE_OUTPUT.GET_LINE (
      line   OUT VARCHAR2,
      status OUT INTEGER
  );
  ```

  Table 2-246 DBE_OUTPUT.GET_LINE API parameters

Parameter	Description
line	Obtained character string.
status	Specifies whether the calling is normal. If a character string is obtained, the value is **0**. Otherwise, the value is **1**.

- DBE_OUTPUT.GET_LINES

  Obtains the character string of a specified number of lines in the buffer as a VARCHAR array. The obtained content is cleared from the buffer and is not output to the client.

  The prototype of DBE_OUTPUT.GET_LINES is as follows:

  ```
  DBE_OUTPUT.GET_LINES (
      lines    OUT   VARCHAR[],
      numlines IN OUT INTEGER
  );
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-247 DBE_OUTPUT.GET_LINES API parameters

Parameter	Description
lines	Outputs an array of multi-line strings read from the buffer.
numlines	Inputs the number of lines to be retrieved from the buffer. The output value is the number of rows actually retrieved. The value must be greater than 0 and cannot be **NULL**. If the output value is less than the input value, it indicates that the number of rows in the buffer is less than the input value.

- DBE_OUTPUT.NEW_LINE

  Places a line at the end of the buffer, places an end-of-line marker, and leaves a new line empty.

  The prototype of DBE_OUTPUT.NEW_LINE is as follows:

  ```
  DBE_OUTPUT.NEW_LINE;
  ```

- DBE_OUTPUT.PUT

  Places an input string in the buffer without a newline character at the end. After the anonymous block is executed, the line ending with the newline character is displayed.

  The prototype of DBE_OUTPUT.PUT is as follows:

  ```
  DBE_OUTPUT.PUT (
     item IN VARCHAR2);
  ```

  Table 2-248 DBE_OUTPUT.PUT API parameters

Parameter	Description
item	Character or string item to be placed in the buffer. The value can contain up to 32767 bytes.

- DBE_OUTPUT.PUT_LINE

  Places an input string in the buffer with a newline character at the end. After the anonymous block is executed, the line ending with the newline character is displayed.

  The prototype of DBE_OUTPUT.PUT_LINE is as follows:

  ```
  DBE_OUTPUT.PUT_LINE (
     item IN VARCHAR2);
  ```

  Table 2-249 DBE_OUTPUT.PUT_LINE API parameters

Parameter	Description
item	Character or string item to be placed in the buffer. The value can contain up to 32767 bytes.

Examples

```
BEGIN
   DBE_OUTPUT.SET_BUFFER_SIZE(50);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    DBE_OUTPUT.PRINT('hello, ');
    DBE_OUTPUT.PRINT_LINE('database!'); -- Output "hello, database!"
END;
/
-- The expected result is as follows:
hello, database!
ANONYMOUS BLOCK EXECUTE

-- Test DISABLE: Disable the calling of PUT, PUT_LINE, NEW_LINE, GET_LINE, and GET_LINES. The PUT_LINE
has no output.
BEGIN
 dbe_output.disable();
        dbe_output.put_line('1');
END;
/
-- The expected result is as follows:
ANONYMOUS BLOCK EXECUTE

-- Test ENABLE: Enable the calling of PUT, PUT_LINE, NEW_LINE, GET_LINE, and GET_LINES. The output of
PUT_LINE is 1.
BEGIN
 dbe_output.enable();
        dbe_output.put_line('1');
END;
/
-- The expected result is as follows:
1
ANONYMOUS BLOCK EXECUTE

-- Test PUT: Place the input character string a in the buffer without adding a newline character at the end.
a is not output.
BEGIN
 dbe_output.enable();
 dbe_output.put('a');
END;
/
-- The expected result is as follows:
ANONYMOUS BLOCK EXECUTE

-- Test NEW_LINE: Add a newline character. The output is a.
BEGIN
 dbe_output.enable();
 dbe_output.put('a');
        dbe_output.new_line;
END;
/
-- The expected result is as follows:
a
ANONYMOUS BLOCK EXECUTE

-- Test GET_LINE: Obtain buffer data and save the data to variables and use PUT_LINE to output the data.
DECLARE
 line VARCHAR(32672);
 status INTEGER := 0;
BEGIN
 dbe_output.put_line('hello');
 dbe_output.get_line(line, status);
 dbe_output.put_line('-----------');
 dbe_output.put_line(line);
 dbe_output.put_line(status);
END;
/
-- The expected result is as follows:
-----------
hello
0
ANONYMOUS BLOCK EXECUTE

-- Test GET_LINES: Obtain multiple lines of content in the buffer and use PUT_LINE to output the content.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DECLARE
    lines varchar[];
    linenum integer;
BEGIN
    dbe_output.put_line('line 1');
    dbe_output.put_line('line 2');
    dbe_output.put_line('line 3');
    linenum := 100;
    dbe_output.get_lines(lines, linenum);
    dbe_output.put_line('num: ' || linenum);
    FOR i IN 1 .. linenum LOOP
        dbe_output.put_line(lines[i]);
    END LOOP;
END;
/
-- The expected result is as follows:
num: 3
line 1
line 2
line 3
ANONYMOUS BLOCK EXECUTE
```

NOTE

When the character encoding type **server_encoding** of the server is not UTF-8, if the character encoding in the data is valid UTF-8 encoding, when processing the character encoding, the DBE_OUTPUT.PUT_LINE and DBE_OUTPUT.PUT functions convert the character encoding into the **server_encoding** encoding format based on the UTF-8 encoding logic before performing subsequent operations; as a result, the return values of the functions may not meet the expectation or an error may be reported. After the GUC parameter **enable_convert_illegal_char** is set to **on**, the DBE_OUTPUT.PUT_LINE and DBE_OUTPUT.PUT functions process such codes in the same way as when the parameter is disabled. Special characters are not output as placeholders; therefore, you are advised not to use the DBE_OUTPUT.PUT_LINE and DBE_OUTPUT.PUT functions for data containing special characters.

2.12.2.13 DBE_RANDOM

APIs

Table 2-250 lists APIs supported by the DBE_RANDOM package.

Table 2-250 DBE_RANDOM

API	Description
DBE_RANDOM.SET_SEED	Sets a seed for a random number.
DBE_RANDOM.GET_VALUE	Generates a random number between a specified **low** and a specified **high**.

- DBE_RANDOM.SET_SEED

  Sets a seed for a random number.

  The prototype of DBE_RANDOM.SET_SEED is as follows:

  ```
  DBE_RANDOM.SET_SEED (seed  IN  INTEGER);
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-251 DBE_RANDOM.SET_SEED API parameter

Parameter	Description
seed	Generates a seed for a random number.

- DBE_RANDOM.GET_VALUE

  Generates a random number between a specified **low** and a specified **high**.

  The prototype of DBE_RANDOM.GET_VALUE is as follows:

```
DBE_RANDOM.GET_VALUE(
min  IN  NUMBER default 0,
max  IN  NUMBER default 1)
RETURN NUMBER;
```

Table 2-252 DBE_RANDOM.GET_VALUE API parameter

Parameter	Description
min	Sets the low bound for a random number. The generated random number is greater than or equal to **min**.
max	Sets the high bound for a random number. The generated random number is less than **max**.

NOTE

- DBE_RANDOM.GET_VALUE requires that the parameter type be NUMERIC. There is no requirement on the size of the left and right bound values.
- DBE_RANDOM implements pseudo-random numbers. Therefore, if the initial value (seed) remains unchanged, the sequence of the pseudo-random numbers also remains unchanged. You are advised not to use DBE_RANDOM in security scenarios.
- The generated random number contains 15 valid digits.

Example:

```
-- Generate a random number between 0 and 1.
SELECT DBE_RANDOM.GET_VALUE(0,1);
    get_value
------------------
 .917468812743886
(1 row)

-- For integers within a specified range, add the arguments min and max, and truncate the decimals from the result (the maximum value is not included as a possible value). Therefore, for integers from 0 to 99, you can use the following code:
SELECT TRUNC(DBE_RANDOM.GET_VALUE(0,100));
 trunc
-------
    26
(1 row)
```

2.12.2.14 DBE_RAW

The DBE_RAW package provides APIs for operating RAW types.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

RAW data is represented as hexadecimal characters externally, and stored as binary characters internally. For example, one byte of RAW data with bits **11001011** is displayed and entered as **'CB'**.

Permissions

The DBE_RAW schema is a system catalog. Only a system administrator has the permission to create the DBE_RAW schema during database initialization. All users have the permission to use the schema and APIs under the schema.

APIs

Table 2-253 lists APIs supported by the DBE_RAW package.

Table 2-253 DBE_RAW

API	Description
DBE_RAW.CAST_FROM_BINARY_INTEGER_TO_RAW	Converts a value of the INTEGER type to a binary representation (RAW type).
DBE_RAW.CAST_FROM_RAW_TO_BINARY_INTEGER	Converts a RAW value to an INTEGER value.
DBE_RAW.GET_LENGTH	Obtains the length of a RAW object.
DBE_RAW.CAST_FROM_VARCHAR2_TO_RAW	Converts a VARCHAR2 value to a RAW value.
DBE_RAW.CAST_TO_VARCHAR2	Converts a value of the RAW type to a value of the VARCHAR2 type.
DBE_RAW.BIT_OR	Returns a RAW value after bitwise OR calculation.
DBE_RAW.SUBSTR	Returns the substring of a RAW value.
DBE_RAW.BIT_AND	Returns a RAW value after bitwise AND calculation.
DBE_RAW.BIT_COMPLEMENT	Returns a RAW value after bitwise NOT calculation.
DBE_RAW.BIT_XOR	Returns a RAW value after bitwise XOR calculation.
DBE_RAW.CAST_FROM_BINARY_DOUBLE_TO_RAW	Converts a BINARY_DOUBLE value to a RAW value.
DBE_RAW.CAST_FROM_RAW_TO_BINARY_DOUBLE	Converts a RAW value to a BINARY_DOUBLE value.
DBE_RAW.CAST_FROM_RAW_TO_BINARY_FLOAT	Converts a RAW value to a FLOAT4 value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_RAW.CAST_FROM_BINARY_FLOAT_TO_RAW	Converts a FLOAT4 value to a RAW value.
DBE_RAW.CAST_FROM_RAW_TO_NUMBER	Converts a RAW value to a NUMERIC value.
DBE_RAW.CAST_FROM_NUMBER_TO_RAW	Converts a NUMERIC value to a RAW value.
DBE_RAW.CAST_FROM_RAW_TO_NVARCHAR2	Converts a RAW value to an NVARCHAR2 value.
DBE_RAW.COMPARE	Returns the first different position of two RAW values.
DBE_RAW.CONCAT	Concatenates a maximum of 12 RAW values into a new RAW value and returns the result.
DBE_RAW.CONVERT	Converts a RAW value from the source encoding mode **from_charset** to the target encoding mode **to_charset**.
DBE_RAW.COPIES	Copies a RAW value for *n* times, concatenates the values, and returns the concatenated result.
DBE_RAW.OVERLAY	Overlays one RAW data with another RAW data by specifying the start position and length to be overlaid.
DBE_RAW.REVERSE	Reverses RAW data by byte.
DBE_RAW.TRANSLATE	Converts or discards a specified byte of a RAW value.
DBE_RAW.TRANSLITERATE	Converts a specified byte of a RAW value.
DBE_RAW.XRANGE	Returns a value containing the succession of one-byte encodings beginning and ending with the specified byte-code.

- DBE_RAW.CAST_FROM_BINARY_INTEGER_TO_RAW

  Converts an INTEGER value to a RAW value.

  The prototype of DBE_RAW.CAST_FROM_BINARY_INTEGER_TO_RAW is as follows:

  ```
  DBE_RAW.CAST_FROM_BINARY_INTEGER_TO_RAW (
  value          IN  BIGINT,
  endianess      IN  INTEGER DEFAULT 1)
  RETURN RAW;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-254 DBE_RAW.CAST_FROM_BINARY_INTEGER_TO_RAW API parameters

Parameter	Description
value	INTEGER value to be converted. **NOTE** DBE_RAW.CAST_FROM_BINARY_INTEGER_TO_RAW expects to process values of the INTEGER type. For compatibility purposes, this parameter is defined as the BIGINT type. The BIGINT type has a larger range than the INTEGER type, which does not affect the API use.
endianess	Specifies the endianness. ● **1**: BIG_ENDIAN. This is the default value. ● **2**: LITTLE-ENDIAN.

- DBE_RAW.CAST_FROM_RAW_TO_BINARY_INTEGER

  Converts a RAW value to an INTEGER value.

  The prototype of DBE_RAW.CAST_FROM_RAW_TO_BINARY_INTEGER is as follows:

```
DBE_RAW.CAST_FROM_RAW_TO_BINARY_INTEGER (
value          IN  RAW,
endianess      IN  INTEGER DEFAULT 1)
RETURN INTEGER;
```

Table 2-255 DBE_RAW.CAST_FROM_RAW_TO_BINARY_INTEGER API parameters

Parameter	Description
value	RAW value to be converted to the INTEGER type.
endianess	Endianness. The options are as follows: ● **1** (default value): BIG_ENDIAN. This is the default value. ● **2**: LITTLE_ENDIAN. ● **3**: MACHINE_ENDIAN. If BIG_ENDIAN is used on the host where the function is executed, the function execution result using MACHINE_ENDIAN is the same as that using BIG_ENDIAN. If LITTLE_ENDIAN is used on the host where the function is executed, the function execution result using MACHINE_ENDIAN is the same as that using LITTLE_ENDIAN.

- DBE_RAW.GET_LENGTH

  Returns the length of a RAW value.

  The prototype of DBE_RAW.GET_LENGTH is as follows:

```
DBE_RAW.GET_LENGTH(
value     IN    RAW)
RETURN INTEGER;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-256 DBE_RAW.GET_LENGTH API parameters

Parameter	Description
value	RAW object.

- DBE_RAW.CAST_FROM_VARCHAR2_TO_RAW

  Converts a VARCHAR2 value to a RAW value.

  The prototype of DBE_RAW.CAST_FROM_VARCHAR2_TO_RAW is as follows:

```
DBE_RAW.CAST_TO_RAW(
str     IN    VARCHAR2)
RETURN RAW;
```

Table 2-257 DBE_RAW.CAST_FROM_VARCHAR2_TO_RAW API parameters

Parameter	Description
str	VARCHAR2 value to be converted.

- DBE_RAW.CAST_TO_VARCHAR2

  Converts a RAW value to a VARCHAR2 value.

  The prototype of DBE_RAW.CAST_TO_VARCHAR2 is as follows:

```
DBE_RAW.CAST_TO_VARCHAR2(
str     IN    RAW)
RETURN VARCHAR2;
```

Table 2-258 DBE_RAW.CAST_TO_VARCHAR2 API parameters

Parameter	Description
str	VARCHAR2 value to be converted.

- DBE_RAW.BIT_OR

  Returns two RAW values after bitwise OR calculation.

  The prototype of DBE_RAW.BIT_OR is as follows:

```
DBE_RAW.BIT_OR(
   str1 IN TEXT,
   str2 IN TEXT)
RETURN TEXT;
```

Table 2-259 DBE_RAW.BIT_OR API parameters

Parameter	Description
str1	First character string of the bitwise OR calculation. **NOTE** Due to legacy reasons, this parameter is defined as the TEXT type. However, the DBE_RAW.BIT_OR API expects to process RAW values. Defining this parameter as the TEXT type does not affect the processing of RAW values.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
str2	Second character string of the bitwise OR calculation. **NOTE** Due to legacy reasons, this parameter is defined as the TEXT type. However, the DBE_RAW.BIT_OR API expects to process RAW values. Defining this parameter as the TEXT type does not affect the processing of RAW values.

- DBE_RAW.SUBSTR

  Truncates a RAW value based on the start position (*off_set*) and length (*amount*).

  The prototype of DBE_RAW.SUBSTR is as follows:

```
DBE_RAW.SUBSTR(
   IN lob_loc BLOB,
   IN off_set INTEGER DEFAULT 1,
   IN amount INTEGER DEFAULT 32767)
RETURN RAW;
```

  Table 2-260 DBE_RAW.SUBSTR API parameters

Parameter	Description
lob_loc	RAW value to be truncated. **NOTE** Due to legacy reasons, this parameter is defined as the BLOB type. However, the DBE_RAW.SUBSTR API expects to process RAW values. Defining this parameter as the BLOB type does not affect the processing of RAW values.
off_set	Start position of a substring. The default value is **1**.
amount	Length of a substring. The default value is **32767**.

- DBE_RAW.BIT_AND

  Obtains the bitwise AND result of two RAW values.

  The prototype of DBE_RAW.BIT_AND is as follows:

```
DBE_RAW.BIT_AND(
   r1 IN RAW,
   r2 IN RAW)
RETURN RAW;
```

  Table 2-261 DBE_RAW.BIT_AND API parameters

Parameter	Description
r1	The RAW value for the bitwise AND operation with r2. The maximum length is 32767.
r2	The RAW value for the bitwise AND operation with r1. The maximum length is 32767.

- DBE_RAW.BIT_COMPLEMENT

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Obtains the bitwise NOT result of a RAW value.

The prototype of DBE_RAW.BIT_COMPLEMENT is as follows:

```
DBE_RAW.BIT_COMPLEMENT(
   r IN RAW)
RETURN RAW;
```

Table 2-262 DBE_RAW.BIT_COMPLEMENT API parameters

Parameter	Description
r	RAW value for the bitwise NOT operation. The maximum length is 32767.

- DBE_RAW.BIT_XOR

  Obtains the bitwise XOR result of two RAW values.

  The prototype of DBE_RAW.BIT_XOR is as follows:

  ```
  DBE_RAW.BIT_XOR(
     r1 IN RAW,
     r2 IN RAW)
  RETURN RAW;
  ```

  Table 2-263 DBE_RAW.BIT_XOR API parameters

Parameter	Description
r1	RAW value for the bitwise XOR operation with r2. The maximum length is 32767.
r2	RAW value for the bitwise XOR operation with r1. The maximum length is 32767.

- DBE_RAW.CAST_FROM_BINARY_DOUBLE_TO_RAW

  Converts a BINARY_DOUBLE value to a RAW value.

  The prototype of DBE_RAW.CAST_FROM_BINARY_DOUBLE_TO_RAW is as follows:

  ```
  DBE_RAW.CAST_FROM_BINARY_DOUBLE_TO_RAW (
     n         IN BINARY_DOUBLE,
     endianess IN INTEGER DEFAULT 1)
  RETURN RAW;
  ```

  Table 2-264 DBE_RAW.CAST_FROM_BINARY_DOUBLE_TO_RAW API parameters

Parameter	Description
n	BINARY_DOUBLE value to be converted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
endianess	Endianness. The options are as follows: ● **1** (default value): BIG_ENDIAN. This is the default value. ● **2**: LITTLE_ENDIAN. ● **3**: MACHINE_ENDIAN. If BIG_ENDIAN is used on the host where the function is executed, the function execution result using MACHINE_ENDIAN is the same as that using BIG_ENDIAN. If LITTLE_ENDIAN is used on the host where the function is executed, the function execution result using MACHINE_ENDIAN is the same as that using LITTLE_ENDIAN.

- DBE_RAW.CAST_FROM_RAW_TO_BINARY_DOUBLE

  Converts a RAW value to a BINARY_DOUBLE value.

  The prototype of DBE_RAW.CAST_FROM_RAW_TO_BINARY_DOUBLE is as follows:

```
DBE_RAW.CAST_FROM_RAW_TO_BINARY_DOUBLE(
   r         IN RAW,
   endianess IN INTEGER DEFAULT 1)
RETURN BINARY_DOUBLE;
```

  Table 2-265 DBE_RAW.CAST_FROM_RAW_TO_BINARY_DOUBLE API parameters

Parameter	Description
r	RAW value to be converted. The value contains 8 to 32767 characters.
endianess	Endianness. The options are as follows: ● **1** (default value): BIG_ENDIAN. This is the default value. ● **2**: LITTLE_ENDIAN. ● **3**: MACHINE_ENDIAN. If BIG_ENDIAN is used on the host where the function is executed, the function execution result using MACHINE_ENDIAN is the same as that using BIG_ENDIAN. If LITTLE_ENDIAN is used on the host where the function is executed, the function execution result using MACHINE_ENDIAN is the same as that using LITTLE_ENDIAN.

- DBE_RAW.CAST_FROM_RAW_TO_BINARY_FLOAT

  Converts a RAW value to a FLOAT4 value.

  The prototype of DBE_RAW.CAST_FROM_RAW_TO_BINARY_FLOAT is as follows:

```
DBE_RAW.CAST_FROM_RAW_TO_BINARY_FLOAT(
   r         IN RAW,
   endianess IN INTEGER DEFAULT 1)
RETURN FLOAT4;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-266 DBE_RAW.CAST_FROM_RAW_TO_BINARY_FLOAT API parameters

Parameter	Description
r	RAW value to be converted. The value contains 4 to 32767 characters.
endianess	Endianness. The options are as follows: ● **1** (default value): BIG_ENDIAN. This is the default value. ● **2**: LITTLE_ENDIAN. ● **3**: MACHINE_ENDIAN. If BIG_ENDIAN is used on the host where the function is executed, the function execution result using MACHINE_ENDIAN is the same as that using BIG_ENDIAN. If LITTLE_ENDIAN is used on the host where the function is executed, the function execution result using MACHINE_ENDIAN is the same as that using LITTLE_ENDIAN.

- DBE_RAW.CAST_FROM_BINARY_FLOAT_TO_RAW

  Converts a FLOAT4 value to a RAW value.

  NOTE

  Before converting a FLOAT4 value *n* to a RAW value, DBE_RAW.CAST_FROM_BINARY_FLOAT_TO_RAW truncates the input value based on its range and precision. The specific rules are as follows:

  - If *n* is greater than the maximum value 3.40282E+38 of FLOAT4, 3.40282E+38 is used.
  - If *n* is less than the minimum value –3.40282E+38 of FLOAT4, –3.40282E+38 is used.
  - If –1.4013E–45 < *n* < 1.4013E–45, **0** is used.

  The final conversion result is generated based on the truncated value.

  The prototype of DBE_RAW.CAST_FROM_BINARY_FLOAT_TO_RAW is as follows:

```
DBE_RAW.CAST_FROM_BINARY_FLOAT_TO_RAW(
   n         IN FLOAT4,
   endianess IN INTEGER DEFAULT 1)
RETURN RAW;
```

Table 2-267 DBE_RAW.CAST_FROM_BINARY_FLOAT_TO_RAW API parameters

Parameter	Description
n	FLOAT4 value to be converted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
endianess	Endianness. The options are as follows: • **1** (default value): BIG_ENDIAN. This is the default value. • **2**: LITTLE_ENDIAN. • **3**: MACHINE_ENDIAN. If BIG_ENDIAN is used on the host where the function is executed, the function execution result using MACHINE_ENDIAN is the same as that using BIG_ENDIAN. If LITTLE_ENDIAN is used on the host where the function is executed, the function execution result using MACHINE_ENDIAN is the same as that using LITTLE_ENDIAN.

- DBE_RAW.CAST_FROM_RAW_TO_NUMBER

  Converts a RAW value to a NUMERIC value.

  NOTE

  The bottom-layer implementation of the number data type in database ORA is different from that in GaussDB, and the RAW data type is the hexadecimal representation of the binary stream implemented at the bottom layer. Therefore, the function in database ORA is different from that in GaussDB. You cannot obtain the same NUMBER-type data from GaussDB based on the RAW data corresponding to the NUMBER-type data in database ORA. For details about the NUMBER-type data in GaussDB, see **Examples**.

  The prototype of DBE_RAW.CAST_FROM_RAW_TO_NUMBER is as follows:

```
DBE_RAW.CAST_FROM_RAW_TO_NUMBER(
    r IN RAW)
RETURN NUMERIC;
```

Table 2-268 DBE_RAW.CAST_FROM_RAW_TO_NUMBER API parameters

Parameter	Description
r	RAW value to be converted. The value contains 4 to 32767 characters.

- DBE_RAW.CAST_FROM_NUMBER_TO_RAW

  Converts a NUMERIC value to a RAW value.

  NOTE

  The bottom-layer implementation of the number data type in database ORA is different from that in GaussDB, and the RAW data type is the hexadecimal representation of the binary stream implemented at the bottom layer. Therefore, the function in database ORA is different from that in GaussDB. You cannot restore the RAW data corresponding to the NUMBER-type data in database ORA in GaussDB. For details about the performance of this function in GaussDB, see **Examples**.

  The prototype of DBE_RAW.CAST_FROM_NUMBER_TO_RAW is as follows:

```
DBE_RAW.CAST_FROM_NUMBER_TO_RAW(
    value IN NUMERIC)
RETURN RAW;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-269 DBE_RAW.CAST_FROM_NUMBER_TO_RAW API parameters

Parameter	Description
value	NUMERIC value to be converted.

- DBE_RAW.CAST_FROM_RAW_TO_NVARCHAR2

  Converts a RAW value to an NVARCHAR2 value.

  The prototype of DBE_RAW.CAST_FROM_RAW_TO_NVARCHAR2 is as follows:

```
DBE_RAW.CAST_FROM_RAW_TO_NVARCHAR2(
   r IN RAW)
RETURN NVARCHAR2;
```

Table 2-270 DBE_RAW.CAST_FROM_RAW_TO_NVARCHAR2 API parameters

Parameter	Description
r	RAW value to be converted. The maximum length is 32767.

- DBE_RAW.COMPARE

  Returns the first different position of two RAW values.

  The prototype of DBE_RAW.COMPARE is as follows:

```
DBE_RAW.COMPARE(
   r1  IN RAW,
   r2  IN RAW,
   pad IN RAW DEFAULT NULL)
RETURN INTEGER;
```

Table 2-271 DBE_RAW.COMPARE API parameters

Parameter	Description
r1	First data to be compared. The value can be **NULL** or the length is 0. The maximum length is 32767.
r2	Second data to be compared. The value can be **NULL** or the length is 0. The maximum length is 32767.
pad	The first byte of **pad** is used to extend the shorter one of **r1** or **r2**. The maximum length is 32767. If this parameter is set to **NULL**, the length is 0, or the default value is used, the extended value is **0x00**.

- DBE_RAW.CONCAT

  Concatenates a maximum of 12 RAW values into a new RAW value and returns the value. If the length after concatenation exceeds 32767, an error is reported.

  The prototype of DBE_RAW.CONCAT is as follows:

```
DBE_RAW.CONCAT(
   r1  IN RAW DEFAULT NULL,
   r2  IN RAW DEFAULT NULL,
   r3  IN RAW DEFAULT NULL,
   r4  IN RAW DEFAULT NULL,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  r5  IN RAW DEFAULT NULL,
  r6  IN RAW DEFAULT NULL,
  r7  IN RAW DEFAULT NULL,
  r8  IN RAW DEFAULT NULL,
  r9  IN RAW DEFAULT NULL,
  r10 IN RAW DEFAULT NULL,
  r11 IN RAW DEFAULT NULL,
  r12 IN RAW DEFAULT NULL)
RETURN RAW;
```

Table 2-272 DBE_RAW.CONCAT API parameters

Parameter	Description
r1...r12	RAW values to be concatenated.

- DBE_RAW.CONVERT

  Converts a RAW value from the source encoding mode **from_charset** to the target encoding mode **to_charset**.

  If the rule for converting between source and target encoding (for example, GBK and LATIN1) does not exist, the parameter **r** is returned without conversion. See the **PG_CONVERSION** system catalog for details.

  The prototype of DBE_RAW.CONVERT is as follows:

```
DBE_RAW.CONVERT(
  r           IN RAW,
  to_charset   IN NAME,
  from_charset IN NAME)
RETURN RAW;
```

Table 2-273 DBE_RAW.CONVERT API parameters

Parameter	Description
r	RAW value to be converted. The maximum length is 32767.
to_charset	Name of the target character set.
from_charset	Name of the source character set. In this encoding, **r** must be valid.

- DBE_RAW.COPIES

  Copies a RAW value for *n* times, concatenates the values, and returns the concatenated result. If the length after copying exceeds 32767, an error is reported.

  The prototype of DBE_RAW.COPIES is as follows:

```
DBE_RAW.COPIES(
  r IN RAW,
  n IN NUMERIC)
RETURN RAW;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-274 DBE_RAW.COPIES API parameters

Parameter	Description
r	RAW values to be copied.
n	Number of replication times. The value must be a positive number. If the value is a decimal, the value is rounded off.

- DBE_RAW.OVERLAY

  Overlays one RAW value with another RAW value by specifying the start position and length to be overlaid.

  The prototype of DBE_RAW.OVERLAY is as follows:

```
DBE_RAW.OVERLAY(
   overlay_str IN RAW,
   target      IN RAW,
   pos         IN BINARY_INTEGER DEFAULT 1,
   len         IN BINARY_INTEGER DEFAULT NULL,
   pad         IN RAW            DEFAULT NULL)
RETURN RAW;
```

Table 2-275 DBE_RAW.OVERLAY API parameters

Parameter	Description
overlay_str	Byte used for overwriting. The value cannot be **NULL**.
target	Overwritten byte string, which is of the RAW type. The length can exceed 32767 bytes. However, due to the constraints of the **pos** and **len** parameters, the overwrite operation can be performed only in the first 32767 bytes and the value cannot be NULL.
pos	Byte from which the overlay starts. The position of the first byte is 1. The value of **pos** must be greater than or equal to 1 and the value of **len** + (**pos** – 1) must be less than or equal to 32767. The default value is **1**.
len	Length to be overlaid. The value of **len** must be greater than or equal to 0 and the value of **len** + (**pos** – 1) must be less than or equal to 32767. The default value is the length of **overlay_str**.
pad	Padding byte. Only the first byte is valid. The default value is **NULL**. If the value is **NULL**, it is regarded as **0x00** by default.

- DBE_RAW.REVERSE

  Reverses RAW values by byte.

  The prototype of DBE_RAW.REVERSE is as follows:

```
DBE_RAW.REVERSE(
   r IN RAW
)
RETURN RAW;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-276 DBE_RAW.REVERSE API parameters

Parameter	Description
r	RAW value to be reversed. The maximum length is 32767. If the value is **NULL**, **NULL** is returned.

- DBE_RAW.TRANSLATE

  Converts or discards a specified byte of a RAW value.

  The prototype of DBE_RAW.TRANSLATE is as follows:

```
DBE_RAW.TRANSLATE(
   r        IN RAW,
   from_set IN RAW,
   to_set   IN RAW)
RETURN RAW;
```

Table 2-277 DBE_RAW.TRANSLATE API parameters

Parameter	Description
r	Source byte string to be converted. The value is of the RAW type and contains a maximum of 32767 bytes. The value cannot be **NULL**.
from_set	Bytecode to be converted in the source byte string. The value is of the RAW type. The value cannot be **NULL**. The bytes in **from_set** in the source byte string are replaced with the bytes in the corresponding positions in **to_set**. If **from_set** contains multiple identical bytes, only the first byte corresponding to the byte is replaced. For example, if **r[x]=from_set[n]**, **r[x]** is replaced with **to_set[n]**. If **to_set[n]** corresponding to **from_set[n]** does not exist (that is, the number of bytes of **to_set** does not exceed **n**), **r[x]** will be discarded.
to_set	Byte code converted from the **from_set** byte. The value is of the RAW type. The value cannot be **NULL**.

- DBE_RAW.TRANSLITERATE

  Converts a RAW value to a NUMERIC value.

  The prototype of DBE_RAW.TRANSLITERATE is as follows:

```
DBE_RAW.TRANSLITERATE(
   r        IN RAW,
   to_set   IN RAW DEFAULT NULL,
   from_set IN RAW DEFAULT NULL,
   pad      IN RAW DEFAULT NULL)
RETURN NUMERIC;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-278 DBE_RAW.TRANSLITERATE API parameters

Parameter	Description
r	Byte string to be converted. The value is of the RAW type and contains a maximum of 32767 bytes. The value cannot be **NULL**.
to_set	Byte code converted from the **from_set** byte. The value is of the RAW type. The default value is **NULL**. If the value is **NULL**, all bytes in **r** that exist in **from_set** are converted to **pad**.
from_set	Bytecode to be converted in the byte string **r**. The value is of the RAW type. The default value is **NULL**. If **from_set** is **NULL**, all bytes in the byte string **r** are converted into equal-length data filled by **pad**. Otherwise, the bytes in **from_set** in the byte string **r** are replaced with the bytes in the corresponding position in **to_set**. For example, if **r[x]=from_set[n]**, **r[x]** is converted to **to_set[n]**. If **to_set[n]** does not exist, **r[x]** will be converted to **pad**.
pad	Padding byte. Only the first byte is valid. The default value is **NULL**. If the value is **NULL**, it is regarded as **0x00** by default.

- DBE_RAW.XRANGE

  Returns a RAW value containing the succession of one-byte encodings beginning and ending with the specified byte-code.

  The prototype of DBE_RAW.XRANGE is as follows:

```
DBE_RAW.XRANGE(
   start_byte IN RAW,
   end_byte   IN RAW)
RETURN RAW;
```

Table 2-279 DBE_RAW.XRANGE API parameters

Parameter	Description
start_byte	Start byte. Only the first byte is valid. If the value is **NULL**, it is regarded as **0x00** by default.
end_byte	End byte. Only the first byte is valid. If the value is **NULL**, it is regarded as **0xFF** by default. If the value of **end_byte** is less than that of **start_byte**, the function triggers the wraparound mechanism. Starting from the start byte, the value increases to the maximum hexadecimal value FF and then continues to increase from 00 until the end byte is reached.

Examples

```
-- Perform operations on RAW data in a stored procedure.
CREATE OR REPLACE PROCEDURE proc_raw
AS
str varchar2(100) := 'abcdef';
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
source raw(100);
amount integer;
BEGIN
source := dbe_raw.cast_from_varchar2_to_raw(str); -- Convert the type.
amount := dbe_raw.get_length(source); -- Obtain the length.
dbe_output.print_line(amount);
END;
/
CREATE PROCEDURE

-- Call the stored procedure.
CALL proc_raw();
6
 proc_raw
----------

(1 row)
-- Drop the stored procedure.
DROP PROCEDURE proc_raw;
DROP PROCEDURE

DECLARE
v_raw RAW;
v_double BINARY_DOUBLE;
v_float FLOAT4;
v_numeric NUMERIC;
v_nvarchar2 NVARCHAR2;
BEGIN
-- Perform bitwise AND calculation on RAW values.
SELECT DBE_RAW.BIT_AND('AFF', 'FF0B') INTO v_raw; -- 0A0B
-- Perform bitwise NOT calculation on RAW values.
SELECT DBE_RAW.BIT_COMPLEMENT('0AFF') INTO v_raw; -- F500
-- Perform bitwise XOR calculation on RAW values.
SELECT DBE_RAW.BIT_XOR('AFF', 'FF0B') INTO v_raw; -- F5F4
-- Convert a BINARY_DOUBLE value to a RAW value.
SELECT DBE_RAW.CAST_FROM_BINARY_DOUBLE_TO_RAW(1.0001,1) INTO v_raw; -- 3FF00068DB8BAC71
-- Convert a RAW value to a BINARY_DOUBLE value.
SELECT DBE_RAW.CAST_FROM_RAW_TO_BINARY_DOUBLE('3FF00068DB8BAC71',1) INTO v_double; --
1.0001
-- Convert a RAW value to a FLOAT4 value.
SELECT DBE_RAW.CAST_FROM_RAW_TO_BINARY_FLOAT('40200000',1) INTO v_float; -- 2.5
-- Convert a FLOAT4 value to a RAW value.
SELECT DBE_RAW.CAST_FROM_BINARY_FLOAT_TO_RAW('2.5',1) INTO v_raw; -- 40200000
-- Convert a RAW value to a NUMERIC value.
SELECT DBE_RAW.CAST_FROM_RAW_TO_NUMBER('808002008813') INTO v_numeric; -- 2.5
-- Convert a NUMERIC value to a RAW value.
SELECT DBE_RAW.CAST_FROM_NUMBER_TO_RAW('2.5') INTO v_raw; -- 808002008813
-- Convert a RAW value to an NVARCHAR2 value.
SELECT DBE_RAW.CAST_FROM_RAW_TO_NVARCHAR2('12345678') INTO v_nvarchar2; -- \x124Vx
-- Compare RAW values.
SELECT DBE_RAW.COMPARE('ABCD','AB') INTO v_numeric; -- 2
-- Concatenate RAW values.
SELECT DBE_RAW.CONCAT('ABCD','AB') INTO v_raw; -- ABCDAB
-- Convert RAW values.
SELECT DBE_RAW.CONVERT('E695B0', 'GBK','UTF8') INTO v_raw; -- CAFD
-- Copy RAW values.
SELECT DBE_RAW.COPIES('ABCD',2) INTO v_raw; -- ABCDABCD
-- Specify the start position and length of a RAW value to be overlaid.
SELECT DBE_RAW.OVERLAY('abcef', '12345678123456', 2, 5, '9966') INTO v_raw; -- 120ABCEF999956
-- Reverse a RAW value by byte.
SELECT DBE_RAW.REVERSE('12345678') INTO v_raw; -- 78563412
-- Convert bytes of the RAW type (without padding code).
SELECT DBE_RAW.TRANSLATE('1122112233', '1133','55') INTO v_raw; -- 55225522
-- Convert bytes of the RAW type (with padding code).
SELECT DBE_RAW.TRANSLITERATE('1122112233', '55','1133','FFEE') INTO v_raw; -- 55225522FF
-- Return all bytes between two bytes of the RAW type.
SELECT DBE_RAW.XRANGE('00','03') INTO v_raw; -- 00010203
SELECT DBE_RAW.XRANGE('FA'::raw, '01'::raw); -- FAFBFCFDFEFF0001
END;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
/
ANONYMOUS BLOCK EXECUTE
```

2.12.2.15 DBE_SCHEDULER

APIs

The DBE_SCHEDULER advanced package supports more flexible creation of scheduled jobs through scheduling and programing. For details about supported APIs, see **Table 2-280**.

NOTE

- DBE_SCHEDULER supports scheduled jobs for synchronizing data between CNs. Jobs with the same name cannot be created between different CNs.
- The synchronization function is not implemented in versions earlier than 505.0.0. Therefore, all synchronization APIs synchronize data only in 505.0.0 and later versions.
- Before the upgrade, the system checks whether jobs with the same name exist on CNs of the source version. If jobs with the same name exist on CNs of the source version and they are created on different nodes, an error message is displayed, prompting you to delete the jobs and perform the upgrade again.

Table 2-280 DBE_SCHEDULER

API	Description
DBE_SCHEDULER.CREATE_JOB	Creates a scheduled job.
DBE_SCHEDULER.DROP_JOB	Deletes scheduled jobs.
DBE_SCHEDULER.DROP_SINGLE_JOB	Deletes a single scheduled job.
DBE_SCHEDULER.SET_ATTRIBUTE	Sets object attributes.
DBE_SCHEDULER.RUN_JOB	Runs a scheduled job.
DBE_SCHEDULER.RUN_BACKEND_JOB	Runs a scheduled job in the background.
DBE_SCHEDULER.RUN_FOREGROUND_JOB	Runs a scheduled job in the foreground.
DBE_SCHEDULER.STOP_JOB	Stops scheduled jobs.
DBE_SCHEDULER.STOP_SINGLE_JOB	Stops a single scheduled job.
DBE_SCHEDULER.GENERATE_JOB_NAME	Generates a scheduled job name.
DBE_SCHEDULER.CREATE_PROGRAM	Creates a program.
DBE_SCHEDULER.DEFINE_PROGRAM_ARGUMENT	Defines program parameters.
DBE_SCHEDULER.DROP_PROGRAM	Deletes programs.
DBE_SCHEDULER.DROP_SINGLE_PROGRAM	Deletes a single program.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_SCHEDULER.SET_JOB_ARGUMENT_VALUE	Sets scheduled job parameters.
DBE_SCHEDULER.CREATE_SCHEDULE	Creates a schedule.
DBE_SCHEDULER.DROP_SCHEDULE	Deletes schedules.
DBE_SCHEDULER.DROP_SINGLE_SCHEDULE	Deletes a single schedule.
DBE_SCHEDULER.CREATE_JOB_CLASS	Creates a scheduled job class.
DBE_SCHEDULER.DROP_JOB_CLASS	Deletes scheduled job classes.
DBE_SCHEDULER.DROP_SINGLE_JOB_CLASS	Deletes a single scheduled job class.
DBE_SCHEDULER.GRANT_USER_AUTHORIZATION	Grants special permissions to a user.
DBE_SCHEDULER.REVOKE_USER_AUTHORIZATION	Revokes special permissions from a user.
DBE_SCHEDULER.CREATE_CREDENTIAL	Creates a certificate.
DBE_SCHEDULER.DROP_CREDENTIAL	Destroys a certificate.
DBE_SCHEDULER.ENABLE	Enables objects.
DBE_SCHEDULER.ENABLE_SINGLE	Enables a single object.
DBE_SCHEDULER.DISABLE	Disables objects.
DBE_SCHEDULER.DISABLE_SINGLE	Disables a single object.
DBE_SCHEDULER.EVAL_CALENDAR_STRING	Analyzes the scheduled job period.
DBE_SCHEDULER.EVALUATE_CALENDAR_STRING	Analyzes the scheduled job period.

- DBE_SCHEDULER.CREATE_JOB

  Creates a scheduled job and synchronizes it to all CNs.

  The prototypes of DBE_SCHEDULER.CREATE_JOB are as follows:

```
-- Scheduled jobs of an inline schedule and a program.
DBE_SCHEDULER.CREATE_JOB(
job_name TEXT,
job_type TEXT,
job_action TEXT,
number_of_arguments INTEGER             DEFAULT 0,
start_date TIMESTAMP WITH TIME ZONE     DEFAULT NULL,
repeat_interval TEXT                    DEFAULT NULL,
end_date TIMESTAMP WITH TIME ZONE       DEFAULT NULL,
job_class TEXT          DEFAULT 'DEFAULT_JOB_CLASS',
enabled BOOLEAN                         DEFAULT FALSE,
auto_drop BOOLEAN                       DEFAULT TRUE,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
comments TEXT                       DEFAULT NULL,
credential_name TEXT                DEFAULT NULL,
destination_name TEXT               DEFAULT NULL
)

-- Reference the created scheduled jobs of the schedule and the program.
DBE_SCHEDULER.CREATE_JOB(
job_name TEXT,
program_name TEXT,
schedule_name TEXT,
job_class TEXT          DEFAULT 'DEFAULT_JOB_CLASS',
enabled BOOLEAN                     DEFAULT FALSE,
auto_drop BOOLEAN                   DEFAULT TRUE,
comments TEXT                       DEFAULT NULL,
job_style TEXT                      DEFAULT 'REGULAR',
credential_name TEXT                DEFAULT NULL,
destination_name TEXT               DEFAULT NULL
)

-- Reference the created program and the scheduled job of the inline schedule.
DBE_SCHEDULER.CREATE_JOB(
job_name text,
program_name TEXT,
start_date TIMESTAMP WITH TIME ZONE     DEFAULT NULL,
repeat_interval TEXT                DEFAULT NULL,
end_date TIMESTAMP WITH TIME ZONE       DEFAULT NULL,
job_class TEXT          DEFAULT 'DEFAULT_JOB_CLASS',
enabled BOOLEAN                     DEFAULT FALSE,
auto_drop BOOLEAN                   DEFAULT TRUE,
comments TEXT                       DEFAULT NULL,
job_style TEXT                      DEFAULT 'REGULAR',
credential_name TEXT                DEFAULT NULL,
destination_name TEXT               DEFAULT NULL
)

-- Reference the created schedule and the scheduled job of the inline program.
DBE_SCHEDULER.CREATE_JOB(
job_name TEXT,
schedule_name TEXT,
job_type TEXT,
job_action TEXT,
number_of_arguments INTEGER         DEFAULT 0,
job_class TEXT          DEFAULT 'DEFAULT_JOB_CLASS',
enabled BOOLEAN                     DEFAULT FALSE,
auto_drop BOOLEAN                   DEFAULT TRUE,
comments TEXT                       DEFAULT NULL,
credential_name TEXT                DEFAULT NULL,
destination_name TEXT               DEFAULT NULL
)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The scheduled job created by DBE_SCHEDULER does not conflict with the scheduled job in DBE_TASK.
- The scheduled job created by DBE_SCHEDULER generates the corresponding **job_id**. However, the **job_id** is meaningless.
- For the CREATE API, the input parameter types are not validated. Successful creation does not mean successful execution. You can query the execution status of the current job in the **PG_JOB** system catalog.
- Common users cannot execute scheduled jobs, unless they are granted the required permission using DBE_SCHEDULER.GRANT_USER_AUTHORIZATION.
- When **job_type** is set to **'PLSQL_BLOCK'**, program parameters are not accepted, and the value of **number_of_arguments** cannot be greater than 0. Otherwise, the scheduled job will fail to be created.
- If the value of **start_date** is greater than that of **end_date**, the job can be created successfully and no error is reported. However, the job cannot be triggered for scheduling.

Table 2-281 DBE_SCHEDULER.CREATE_JOB API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
job_name	TEXT	IN	No	Name of a scheduled job. In non-ILM scheduling scenarios, jobs prefixed with "ilmadowind$_" and "ilmjob$_" cannot be created.
job_type	TEXT	IN	No	Inline program type of a scheduled job. The options are as follows: • **'PLSQL_BLOCK'**: fast anonymous stored procedure. • **'STORED_PROCEDURE'**: stored procedure that is saved. • **'EXTERNAL_SCRIPT'**: external script.
job_action	TEXT	IN	No	Content executed by the inline program of a scheduled job.
number_of_arguments	INTEGER	IN	No	Number of inline program parameters of a scheduled job. The value ranges from 0 to 255.
program_name	TEXT	IN	No	Name of the program referenced by a scheduled job.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
start_date	TIMESTAMP WITH TIME ZONE	IN	Yes	Inline scheduling start time of a scheduled job.
repeat_interval	TEXT	IN	Yes	Inline scheduling period of a scheduled job.
end_date	TIMESTAMP WITH TIME ZONE	IN	Yes	Inline scheduling expiration time of a scheduled job.
schedule_name	TEXT	IN	No	Name of the schedule referenced by a scheduled job.
job_class	TEXT	IN	No	Class name of a scheduled job.
enabled	BOOLEAN	IN	No	Status of a scheduled job.
auto_drop	BOOLEAN	IN	No	Automatic deletion of a scheduled job.
comments	TEXT	IN	Yes	Comments.
job_style	TEXT	IN	No	Behavior pattern of a scheduled job. Only **'REGULAR'** is supported.
credential_name	TEXT	IN	Yes	Certificate name of a scheduled job.
destination_name	TEXT	IN	Yes	Target name of a scheduled job. Currently, only **'CCN'** (case-insensitive) is supported. Jobs whose **destination_name** is set to **CCN** can be executed only on the central coordinator. Otherwise, the original processing logic is used.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_program('program1', 'STORED_PROCEDURE', 'select
pg_sleep(1);', 3, false, 'test');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 create_program
----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.create_schedule('schedule1', NULL, 'sysdate', NULL, 'test');
 create_schedule
-----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.create_job(job_name=>'job1', program_name=>'program1',
schedule_name=>'schedule1');
 create_job
------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job('job1', true, false, 'STOP_ON_FIRST_ERROR');
 drop_job
----------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_schedule('schedule1');
 drop_schedule
---------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_program('program1', false);
 drop_program
--------------

(1 row)
```

NOTE

To create a scheduled job of the EXTERNAL_SCRIPT type, the administrator must grant related permissions and certificates and the user who starts the database must have the read permission on the external script.

- DBE_SCHEDULER.DROP_JOB

  Deletes scheduled jobs. The operation can be performed only when the job has been synchronized.

  The prototype of the DBE_SCHEDULER.DROP_JOB function is as follows:

```
DBE_SCHEDULER.drop_job(
job_name text,
force boolean                   default false,
defer boolean                   default false,
commit_semantics text      default 'STOP_ON_FIRST_ERROR'
)
```

NOTE

DBE_SCHEDULER.DROP_JOB can delete one or more scheduled jobs, or a scheduled job class.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-282 DBE_SCHEDULER.DROP_JOB API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
job_name	TEXT	IN	No	One or more scheduled job names or classes. If you specify multiple scheduled jobs, separate them with commas (,).
force	BOOLEAN	IN	No	Specifies how to delete a scheduled job. ● **true**: The current scheduled job is stopped and then deleted. ● **false**: The scheduled job fails to be deleted if it is running.
defer	BOOLEAN	IN	No	Specifies how to delete a scheduled job. ● **true**: A scheduled job can be deleted after it is complete. ● **false**: The scheduled job cannot be executed and is deleted.
commit_semantics	TEXT	IN	No	Commit rules: ● **'STOP_ON_FIRST_ERROR'**: The deletion operation performed before the first error is reported is committed. ● **'TRANSACTIONAL'**: Transaction-level commit. The deletion operation performed before an error is reported will be rolled back. ● **'ABSORB_ERRORS'**: The system attempts to bypass an error and commit the deletion operation that is performed successfully.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_program('program1', 'STORED_PROCEDURE', 'SELECT
pg_sleep(1);', 3, false, 'test');
 create_program
----------------

(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CALL DBE_SCHEDULER.create_schedule('schedule1', NULL, 'sysdate', NULL, 'test');
 create_schedule
-----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.create_job(job_name=>'job1', program_name=>'program1',
schedule_name=>'schedule1');
 create_job
------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job('job1', true, false, 'STOP_ON_FIRST_ERROR');
 drop_job
----------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_schedule('schedule1');
 drop_schedule
---------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_program('program1', false);
 drop_program
--------------

(1 row)
```

NOTE

The **TRANSACTIONAL** option in **commit_semantic** takes effect only when **force** is set to **false**.

- DBE_SCHEDULER.DROP_SINGLE_JOB

  Deletes a single scheduled job. The operation can be performed only when the job has been synchronized.

  The prototype of DBE_SCHEDULER.DROP_SINGLE_JOB is as follows:

```
DBE_SCHEDULER.drop_single_job(
job_name text,
force boolean                   default false,
defer boolean                   default false
)
```

Table 2-283 DBE_SCHEDULER.DROP_SINGLE_JOB API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
job_name	TEXT	IN	No	Specifies the name of a scheduled job or scheduled job class.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
force	BOOLEAN	IN	No	Specifies how to delete a scheduled job. • **true**: The current scheduled job is stopped and then deleted. • **false**: The scheduled job fails to be deleted if it is running.
defer	BOOLEAN	IN	No	Specifies how to delete a scheduled job. • **true**: A scheduled job can be deleted after it is complete. • **false**: The scheduled job cannot be executed and is deleted.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_program('program1', 'STORED_PROCEDURE', 'SELECT
pg_sleep(1);', 0, false, 'test');
 create_program
----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.create_job('job1', 'program1', '2021-07-20', 'interval ''3 minute''',
'2121-07-20', 'DEFAULT_JOB_CLASS', false, false,'test', 'style', NULL, NULL);
 create_job
------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_single_job('job1', false, false);
 drop_single_job
-----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_program('program1', false);
 drop_program
--------------

(1 row)
```

- DBE_SCHEDULER.SET_ATTRIBUTE

  Modifies the attributes of a scheduled job. The operation can be performed only when the job has been synchronized.

  The prototypes of DBE_SCHEDULER.SET_ATTRIBUTE are as follows:

```
DBE_SCHEDULER.set_attribute(
name                 text,
attribute            text,
value               boolean
)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_SCHEDULER.set_attribute(
name                 text,
attribute            text,
value                datea
)

DBE_SCHEDULER.set_attribute(
name                 text,
attribute            text,
value                timestamp
)

DBE_SCHEDULER.set_attribute(
name                 text,
attribute            text,
value                timestamp with time zone
)

DBE_SCHEDULER.set_attribute(
name text,
attribute text,
value text,
value2 text                      default NULL
)
```

NOTE

- **name** specifies any object in DBE_SCHEDULER.
- The DBE_SCHEDULER.set_attribute function with **datea** as the input parameter can be used only when **mapping_date_to_datea** is set to **on**.

Table 2-284 DBE_SCHEDULER.SET_ATTRIBUTE API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
name	TEXT	IN	No	Object name.
attribute	TEXT	IN	No	Attribute name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input / Output Parameter	Can Be Null	Description
value	BOOLEAN/DATE/TIMESTAMP/TIMESTAMP WITH TIME ZONE/TEXT	IN	No	Attribute value. The options are as follows: ● Scheduled-job-related: **job_type**, **job_action**, **number_of_arguments**, **start_date**, **repeat_interval**, **end_date**, **job_class**, **enabled**, **auto_drop**, **comments**, **credential_name**, **destination_name**, **program_name**, **schedule_name**, and **job_style**. ● Program-related: **program_action**, **program_type**, **number_of_arguments**, and **comments**. ● Scheduling-related: **start_date**, **repeat_interval**, **end_date**, and **comments**.
value2	TEXT	IN	Yes	Additional attribute value. This is a reserved parameter bit. Currently, target attributes with extra attribute values are not supported.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_program('program1', 'STORED_PROCEDURE', 'SELECT
pg_sleep(1);', 3, false, 'test');
 create_program
----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.set_attribute('program1', 'number_of_arguments', 0);
 set_attribute
---------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.set_attribute('program1', 'program_type', 'STORED_PROCEDURE');
 set_attribute
---------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_program('program1', false);
 drop_program
--------------

(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- You are advised not to use DBE_SCHEDULER.SET_ATTRIBUTE to leave the parameters empty.
- The object name cannot be changed using DBE_SCHEDULER.SET_ATTRIBUTE.
- Inline objects cannot be changed by DBE_SCHEDULER.SET_ATTRIBUTE.

- DBE_SCHEDULER.RUN_JOB

  Runs a scheduled job.

  The prototype of DBE_SCHEDULER.RUN_JOB is as follows:

```
DBE_SCHEDULER.run_job(
job_name text,
use_current_session boolean          default true
)
```

NOTE

DBE_SCHEDULER.RUN_JOB is used to run scheduled jobs immediately. It is independent of the scheduling of scheduled jobs and can even run at the same time.

Table 2-285 DBE_SCHEDULER.RUN_JOB API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
job_name	TEXT	IN	No	One or more scheduled job names. If you specify multiple scheduled jobs, separate them with commas (,).
use_current_session	BOOLEAN	IN	No	Specifies whether to run a scheduled job. ● **true**: The current session is used to check whether the scheduled job can run properly. ● **false**: The scheduled job is started in the background. The execution result is recorded in logs.

Example:

```
gaussdb=# SELECT DBE_SCHEDULER.create_job('job1','PLSQL_BLOCK','BEGIN INSERT INTO test1
VALUES(12); end;',0,null,null,null,'DEFAULT_JOB_CLASS',false,false,null,null,null);
 create_job
------------


(1 row)

gaussdb=# CALL DBE_SCHEDULER.run_job('job1', false);
 run_job
```

```
---------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job('job1', true, false, 'STOP_ON_FIRST_ERROR');
 drop_job
----------

(1 row)
```

NOTE

Currently, **use_current_session** applies only to scheduled jobs whose **job_type** is set to **EXTERNAL_SCRIPT**.

- DBE_SCHEDULER.RUN_BACKEND_JOB

  Runs a scheduled job in the background.

  The prototype of DBE_SCHEDULER.RUN_BACKEND_JOB is as follows:

```
DBE_SCHEDULER.run_backend_job(
job_name text
)
```

Table 2-286 DBE_SCHEDULER.RUN_BACKEND_JOB API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
job_name	text	IN	No	Name of a scheduled job.

  Example:

```
gaussdb=# SELECT DBE_SCHEDULER.create_job('job1','PLSQL_BLOCK','BEGIN INSERT INTO test1
VALUES(12); end;',0,null,null,null,'DEFAULT_JOB_CLASS',false,false,null,null,null);
 create_job
------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.run_backend_job('job1');
 run_backend_job
-----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job('job1', true, false, 'STOP_ON_FIRST_ERROR');
 drop_job
----------

(1 row)
```

- DBE_SCHEDULER.RUN_FOREGROUND_JOB

  Runs a scheduled job in the current session. Only external jobs can be executed.

  Return value: text

  The prototype of DBE_SCHEDULER.RUN_FOREGROUND_JOB is as follows:

```
DBE_SCHEDULER.run_foreground_job(
job_name text
)return text
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-287 DBE_SCHEDULER.RUN_FOREGROUND_JOB API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
job_name	TEXT	IN	No	Name of a scheduled job.

Example:

```
gaussdb=# CREATE USER test1 IDENTIFIED BY '*********';
NOTICE:  The encrypted password contains MD5 ciphertext, which is not secure.
CREATE ROLE
gaussdb=# SELECT DBE_SCHEDULER.create_credential('cre_1', 'test1', '*********');
 create_credential
-------------------

(1 row)

gaussdb=# SELECT DBE_SCHEDULER.create_job(job_name=>'job1', job_type=>'EXTERNAL_SCRIPT',
job_action=>'/usr/bin/pwd', enabled=>true, auto_drop=>false, credential_name => 'cre_1');
 create_job
------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.run_foreground_job('job1');
  run_foreground_job
---------------------------------
 Host key verification failed.\r+

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job('job1', true, false, 'STOP_ON_FIRST_ERROR');
 drop_job
----------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_credential('cre_1', false);
 drop_credential
-----------------

(1 row)

gaussdb=# DROP USER test1;
DROP ROLE
```

- DBE_SCHEDULER.STOP_JOB

  Stops scheduled jobs.

  The prototype of DBE_SCHEDULER.STOP_JOB is as follows:

  ```
  DBE_SCHEDULER.stop_job(
  job_name text,
  force boolean                    default false,
  commit_semantics text                default 'STOP_ON_FIRST_ERROR'
  )
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-288 DBE_SCHEDULER.STOP_JOB API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
job_name	TEXT	IN	No	One or more scheduled job names or classes. If you specify multiple scheduled jobs, separate them with commas (,).
force	BOOLEAN	IN	No	Specifies how to delete a scheduled job. • **true**: The scheduler sends a termination signal to end the job thread immediately. • **false**: The scheduler attempts to use the interrupt signal to terminate the scheduled job thread.
commit_semantics	TEXT	IN	No	Commit rules: • **'STOP_ON_FIRST_ERROR'**: The interrupt operation performed before the first error is reported is committed. • **'ABSORB_ERRORS'**: The system attempts to bypass an error and commit the interrupt operation that is performed successfully.

Example:

```
gaussdb=# SELECT dbe_scheduler.create_job('job1','PLSQL_BLOCK','BEGIN INSERT INTO test1
VALUES(12); end;',0,null,null,null,'DEFAULT_JOB_CLASS',false,false,null,null,null);
 create_job
------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.stop_job('job1', true, 'STOP_ON_FIRST_ERROR');
 stop_job
----------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job('job1', true, false, 'STOP_ON_FIRST_ERROR');
 drop_job
----------

(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_SCHEDULER.STOP_SINGLE_JOB

  Stops a single scheduled job.

  The prototype of DBE_SCHEDULER.STOP_SINGLE_JOB is as follows:

```
DBE_SCHEDULER.stop_single_job(
job_name text,
force boolean                    default false
)
```

Table 2-289 DBE_SCHEDULER.STOP_SINGLE_JOB API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
job_name	TEXT	IN	No	Name of a scheduled job or scheduled job class.
force	BOOLEAN	IN	No	Specifies how to delete a scheduled job. ● **true**: The scheduler sends a termination signal to end the job thread immediately. ● **false**: The scheduler attempts to use the interrupt signal to terminate the scheduled job thread.

Example:

```
gaussdb=# SELECT dbe_scheduler.create_job('job1','PLSQL_BLOCK','BEGIN INSERT INTO test1
VALUES(12); end;',0,null,null,null,'DEFAULT_JOB_CLASS',false,false,null,null,null);
 create_job
------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.stop_single_job('job1', true);
 stop_single_job
-----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job('job1', true, false, 'STOP_ON_FIRST_ERROR');
 drop_job
----------

(1 row)
```

- DBE_SCHEDULER.GENERATE_JOB_NAME

  Generates a scheduled job name.

  The prototype of DBE_SCHEDULER.GENERATE_JOB_NAME is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_SCHEDULER.generate_job_name(
prefix text                    default 'JOB$_'
)return text
```

Table 2-290 DBE_SCHEDULER.GENERATE_JOB_NAME API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
prefix	TEXT	IN	No	Prefix of the generated name. The default value is **'JOB$_'**. Scheduled jobs that are repeatedly executed are named as follows: job$_1, job$_2, job$_3 ...

Example:

```
gaussdb=# CALL DBE_SCHEDULER.generate_job_name();
 generate_job_name
-------------------
 JOB$_1
(1 row)

gaussdb=# CALL DBE_SCHEDULER.generate_job_name();
 generate_job_name
-------------------
 JOB$_2
(1 row)

gaussdb=# CALL DBE_SCHEDULER.generate_job_name('job');
 generate_job_name
-------------------
 job3
(1 row)

gaussdb=# CALL DBE_SCHEDULER.generate_job_name('job');
 generate_job_name
-------------------
 job4
(1 row)
```

NOTE

When DBE_SCHEDULER.GENERATE_JOB_NAME is executed for the first time, a temporary sequence is created in **public** to store the sequence number of the current name. Common users do not have the CREATE permission on **public**. Therefore, if a common user calls the function for the first time in the current database, the function fails to be called. In this case, you need to grant the CREATE permission on **public** to the common user or call the API as a user with the CREATE permission to create a temporary sequence.

- DBE_SCHEDULER.CREATE_PROGRAM

  Creates a program and synchronizes it to all CNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The prototype of DBE_SCHEDULER.CREATE_PROGRAM is as follows:

```
DBE_SCHEDULER.create_program(
program_name text,
program_type text,
program_action text,
number_of_arguments integer          default 0,
enabled boolean                    default false,
comments text                      default NULL
)
```

Table 2-291 DBE_SCHEDULER.CREATE_PROGRAM API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
program_name	TEXT	IN	No	Program name.
program_type	TEXT	IN	No	Program type. The options are as follows: • **'PLSQL_BLOCK'**: fast anonymous stored procedure. • **'STORED_PROCEDURE'**: stored procedure that is saved. • **'EXTERNAL_SCRIPT'**: external script.
program_action	TEXT	IN	No	Program operation.
number_of_arguments	INTEGER	IN	No	Number of program parameters. The value ranges from 0 to 255.
enabled	BOOLEAN	IN	No	Specifies whether the program is enabled.
comments	TEXT	IN	Yes	Comments.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_program('program1', 'STORED_PROCEDURE', 'SELECT
pg_sleep(1);', 3, false, 'test');
 create_program
----------------


(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_program('program1', false);
 drop_program
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------------

(1 row)
```

- DBE_SCHEDULER.DEFINE_PROGRAM_ARGUMENT

  Defines program parameters. The operation can be performed only when the program has been synchronized. An API with the default value **default_value** will not convert characters to lowercase letters by default. In this version, characters are case-sensitive.

  The prototype of DBE_SCHEDULER.DEFINE_PROGRAM_ARGUMENT is as follows:

```
DBE_SCHEDULER.define_program_argument(
program_name text,
argument_position integer,
argument_name text                  default NULL,
argument_type text,
out_argument boolean                 default false
)

-- With a default value --
DBE_SCHEDULER.define_program_argument(
program_name text,
argument_position integer,
argument_name text                  default NULL,
argument_type text,
default_value text,
out_argument boolean                 default false
)
```

Table 2-292 DBE_SCHEDULER.DEFINE_PROGRAM_ARGUMENT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
program_name	TEXT	IN	No	Program name.
argument_position	INTEGER	IN	No	Parameter position. The value ranges from 1 to *number_of_arguments*. If the value of **number_of_arguments** of a program is **0**, an error is reported when this API is called.
argument_name	TEXT	IN	No	Parameter name.
argument_type	TEXT	IN	No	Parameter type.
default_value	TEXT	IN	No	Default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
out_argument	BOOLEAN	IN	No	Reserved parameter.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_program('program1', 'STORED_PROCEDURE', 'SELECT
pg_sleep(1);', 2, false, 'test');
create_program
----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.define_program_argument('program1', 1, 'pa1', 'type1', false);
 define_program_argument
-------------------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.define_program_argument('program1', 1, 'pa1', 'type1', 'value1',
false);
 define_program_argument
-------------------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_program('program1', false);
 drop_program
--------------

(1 row)
```

- DBE_SCHEDULER.DROP_PROGRAM

  Deletes a program. The operation can be performed only when the program has been synchronized.

  The prototype of DBE_SCHEDULER.DROP_PROGRAM is as follows:

  ```
  DBE_SCHEDULER.drop_program(
  program_name text,
  force boolean                    default false
  )
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-293 DBE_SCHEDULER.DROP_PROGRAM API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
program_name	TEXT	IN	No	Program name.
force	BOOLEAN	IN	No	Specifies how to delete a program. • **true**: Before a program is deleted, all jobs that use the program are disabled. • **false**: The program cannot be referenced by any job. Otherwise, an error is sent.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_program('program1', 'STORED_PROCEDURE', 'SELECT
pg_sleep(1);', 2, false, 'test');
create_program
----------------


(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_program('program1', false);
 drop_program
--------------


(1 row)
```

- DBE_SCHEDULER.DROP_SINGLE_PROGRAM

  Deletes a single program. The operation can be performed only when the program has been synchronized.

  The prototype of DBE_SCHEDULER.DROP_SINGLE_PROGRAM is as follows:

  ```
  DBE_SCHEDULER.drop_single_program(
  program_name text,
  force boolean                    default false
  )
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-294 DBE_SCHEDULER.DROP_SINGLE_PROGRAM API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
program_name	TEXT	IN	No	Program name.
force	BOOLEAN	IN	No	Specifies how to delete a program. • **true**: Before a program is deleted, all jobs that use the program are disabled. • **false**: The program cannot be referenced by any job. Otherwise, an error is sent.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_program('program1', 'STORED_PROCEDURE', 'SELECT
pg_sleep(1);', 2, false, 'test');
create_program
----------------


(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_single_program('program1', false);
 drop_single_program
---------------------


(1 row)
```

- DBE_SCHEDULER.SET_JOB_ARGUMENT_VALUE

  Sets scheduled job parameters. The **argument_value** parameter can be left empty. The operation can be performed only when the job has been synchronized.

  The prototype of DBE_SCHEDULER.SET_JOB_ARGUMENT_VALUE is as follows:

```
DBE_SCHEDULER.set_job_argument_value(
job_name text,
argument_position integer,
argument_value text
)

DBE_SCHEDULER.set_job_argument_value(
job_name text,
argument_name text,
argument_value text
)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-295 DBE_SCHEDULER.SET_JOB_ARGUMENT_VALUE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
job_name	TEXT	IN	No	Name of a scheduled job.
argument_position	INTEGER	IN	No	Parameter position. The value ranges from 1 to *number_of_arguments*. If the value of **number_of_arguments** of an associated program is **0**, an error is reported when this API is called.
argument_name	TEXT	IN	Yes	Parameter name.
argument_value	TEXT	IN	Yes	Parameter value.

Example:

```
gaussdb=# CALL dbe_scheduler.create_job('job1','EXTERNAL_SCRIPT','BEGIN INSERT INTO test1
VALUES(12); end;',2,null,null,null,'DEFAULT_JOB_CLASS',false,false,null,null,null);
 create_job
------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.set_job_argument_value('job1', 1, 'value1');
 set_job_argument_value
------------------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job('job1', true, false, 'STOP_ON_FIRST_ERROR');
 drop_job
----------

(1 row)
```

- DBE_SCHEDULER.CREATE_SCHEDULE

  Creates a schedule. Then, it is synchronized to all CNs.

  The prototype of DBE_SCHEDULER.CREATE_SCHEDULE is as follows:

```
DBE_SCHEDULER.create_schedule(
schedule_name text,
start_date timestamp with time zone     default NULL,
repeat_interval text,
end_date timestamp with time zone       default NULL,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
comments text                   default NULL
)
```

Table 2-296 DBE_SCHEDULER.CREATE_SCHEDULE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
schedule_name	TEXT	IN	No	Schedule name.
start_date	TIMESTAMP WITH TIME ZONE	IN	Yes	Start time of a schedule.
repeat_interval	TEXT	IN	No	Repetition frequency of a schedule.
end_date	TIMESTAMP WITH TIME ZONE	IN	Yes	End time of a schedule.
comments	TEXT	IN	Yes	Comments.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_schedule('schedule1', sysdate, 'sysdate + 3 / (24 * 60 * 60)',
null, 'test1');
 create_schedule
-----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.create_schedule('schedule2', sysdate, 'FREQ=DAILY; BYHOUR=6;',
null, 'test1');
 create_schedule
-----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.create_schedule('schedule3', sysdate, 'FREQ=DAILY; BYHOUR=6;');
 create_schedule
-----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_schedule('schedule1');
 drop_single_schedule
----------------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_schedule('schedule2', false);
 drop_single_schedule
----------------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_schedule('schedule3', true);
 drop_single_schedule
----------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)
```

- DBE_SCHEDULER.DROP_SCHEDULE

  Deletes a schedule. The operation can be performed only when the schedule has been synchronized.

  The prototype of DBE_SCHEDULER.DROP_SCHEDULE is as follows:

```
DBE_SCHEDULER.drop_schedule(
schedule_name text,
force boolean                    default false
)
```

Table 2-297 DBE_SCHEDULER.DROP_SCHEDULE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
schedule_name	TEXT	IN	No	Schedule name.
force	BOOLEAN	IN	No	Specifies how to delete a schedule. • **true**: Any jobs or windows that use this schedule are disabled before the schedule is deleted. • **false**: The schedule cannot be referenced by any job or window. Otherwise, an error occurs.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_schedule('schedule1', sysdate, 'sysdate + 3 / (24 * 60 * 60)',
null, 'test1');
 create_schedule
-----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.create_schedule('schedule2', sysdate, 'FREQ=DAILY; BYHOUR=6;',
null, 'test1');
 create_schedule
-----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.create_schedule('schedule3', sysdate, 'FREQ=DAILY; BYHOUR=6;');
 create_schedule
-----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_schedule('schedule1');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 drop_single_schedule
----------------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_schedule('schedule2', false);
 drop_single_schedule
----------------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_schedule('schedule3', true);
 drop_single_schedule
----------------------

(1 row)
```

- DBE_SCHEDULER.DROP_SINGLE_SCHEDULE

  Deletes a single schedule. The operation can be performed only when the schedule has been synchronized.

  The prototype of DBE_SCHEDULER.DROP_SINGLE_SCHEDULE is as follows:

```
DBE_SCHEDULER.drop_single_schedule(
schedule_name text,
force boolean                   default false
)
```

Table 2-298 DBE_SCHEDULER.DROP_SINGLE_SCHEDULE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
schedule_name	TEXT	IN	No	Schedule name.
force	BOOLEAN	IN	No	Specifies how to delete a schedule. • **true**: Any jobs or windows that use this schedule are disabled before the schedule is deleted. • **false**: The schedule cannot be referenced by any job or window. Otherwise, an error occurs.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_schedule('schedule1', sysdate, 'sysdate + 3 / (24 * 60 * 60)',
null, 'test1');
 create_schedule
-----------------

(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CALL DBE_SCHEDULER.create_schedule('schedule2', sysdate, 'FREQ=DAILY; BYHOUR=6;',
null, 'test1');
 create_schedule
-----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.create_schedule('schedule3', sysdate, 'FREQ=DAILY; BYHOUR=6;');
 create_schedule
-----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_single_schedule('schedule1');
 drop_single_schedule
----------------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_single_schedule('schedule2', false);
 drop_single_schedule
----------------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_single_schedule('schedule3', true);
 drop_single_schedule
----------------------

(1 row)
```

- DBE_SCHEDULER.CREATE_JOB_CLASS

  Creates a scheduled job class. Then, it is synchronized to all CNs.

  The prototype of DBE_SCHEDULER.CREATE_JOB_CLASS is as follows:

```
DBE_SCHEDULER.create_job_class(
job_class_name text,
resource_consumer_group text           default NULL,
service text                        default NULL,
logging_level integer                  default 0,
log_history integer                   default NULL,
comments text                          default NULL
)
```

Table 2-299 DBE_SCHEDULER.CREATE_JOB_CLASS API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
job_class_name	TEXT	IN	No	Name of a scheduled job class.
resource_consumer_group	TEXT	IN	Yes	Inline resource consumer group of a scheduled job class.
service	TEXT	IN	Yes	Inline database service of a scheduled job class.
logging_level	INTEGER	IN	No	Number of scheduled job records.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
log_history	INTEGER	IN	Yes	Number of days for storing scheduled job records.
comments	TEXT	IN	Yes	Comments.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_job_class(job_class_name => 'jc1', resource_consumer_group
=> '123');
 create_job_class
------------------


(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job_class('jc1', false);
 drop_job_class
----------------


(1 row)
```

- DBE_SCHEDULER.DROP_JOB_CLASS

  Deletes scheduled job classes. The operation can be performed only when the scheduled job class has been synchronized.

  The prototype of DBE_SCHEDULER.DROP_JOB_CLASS is as follows:

  ```
  DBE_SCHEDULER.drop_job_class(
  job_class_name text,
  force boolean                    default false
  )
  ```

  Table 2-300 DBE_SCHEDULER.DROP_JOB_CLASS API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
job_class_name	TEXT	IN	No	Name of a scheduled job class.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
force	BOOLEAN	IN	No	Specifies how to delete a scheduled job class. ● **true**: Jobs of this class will be disabled, and another class will be set as the default class. Only when such setting is successful, this class can be deleted. ● **false**: The class to be deleted cannot be referenced by any job. Otherwise, an error occurs.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_job_class(job_class_name => 'jc1', resource_consumer_group
=> '123');
 create_job_class
------------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job_class('jc1', false);
 drop_job_class
----------------

(1 row)
```

- DBE_SCHEDULER.DROP_SINGLE_JOB_CLASS

  Deletes a single scheduled job class. The operation can be performed only when the scheduled job class has been synchronized.

  The prototype of DBE_SCHEDULER.DROP_SINGLE_JOB_CLASS is as follows:

```
DBE_SCHEDULER.drop_single_job_class(
job_class_name text,
force boolean                   default false
)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-301 DBE_SCHEDULER.DROP_SINGLE_JOB_CLASS API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
job_class_name	TEXT	IN	No	Name of a scheduled job class.
force	BOOLEAN	IN	No	Specifies how to delete a scheduled job class. • **true**: Jobs of this class will be disabled, and another class will be set as the default class. Only when such setting is successful, this class can be deleted. • **false**: The class to be deleted cannot be referenced by any job. Otherwise, an error occurs.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_job_class(job_class_name => 'jc1', resource_consumer_group
=> '123');
 create_job_class
------------------


(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_single_job_class('jc1', false);
 drop_single_job_class
-----------------------


(1 row)
```

- DBE_SCHEDULER.GRANT_USER_AUTHORIZATION

  Grants the scheduled job permissions to a database user. The user who calls this function must have the SYSADMIN permission. Then, it is synchronized to all CNs.

  The prototype of DBE_SCHEDULER.GRANT_USER_AUTHORIZATION is as follows:

  ```
  DBE_SCHEDULER.grant_user_authorization(
  username              text,
  privilege             text
  )
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-302 DBE_SCHEDULER.GRANT_USER_AUTHORIZATION API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
username	TEXT	IN	No	Database username.
privilege	TEXT	IN	No	Scheduled job permission.

Example:

```
gaussdb=# CREATE USER user1 PASSWORD '1*s*****';
NOTICE:  The encrypted password contains MD5 ciphertext, which is not secure.
CREATE ROLE
gaussdb=# CALL DBE_SCHEDULER.grant_user_authorization('user1', 'create job');
 grant_user_authorization
--------------------------


(1 row)
gaussdb=# DROP USER user1;
DROP ROLE
```

- DBE_SCHEDULER.REVOKE_USER_AUTHORIZATION

  Revokes the scheduled job permissions from a database user. The user who calls this function must have the SYSADMIN permission. The operation can be performed only when the permission has been granted.

  The prototype of DBE_SCHEDULER.REVOKE_USER_AUTHORIZATION is as follows:

```
DBE_SCHEDULER.revoke_user_authorization(
username              text,
privilege             text
)
```

Table 2-303 DBE_SCHEDULER.REVOKE_USER_AUTHORIZATION API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
username	TEXT	IN	No	Database username.
privilege	TEXT	IN	No	Scheduled job permission.

Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CREATE USER user1 PASSWORD '1*s*****';
NOTICE:  The encrypted password contains MD5 ciphertext, which is not secure.
CREATE ROLE
gaussdb=# CALL DBE_SCHEDULER.grant_user_authorization('user1', 'create job');
 grant_user_authorization
--------------------------


(1 row)

gaussdb=# CALL DBE_SCHEDULER.revoke_user_authorization('user1', 'create job');
 revoke_user_authorization
---------------------------


(1 row)
gaussdb=# DROP USER user1;
DROP ROLE
```

- DBE_SCHEDULER.CREATE_CREDENTIAL

  Creates an authorization certificate. The user who calls this function must have the SYSADMIN permission. Then, it is synchronized to all CNs.

  The prototype of DBE_SCHEDULER.CREATE_CREDENTIAL is as follows:

```
DBE_SCHEDULER.create_credential(
credential_name         text,
username                text,
password                text            default NULL,
database_role           text            default NULL,
windows_domain          text            default NULL,
comments                text            default NULL
)
```

Table 2-304 DBE_SCHEDULER.CREATE_CREDENTIAL API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
credential_name	TEXT	IN	No	Name of the authorization certificate.
username	TEXT	IN	No	Database username.
password	TEXT	IN	Yes	User password.
database_role	TEXT	IN	Yes	Database system permission.
windows_domain	TEXT	IN	Yes	Domain to which a Windows user belongs.
comments	TEXT	IN	Yes	Comments.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_credential('cre_1', 'user1', '');
 create_credential
-------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_credential('cre_1', false);
 drop_credential
-----------------

(1 row)
```

NOTE

The **password** parameter of DBE_SCHEDULER.CREATE_CREDENTIAL must be set to **NULL** or **'******'**. This parameter is used only for compatibility processing and does not indicate any actual meaning. Do not use the OS username corresponding to the installation user to create a certificate.

- DBE_SCHEDULER.DROP_CREDENTIAL

  Destroys an authorization certificate. The user who calls this function must have the SYSADMIN permission. The operation can be performed only when the certificate has been synchronized.

  The prototype of DBE_SCHEDULER.DROP_CREDENTIAL is as follows:

```
DBE_SCHEDULER.drop_credential(
credential_name         text,
force boolean default false
)
```

Table 2-305 DBE_SCHEDULER.DROP_CREDENTIAL API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
credential_name	TEXT	IN	No	Name of the authorization certificate.
force	BOOLEAN	IN	No	Specifies how to delete the authorization certificate. • **true**: The certificate will be deleted no matter whether it is referenced by any job. • **false**: No job can reference the certificate. Otherwise, an error occurs.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.create_credential('cre_1', 'user1', '');
 create_credential
-------------------

(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CALL DBE_SCHEDULER.drop_credential('cre_1', false);
 drop_credential
-----------------

(1 row)
```

- DBE_SCHEDULER.ENABLE

  Enables an object. Then, it is synchronized to all CNs.

  The prototype of DBE_SCHEDULER.ENABLE is as follows:

```
DBE_SCHEDULER.enable(
name text,
commit_semantics text                default 'STOP_ON_FIRST_ERROR'
)
```

Table 2-306 DBE_SCHEDULER.ENABLE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
name	TEXT	IN	No	Object name. You can specify one or more objects. If you specify multiple objects, separate them with commas (,).
commit_semantics	TEXT	IN	No	Commit rules. The following types are supported: ● **'STOP_ON_FIRST_ERROR'**: The enabling operation performed before the first error is reported is committed. ● **'TRANSACTIONAL'**: Transaction-level commit. The enabling operation performed before an error is reported will be rolled back. ● **'ABSORB_ERRORS'**: The system attempts to bypass an error and commit the enabling operation that is performed successfully.

Example:

```
gaussdb=# CALL dbe_scheduler.create_job('job1','PLSQL_BLOCK','BEGIN INSERT INTO test1
VALUES(12); end;',0,null,null,null,'DEFAULT_JOB_CLASS',false,false,null,null,null);
 create_job
------------

(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CALL DBE_SCHEDULER.create_program('program1', 'stored_procedure', 'INSERT INTO
tb_job_test(key) VALUES(null);', 0, false, '');
 create_program
----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.enable('job1');
 enable
--------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.enable('program1', 'STOP_ON_FIRST_ERROR');
 enable
--------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job('job1', true, false, 'STOP_ON_FIRST_ERROR');
 drop_job
----------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_program('program1', false);
 drop_program
--------------

(1 row)
```

- DBE_SCHEDULER.ENABLE_SINGLE

  Enables a single object. Then, it is synchronized to all CNs.

  The prototype of DBE_SCHEDULER.ENABLE_SINGLE is as follows:

  ```
  DBE_SCHEDULER.enable_single(
  name text
  )
  ```

  Table 2-307 DBE_SCHEDULER.ENABLE_SINGLE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
name	TEXT	IN	No	Object name.

  Example:

  ```
  gaussdb=# CALL dbe_scheduler.create_job('job1','PLSQL_BLOCK','BEGIN INSERT INTO test1
  VALUES(12); end;',0,null,null,null,'DEFAULT_JOB_CLASS',false,false,null,null,null);
   create_job
  ------------

  (1 row)

  gaussdb=# CALL DBE_SCHEDULER.enable_single('job1');
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 enable_single
---------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job('job1', true, false, 'STOP_ON_FIRST_ERROR');
 drop_job
----------

(1 row)
```

- DBE_SCHEDULER.DISABLE

  Disables multiple objects. The value of **name** is a character string delimited by commas (,). Each part of the string delimited by commas (,) is an object. This operation can be performed only when enabling has been performed.

  The prototype of DBE_SCHEDULER.DISABLE is as follows:

```
DBE_SCHEDULER.disable(
name text,
force boolean                   default false,
commit_semantics text               default 'STOP_ON_FIRST_ERROR'
)
```

Table 2-308 DBE_SCHEDULER.DISABLE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
name	TEXT	IN	No	Object name.
force	BOOLEAN	IN	No	Specifies how to disable an object. • **true**: The object is disabled regardless of whether other objects depend on it. • **false**: No object can depend on the object. Otherwise, an error occurs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
commit_semantics	TEXT	IN	No	Commit rules. The following types are supported: ● **'STOP_ON_FIRST_ERROR'**: The disabling operation performed before the first error is reported is committed. ● **'TRANSACTIONAL'**: Transaction-level commit. The disabling operation performed before an error is reported will be rolled back. ● **'ABSORB_ERRORS'**: The system attempts to bypass an error and commit the disabling operation that is performed successfully.

Example:

```
gaussdb=# CALL dbe_scheduler.create_job('job1','PLSQL_BLOCK','BEGIN INSERT INTO test1
VALUES(12); end;',0,null,null,null,'DEFAULT_JOB_CLASS',false,false,null,null,null);
 create_job
------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.create_program('program1', 'stored_procedure', 'INSERT INTO
tb_job_test(key) VALUES(null);', 0, false, '');
 create_program
----------------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.disable('job1');
 disable
---------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.disable('program1', false, 'STOP_ON_FIRST_ERROR');
 disable
---------

(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job('job1', true, false, 'STOP_ON_FIRST_ERROR');
 drop_job
----------

(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CALL DBE_SCHEDULER.drop_program('program1', false);
 drop_program
--------------


(1 row)
```

- DBE_SCHEDULER.DISABLE_SINGLE

  Disables a single object. This operation can be performed only when enabling has been performed.

  The prototype of DBE_SCHEDULER.DISABLE_SINGLE is as follows:

```
DBE_SCHEDULER.disable_single(
name text,
force boolean                    default false
)
```

Table 2-309 DBE_SCHEDULER.DISABLE_SINGLE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
name	TEXT	IN	No	Object name.
force	BOOLEAN	IN	No	Specifies how to disable an object. • **true**: The object is disabled regardless of whether other objects depend on it. • **false**: No object can depend on the object. Otherwise, an error occurs.

Example:

```
gaussdb=# CALL dbe_scheduler.create_job('job1','PLSQL_BLOCK','BEGIN INSERT INTO test1
VALUES(12); end;',0,null,null,null,'DEFAULT_JOB_CLASS',false,false,null,null,null);
 create_job
------------


(1 row)

gaussdb=# CALL DBE_SCHEDULER.disable_single('job1', false);
 disable_single
----------------


(1 row)

gaussdb=# CALL DBE_SCHEDULER.drop_job('job1', true, false, 'STOP_ON_FIRST_ERROR');
 drop_job
----------


(1 row)
```

- DBE_SCHEDULER.EVAL_CALENDAR_STRING

  Analyzes the scheduled job period.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Return type: timestamp with time zone

The prototype of DBE_SCHEDULER.EVAL_CALENDAR_STRING is as follows:

```
DBE_SCHEDULER.evaluate_calendar_string(
IN calendar_string text,
IN start_date timestamp with time zone,
IN return_date_after timestamp with time zone
)return timestamp with time zone
```

Table 2-310 DBE_SCHEDULER.EVAL_CALENDAR_STRING API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
calendar_string	TEXT	IN	No	Date string of a scheduled job.
start_date	TIMESTAMP WITH TIME ZONE	IN	No	Start time of a scheduled job.
return_date_after	TIMESTAMP WITH TIME ZONE	IN	No	Date returned by a scheduled job, which is used to filter the first matching scheduling time that comes after this date.

Example:

```
gaussdb=# CALL DBE_SCHEDULER.eval_calendar_string('FREQ=DAILY; BYHOUR=6;', sysdate, sysdate);
  eval_calendar_string
------------------------
 2023-09-15 06:47:24+08
(1 row)
```

- DBE_SCHEDULER.EVALUATE_CALENDAR_STRING

  Analyzes the scheduled job period.

  Return type: timestamp with time zone

  The prototype of DBE_SCHEDULER.EVALUATE_CALENDAR_STRING is as follows:

  ```
  DBE_SCHEDULER.evaluate_calendar_string(
  IN calendar_string text,
  IN start_date timestamp with time zone,
  IN return_date_after timestamp with time zone,
  OUT next_run_date timestamp with time zone
  )return timestamp with time zone
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-311 DBE_SCHEDULER.EVALUATE_CALENDAR_STRING API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
calendar_string	TEXT	IN	No	Date string of a scheduled job.
start_date	TIMESTAMP WITH TIME ZONE	IN	No	Start time of a scheduled job.
return_date_after	TIMESTAMP WITH TIME ZONE	IN	No	Date returned by a scheduled job, which is used to filter the first matching scheduling time that comes after this date.
next_run_date	TIMESTAMP WITH TIME ZONE	OUT	No	Next date when a scheduled job is returned.

Example:

```
gaussdb=# CREATE OR REPLACE PROCEDURE pr1(calendar_str text) as
DECLARE
    start_date        timestamp with time zone;
    return_date_after timestamp with time zone;
    next_run_date     timestamp with time zone;
BEGIN
    start_date := '2003-2-1 10:30:00.111111+8'::timestamp with time zone;
    return_date_after := start_date;
    DBE_SCHEDULER.evaluate_calendar_string(
        calendar_str,
        start_date, return_date_after, next_run_date);
    DBE_OUTPUT.PRINT_LINE('next_run_date: ' || next_run_date);
    return_date_after := next_run_date;
END;
/
CREATE PROCEDURE

gaussdb=# CALL pr1('FREQ=hourly;INTERVAL=2;BYHOUR=6,10;BYMINUTE=0;BYSECOND=0');
next_run_date: 2003-02-02 06:00:00+08
 pr1
-----

(1 row)
```

2.12.2.16 DBE_SESSION

APIs

Table 2-312 lists APIs supported by the DBE_SESSION package. DBE_SESSION takes effect at the session level.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-312 DBE_SESSION

API	Description
DBE_SESSION.SET_CONTEXT	Sets the value of an attribute in a specified context.
DBE_SESSION.CLEAR_CONTEXT	Clears the value of an attribute in a specified context.
DBE_SESSION.SEARCH_CONTEXT	Queries the value of an attribute in a specified context.
DBE_SESSION.MODIFY_PACKAGE_STATE	Changes the PL/SQL status of the current session.

- DBE_SESSION.SET_CONTEXT

  Sets the value of an attribute in a specified namespace (context).

  The prototype of DBE_SESSION.SET_CONTEXT is as follows:

```
DBE_SESSION.SET_CONTEXT(
    namespace text,
    attribute text,
    value text
)returns void;
```

  Table 2-313 DBE_SESSION.SET_CONTEXT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
namespace	TEXT	IN	No	Name of the context to be set. If the context does not exist, create a context. The value contains a maximum of 128 bytes. If the value exceeds 128 bytes, it will be truncated.
attribute	TEXT	IN	No	Attribute name. The value contains a maximum of 1024 bytes. If the value exceeds 1024 bytes, it will be truncated.
value	TEXT	IN	No	Name of the value to be set. The value contains a maximum of 1024 bytes. If the value exceeds 1024 bytes, it will be truncated.

- DBE_SESSION.CLEAR_CONTEXT

  Clears the value of an attribute in a specified namespace (context).

  The prototype of DBE_SESSION.CLEAR_CONTEXT is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_SESSION.CLEAR_CONTEXT (
   namespace text,
   client_identifier text default null,
   attribute text default null
)returns void ;
```

Table 2-314 DBE_SESSION.CLEAR_CONTEXT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
namespace	TEXT	IN	No	Context specified by the user. The value contains a maximum of 128 bytes. If the value exceeds 128 bytes, it will be truncated.
client_identifier	TEXT	IN	Yes	Client authentication. The default value is **null**. Generally, you do not need to manually set this parameter.
attribute	TEXT	IN	Yes	Attribute to be cleared. The default value is **null**, indicating that all attributes of the specified context are cleared. The value contains a maximum of 1024 bytes. If the value exceeds 1024 bytes, it will be truncated. **NOTE** To ensure compatibility with earlier versions, if the parameter value is **'null'**, all attributes of the specified context are cleared.

- DBE_SESSION.SEARCH_CONTEXT

  Queries the value of an attribute in a specified namespace (context).

  The prototype of DBE_SESSION.SEARCH_CONTEXT is as follows:

```
DBE_SESSION.SEARCH_CONTEXT (
   namespace text,
   attribute text
)returns text;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-315 DBE_SESSION.SEARCH_CONTEXT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
namespace	TEXT	IN	No	Context specified by the user. The value contains a maximum of 128 bytes. If the value exceeds 128 bytes, it will be truncated.
attribute	TEXT	IN	No	Attribute to be searched for. The value contains a maximum of 1024 bytes. If the value exceeds 1024 bytes, it will be truncated.

Example:

```
DECLARE
    a text;
BEGIN
    DBE_SESSION.set_context('test', 'gaussdb', 'one');   -- Set the gaussdb attribute in the test context
to one.
    a := DBE_SESSION.search_context('test', 'gaussdb');
    DBE_OUTPUT.PRINT_LINE(a);
    DBE_SESSION.clear_context('test', 'test','gaussdb');
END;
/
-- Expected result:
one
ANONYMOUS BLOCK EXECUTE
```

- DBE_SESSION.MODIFY_PACKAGE_STATE

  Changes the PL/SQL status of the current session.

  The prototype of DBE_SESSION.MODIFY_PACKAGE_STATE is as follows:

  ```
  DBE_SESSION.MODIFY_PACKAGE_STATE (
      flags int);
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-316 DBE_SESSION.MODIFY_PACKAGE_STATE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
flags	INT	IN	No	Bit flag of the operation performed on the PL/SQL. When **flags** is set to **1**, the session cache of the PL/SQL that is currently running in the session is released after the top-level PL/SQL code is executed. The current values of global variables of all packages are cleared and the cached cursors are closed. On subsequent use, PL/SQL re-instantiates and re-initializes the global variables of the packages. Other bit flags are not supported.

Example:

```
CALL dbe_session.modify_package_state(1);
 modify_package_state
----------------------

(1 row)
```

2.12.2.17 DBE_SQL

Data Types

- DBE_SQL.DESC_REC

  This type is a composite type and is used to store the description of the DBE_SQL.SQL_DESCRIBE_COLUMNS API.

  The prototype of the DBE_SQL.DESC_REC type is as follows:

```
CREATE TYPE DBE_SQL.DESC_REC AS (
    col_type            int,
    col_max_len         int,
    col_name            VARCHAR2(32),
    col_name_len        int,
    col_schema_name     VARCHAR2(32),
    col_schema_name_len int,
    col_precision       int,
    col_scale           int,
    col_charsetid       int,
    col_charsetform     int,
    col_null_ok         BOOLEAN
);
```

- DBE_SQL.DESC_TAB

  This type is the TABLE type of DESC_REC and is implemented through the TABLE OF syntax.

  The prototype of the DBE_SQL.DESC_TAB type is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
CREATE TYPE DBE_SQL.DESC_TAB AS TABLE OF DBE_SQL.DESC_REC;
```

- DBE_SQL.DATE_TABLE

  This type is the TABLE type of DATE and is implemented through the TABLE OF syntax.

  The prototype of the DBE_SQL.DATE_TABLE type is as follows:

  ```
  CREATE TYPE DBE_SQL.DATE_TABLE AS TABLE OF DATE;
  ```

  NOTE

  - When an advanced package is created, if the GUC parameter **mapping_date_to_datea** is set to **on**, DBE_SQL.DATE_TABLE is mapped to the TABLE type of DATEA.
  - When the EXECUTE DIRECT ON syntax is used to execute functions in the DBE_SQL package, if the GUC parameter **enable_udf_execute_direct** is set to **on**, the functions can be executed; if the parameter is set to **off**, the functions cannot be executed.

- DBE_SQL.NUMBER_TABLE

  This type is the TABLE type of NUMBER and is implemented through the TABLE OF syntax.

  The prototype of the DBE_SQL.NUMBER_TABLE type is as follows:

  ```
  CREATE TYPE DBE_SQL.NUMBER_TABLE AS TABLE OF NUMBER;
  ```

- DBE_SQL.VARCHAR2_TABLE

  This type is the TABLE type of VARCHAR2 and is implemented through the TABLE OF syntax.

  The prototype of the DBE_SQL.VARCHAR2_TABLE type is as follows:

  ```
  CREATE TYPE DBE_SQL.VARCHAR2_TABLE AS TABLE OF VARCHAR2(32767);
  ```

- DBE_SQL.BLOB_TABLE

  This type is the TABLE type of BLOB and is implemented through the TABLE OF syntax.

  The prototype of the DBE_SQL.BLOB_TABLE type is as follows:

  ```
  CREATE TYPE DBE_SQL.BLOB_TABLE AS TABLE OF BLOB;
  ```

APIs

Table 2-317 lists APIs supported by the DBE_SQL package.

Table 2-317 DBE_SQL

API	Description
DBE_SQL.REGISTER_CONTEXT	Opens a cursor.
DBE_SQL.SQL_UNREGISTER_CONTEXT	Closes an open cursor.
DBE_SQL.SQL_SET_SQL	Passes a set of SQL statements or anonymous blocks to a cursor.
DBE_SQL.SQL_RUN	Executes SQL statements or anonymous blocks in a given cursor.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_SQL.NEXT_ROW	Reads a row of cursor data.
DBE_SQL.SET_RESULT_TYPE	Dynamically defines a column.
DBE_SQL.SET_RESULT_TYPE_CHAR	Dynamically defines a column of the CHAR type.
DBE_SQL.SET_RESULT_TYPE_INT	Dynamically defines a column of the INT type.
DBE_SQL.SET_RESULT_TYPE_LONG	Dynamically defines a column of the LONG type.
DBE_SQL.SET_RESULT_TYPE_RAW	Dynamically defines a column of the RAW type.
DBE_SQL.SET_RESULT_TYPE_TEXT	Dynamically defines a column of the TEXT type.
DBE_SQL.SET_RESULT_TYPE_UNKNOWN	Dynamically defines a column of an unknown type.
DBE_SQL.GET_RESULT	Reads a dynamically defined column value.
DBE_SQL.GET_RESULT_CHAR	Reads a dynamically defined column value of the CHAR type.
DBE_SQL.GET_RESULT_INT	Reads a dynamically defined column value of the INT type.
DBE_SQL.GET_RESULT_LONG	Reads a dynamically defined column value of the LONG type.
DBE_SQL.GET_RESULT_RAW	Reads a dynamically defined column value of the RAW type.
DBE_SQL.GET_RESULT_TEXT	Reads a dynamically defined column value of the TEXT type.
DBE_SQL.GET_RESULT_UNKNOWN	Reads a dynamically defined column value of an unknown type.
DBE_SQL.DBE_SQL_GET_RESULT_CHAR	Reads a dynamically defined column value of the CHAR type.
DBE_SQL.DBE_SQL_GET_RESULT_LONG	Reads a dynamically defined column value of the LONG type.
DBE_SQL.DBE_SQL_GET_RESULT_LONG2	Reads a dynamically defined column value of the LONG type.
DBE_SQL.DBE_SQL_GET_RESULT_RAW	Reads a dynamically defined column value of the RAW type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_SQL.IS_ACTIVE	Checks whether a cursor is opened.
DBE_SQL.LAST_ROW_COUNT	Returns the cumulative count of obtained rows.
DBE_SQL.RUN_AND_NEXT	Reads data of a cursor after a set of dynamically defined operations are performed on the cursor.
DBE_SQL.SQL_BIND_VARIABLE	Binds a value to a variable in a statement.
DBE_SQL.SQL_BIND_ARRAY	Binds a group of values to a variable in a statement.
DBE_SQL.SET_RESULT_TYPE_INTS	Dynamically defines a column of the INT array type.
DBE_SQL.SET_RESULT_TYPE_TEXTS	Dynamically defines a column of the TEXT array type.
DBE_SQL.SET_RESULT_TYPE_RAWS	Dynamically defines a column of the RAW array type.
DBE_SQL.SET_RESULT_TYPE_BYTEAS	Dynamically defines a column of the BYTEA array type.
DBE_SQL.SET_RESULT_TYPE_CHARS	Dynamically defines a column of the CHAR array type.
DBE_SQL.SET_RESULTS_TYPE	Dynamically defines a column of the array type.
DBE_SQL.GET_RESULTS_INT	Reads a dynamically defined column value of the INT array type.
DBE_SQL.GET_RESULTS_TEXT	Reads a dynamically defined column value of the TEXT array type.
DBE_SQL.GET_RESULTS_RAW	Reads a dynamically defined column value of the RAW array type.
DBE_SQL.GET_RESULTS_BYTEA	Reads a dynamically defined column value of the BYTEA array type.
DBE_SQL.GET_RESULTS_CHAR	Reads a dynamically defined column value of the CHAR array type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_SQL.GET_RESULTS	Reads a dynamically defined column value.
DBE_SQL.DESCRIBE_COLUMNS	Describes the column information read by the cursor.
DBE_SQL.SQL_DESCRIBE_COLUMNS	Describes the column information read by the cursor.
DBE_SQL.BIND_VARIABLE	Binds parameters.
DBE_SQL.SQL_SET_RESULTS_TYPE_C	Dynamically defines a column of the array type.
DBE_SQL.SQL_GET_VALUES_C	Reads a dynamically defined column value.
DBE_SQL.GET_VARIABLE_RESULT	Reads the return value of an SQL statement.
DBE_SQL.GET_VARIABLE_RESULT_CHAR	Reads the return value (of the CHAR type) of an SQL statement.
DBE_SQL.GET_VARIABLE_RESULT_RAW	Reads the return value (of the RAW type) of an SQL statement.
DBE_SQL.GET_VARIABLE_RESULT_TEXT	Reads the return value (of the TEXT type) of an SQL statement.
DBE_SQL.GET_VARIABLE_RESULT_INT	Reads the return value (of the INT type) of an SQL statement.
DBE_SQL.GET_ARRAY_RESULT_TEXT	Reads the return value (of the TEXT array type) of an SQL statement.
DBE_SQL.GET_ARRAY_RESULT_RAW	Reads the return value (of the RAW array type) of an SQL statement.
DBE_SQL.GET_ARRAY_RESULT_CHAR	Reads the return value (of the CHAR array type) of an SQL statement.
DBE_SQL.GET_ARRAY_RESULT_INT	Reads the return value (of the INT array type) of an SQL statement.

NOTE

- You are advised to use DBE_SQL.SET_RESULT_TYPE and DBE_SQL.GET_RESULT to define columns.
- When the size of the result set exceeds the value of the GUC parameter **work_mem**, the result set is stored on a temporary disk, but the maximum size of the temporary disk does not exceed 512 MB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_SQL.REGISTER_CONTEXT

  Opens a cursor, which is the prerequisite for the subsequent DBE_SQL operations. This function does not pass any parameter. It automatically generates cursor IDs in an ascending order and returns values to integer variables.

  NOTE

  Cursors opened by DBE_SQL are session-level variables. Cross-session calling of opened cursors (such as autonomous transactions) is not supported. If a cross-session cursor is called, the behavior is unpredictable.

  The prototype of DBE_SQL.REGISTER_CONTEXT is as follows:

```
DBE_SQL.REGISTER_CONTEXT(
)
RETURN INTEGER;
```

- DBE_SQL.SQL_UNREGISTER_CONTEXT

  Closes a cursor, which is the end of each DBE_SQL operation. If this function is not called when the stored procedure ends, the memory occupied by the cursor is not released. Therefore, remember to close the cursor. If an exception occurs, the stored procedure may exit in advance and the cursor cannot be closed. It is recommended that this API be included in the exception handling of the stored procedure.

  The prototype of DBE_SQL.SQL_UNREGISTER_CONTEXT is as follows:

```
DBE_SQL.SQL_UNREGISTER_CONTEXT(
    context_id    IN INT
)
RETURN INTEGER;
```

Table 2-318 DBE_SQL.SQL_UNREGISTER_CONTEXT API parameters

Parameter	Type	Input/OutputParameter	CanBeNull	Description
context_id	INT	IN	No	ID of the cursor to be closed.

- DBE_SQL.SQL_SET_SQL

  Parses SQL statements or anonymous blocks in a given cursor. The statement parameters can be passed only through the TEXT type. The length cannot exceed 1 GB.

  The prototype of DBE_SQL.SQL_SET_SQL is as follows:

```
DBE_SQL.SQL_SET_SQL(
   context_id    IN INT,
   query_string  IN TEXT,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    language_flag IN INT
)
RETURN BOOLEAN;
```

Table 2-319 DBE_SQL.SQL_SET_SQL API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor whose query statement is to be parsed.
query_string	TEXT	IN	No	Query statement to be parsed.
language_flag	INT	IN	No	Version language. The value **1** indicates an incompatible version, and the value **2** indicates an ORA-compatible version.

- DBE_SQL.SQL_RUN

  Executes a specified cursor. It receives a cursor ID and executes SQL statements or anonymous blocks in a given cursor.

  The prototype of DBE_SQL.SQL_RUN is as follows:

```
DBE_SQL.SQL_RUN(
    context_id    IN INT,
)
RETURN INTEGER;
```

Table 2-320 DBE_SQL.SQL_RUN API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor whose query statement is to be parsed.

- DBE_SQL.NEXT_ROW

  Returns the number of data rows that meet query conditions. Each time the function is called, the system obtains a set of new rows until all data is read.

  The prototype of DBE_SQL.NEXT_ROW is as follows:

```
DBE_SQL.NEXT_ROW(
    context_id    IN INT,
)
RETURN INTEGER;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-321 DBE_SQL.NEXT_ROW API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.

- DBE_SQL.SET_RESULT_TYPE

  Defines columns returned from a specified cursor. This API applies only to cursors defined by SELECT. The defined columns are identified by the relative positions in the query list. The data type of an input variable determines the corresponding column type.

  The prototype of DBE_SQL.SET_RESULT_TYPE is as follows:

```
DBE_SQL.SET_RESULT_TYPE(
    context_id IN INT,
    pos        IN INT,
    column_ref IN ANYELEMENT,
    maxsize    IN INT default 1024
)
RETURN INTEGER;
```

Table 2-322 DBE_SQL.SET_RESULT_TYPE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Relative position of the queried columns in the returned result. The value starts from 1.
column_ref	ANYELEMENT	IN	No	Variable of any type. You can select an appropriate API to dynamically define columns based on variable types.
maxsize	INT	IN	Yes	Length of the defined column return type.

- DBE_SQL.SET_RESULT_TYPE_CHAR

  Defines columns of the CHAR type returned from a specified cursor. This API applies only to cursors defined by SELECT. The defined columns are identified

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

by the relative positions in the query list. The data type of an input variable determines the corresponding column type.

The prototype of DBE_SQL.SET_RESULT_TYPE_CHAR is as follows:

```
DBE_SQL.SET_RESULT_TYPE_CHAR(
   context_id  IN INT,
   pos         IN INT,
   column_ref  IN TEXT,
   column_size IN INT
)
RETURN INTEGER;
```

Table 2-323 DBE_SQL.SET_RESULT_TYPE_CHAR API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_ref	TEXT	IN	No	Parameter to be defined.
column_size	INT	IN	No	Length of a dynamically defined column.

- DBE_SQL.SET_RESULT_TYPE_INT

  Defines columns of the INT type returned from a specified cursor. This API applies only to cursors defined by SELECT. The defined columns are identified by the relative positions in the query list. The data type of an input variable determines the corresponding column type.

  The prototype of DBE_SQL.SET_RESULT_TYPE_INT is as follows:

  ```
  DBE_SQL.SET_RESULT_TYPE_INT(
     context_id IN INT,
     pos        IN INT
  )
  RETURN INTEGER;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-324 DBE_SQL.SET_RESULT_TYPE_INT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.

- DBE_SQL.SET_RESULT_TYPE_LONG

  Defines columns of the long type (not the long type of data) returned from a specified cursor. This API applies only to cursors defined by SELECT. The defined columns are identified by the relative positions in the query list. The data type of an input variable determines the corresponding column type. The maximum size of a long column is 1 GB.

  The prototype of DBE_SQL.SET_RESULT_TYPE_LONG is as follows:

```
DBE_SQL.SET_RESULT_TYPE_LONG(
   context_id IN INT,
   pos        IN INT
)
RETURN INTEGER;
```

Table 2-325 DBE_SQL.SET_RESULT_TYPE_LONG API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.

- DBE_SQL.SET_RESULT_TYPE_RAW

  Defines columns of the RAW type returned from a specified cursor. This API applies only to cursors defined by SELECT. The defined columns are identified by the relative positions in the query list. The data type of an input variable determines the corresponding column type.

  The prototype of DBE_SQL.SET_RESULT_TYPE_RAW is as follows:

```
DBE_SQL.SET_RESULT_TYPE_RAW(
   context_id  IN INT,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  pos         IN INT,
  column_ref  IN BYTEA,
  column_size IN INT
)
RETURN INTEGER;
```

Table 2-326 DBE_SQL.SET_RESULT_TYPE_RAW API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_ref	BYTEA	IN	No	RAW variable.
column_size	INT	IN	No	Column length.

- DBE_SQL.SET_RESULT_TYPE_TEXT

  Defines columns of the TEXT type returned from a specified cursor. This API applies only to cursors defined by SELECT. The defined columns are identified by the relative positions in the query list. The data type of an input variable determines the corresponding column type.

  The prototype of DBE_SQL.SET_RESULT_TYPE_TEXT is as follows:

```
DBE_SQL.SET_RESULT_TYPE_TEXT(
  context_id IN INT,
  pos        IN INT,
  maxsize    IN INT
)
RETURN INTEGER;
```

Table 2-327 DBE_SQL.SET_RESULT_TYPE_TEXT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	IN	IN	No	ID of the cursor to be executed.
pos	IN	IN	No	Position of a dynamically defined column in the query.
maxsize	IN	IN	No	Maximum length of the defined TEXT type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_SQL.SET_RESULT_TYPE_UNKNOWN

  Processes columns of unknown data types returned from a specified cursor. It is used to report an error and exit only when the data type cannot be identified.

  The prototype of DBE_SQL.SET_RESULT_TYPE_UNKNOWN is as follows:

```
DBE_SQL.SET_RESULT_TYPE_UNKNOWN(
    context_id IN INT,
    pos        IN INT,
    col_type   IN TEXT
)
RETURN INTEGER;
```

  Table 2-328 DBE_SQL.SET_RESULT_TYPE_UNKNOWN API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.
col_type	TEXT	IN	No	Dynamically defined parameter.

- DBE_SQL.GET_RESULT

  Returns the element value of a cursor at a specified position. This API accesses the data obtained by DBE_SQL.NEXT_ROW.

  The prototype of DBE_SQL.GET_RESULT is as follows:

```
DBE_SQL.GET_RESULT(
    context_id   IN INT,
    pos          IN INT,
    column_value INOUT ANYELEMENT
);
```

  Table 2-329 DBE_SQL.GET_RESULT API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input / Output Parameter	Can Be Null	Description
pos	INT	IN	No	Relative position of the queried columns in the returned result. The value starts from 1.
column_value	ANYELEMENT	INOUT	No	Return value of a defined column.

- DBE_SQL.GET_RESULT_CHAR

  Returns a value of the CHAR type at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW.

  The prototype of DBE_SQL.GET_RESULT_CHAR is as follows:

```
DBE_SQL.GET_RESULT_CHAR(
    context_id    IN    INT,
    pos           IN    INT,
    tr            INOUT CHAR,
    err           INOUT NUMERIC DEFAULT 0,
    actual_length INOUT INT DEFAULT 1024
);
```

  Table 2-330 DBE_SQL.GET_RESULT_CHAR API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.
tr	CHAR	INOUT	No	Return value.
err	NUMERIC	INOUT	Yes	Error number. It is an output parameter. The input parameter must be a variable. The default value is **0**. Currently, the output value is **-1** regardless of the input parameter.
actual_length	INT	INOUT	Yes	Length of a return value. The default value is **1024**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_SQL.GET_RESULT_INT

  Returns a value of the INT type at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW.

  The prototype of DBE_SQL.GET_RESULT_INT is as follows:

```
DBE_SQL.GET_RESULT_INT(
   context_id IN    INT,
   pos        IN    INT
)
RETURN INTEGER;
```

Table 2-331 DBE_SQL.GET_RESULT_INT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.

- DBE_SQL.GET_RESULT_LONG

  Returns the value of a long column (not of the long or bigint type) at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW.

  The prototype of DBE_SQL.GET_RESULT_LONG is as follows:

```
DBE_SQL.GET_RESULT_LONG(
   context_id IN    INT,
   pos        IN    INT,
   lgth       IN    INT,
   off_set    IN    INT,
   vl         INOUT TEXT,
   vl_length  INOUT INT
);
```

Table 2-332 DBE_SQL.GET_RESULT_LONG API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input / Output Parameter	Can Be Null	Description
lgth	INT	IN	No	Length of a return value.
off_set	INT	IN	No	Start position of a return value.
vl	TEXT	INOUT	No	Return value.
vl_length	INT	INOUT	No	Length of a return value.

- DBE_SQL.GET_RESULT_RAW

  Returns a value of the RAW type at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW.

  The prototype of DBE_SQL.GET_RESULT_RAW is as follows:

```
DBE_SQL.GET_RESULT_RAW(
    context_id     IN     INT,
    pos            IN     INT,
    tr             INOUT BYTEA,
    err            INOUT NUMERIC DEFAULT 0,
    actual_length INOUT INT DEFAULT 1024
);
```

Table 2-333 DBE_SQL.GET_RESULT_RAW API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.
tr	BYTEA	INOUT	No	Returned column value.
err	NUMERIC	INOUT	No	Error number. It is an output parameter. The input parameter must be a variable. The default value is **0**. Currently, the output value is **-1** regardless of the input parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
actual_length	INT	INOUT	No	Length of a return value. The value longer than this length will be truncated. The default value is **1024**.

- DBE_SQL.GET_RESULT_TEXT

  Returns a value of the TEXT type at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW.

  The prototype of DBE_SQL.GET_RESULT_TEXT is as follows:

```
DBE_SQL.GET_RESULT_TEXT(
    context_id IN INT,
    pos        IN INT
)
RETURN TEXT;
```

  Table 2-334 DBE_SQL.GET_RESULT_TEXT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.

- DBE_SQL.GET_RESULT_UNKNOWN

  Returns a value of an unknown type at a specified position from a cursor. It serves as an error handling API when the type is not unknown.

  The prototype of DBE_SQL.GET_RESULT_UNKNOWN is as follows:

```
DBE_SQL.GET_RESULT_UNKNOWN(
    context_id IN INT,
    pos        IN INT,
    col_type   IN TEXT
)
RETURN INTEGER;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-335 DBE_SQL.GET_RESULT_UNKNOWN API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.
col_type	TEXT	IN	No	Returned parameter type.

- DBE_SQL.DBE_SQL_GET_RESULT_CHAR

  Returns a value of the CHAR type at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW. Different from the DBE_SQL.GET_RESULT_CHAR function, this function does not set the length of the return value but returns the entire string.

  The prototype of DBE_SQL.DBE_SQL_GET_RESULT_CHAR is as follows:

```
DBE_SQL.DBE_SQL_GET_RESULT_CHAR(
    context_id    IN INT,
    pos           IN INT
)
RETURN CHARACTER;
```

Table 2-336 DBE_SQL.DBE_SQL_GET_RESULT_CHAR API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.

- DBE_SQL.DBE_SQL_GET_RESULT_LONG

  Returns the value of a long column (not of the long or bigint type) at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW. Different from DBE_SQL.GET_RESULT_LONG, this function does not set the length of the return value but returns the entire BIGINT value.

  The prototype of DBE_SQL.DBE_SQL_GET_RESULT_LONG is as follows:

```
DBE_SQL.DBE_SQL_GET_RESULT_LONG(
    context_id    IN INT,
    pos           IN INT
)
RETURN BIGINT;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-337 DBE_SQL.DBE_SQL_GET_RESULT_LONG API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.

NOTE

The return type of this function is different from that of DBE_SQL.SET_RESULT_TYPE_LONG and DBE_SQL.GET_RESULT_LONG. Therefore, you are advised to use DBE_SQL.DBE_SQL_GET_RESULT_LONG2 instead.

- DBE_SQL.DBE_SQL_GET_RESULT_LONG2

  Returns the value of a long column (not of the long or bigint type) at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW. Different from DBE_SQL.GET_RESULT_LONG, this function does not set the length of the return value but returns the column value.

  The prototype of DBE_SQL.DBE_SQL_GET_RESULT_LONG2 is as follows:

```
DBE_SQL.DBE_SQL_GET_RESULT_LONG2(
    context_id IN INTEGER,
    pos        IN INTEGER
)
RETURN TEXT;
```

Table 2-338 DBE_SQL.DBE_SQL_GET_RESULT_LONG2 API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INTEGER	IN	No	ID of the cursor to be executed.
pos	INTEGER	IN	No	Position of a dynamically defined column in the query.

- DBE_SQL.DBE_SQL_GET_RESULT_RAW

  Returns a value of the RAW type at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW.

  Different from the DBE_SQL.GET_RESULT_RAW function, this function does not set the length of the return value but returns the entire string.

  The prototype of DBE_SQL.DBE_SQL_GET_RESULT_RAW is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_SQL.DBE_SQL_GET_RESULT_RAW(
   context_id    IN    INT,
   pos           IN    INT
)
RETURN RAW;
```

Table 2-339 DBE_SQL.GET_RESULT_RAW API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be executed.
pos	INT	IN	No	Position of a dynamically defined column in the query.

- DBE_SQL.IS_ACTIVE

  Returns the status of a cursor. When a cursor is opened, parsed, executed, or defined, the value is **true**. When a cursor is closed, the value is **false**. If the status is unknown, an error is reported. In other cases, the cursor is closed by default.

  The prototype of DBE_SQL.IS_ACTIVE is as follows:

  ```
  DBE_SQL.IS_ACTIVE(
     context_id    IN    INT
  )
  RETURN BOOLEAN;
  ```

  Table 2-340 DBE_SQL.IS_ACTIVE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.

- DBE_SQL.LAST_ROW_COUNT

  Returns the cumulative count of data rows obtained after the latest NEXT_ROW execution.

  The prototype of DBE_SQL.LAST_ROW_COUNT is as follows:

  ```
  DBE_SQL.LAST_ROW_COUNT(
  )
  RETURN INTEGER;
  ```

- DBE_SQL.RUN_AND_NEXT

  Equivalent to calling SQL_RUN and NEXT_ROW in sequence.

  The prototype of DBE_SQL.RUN_AND_NEXT is as follows:

  ```
  DBE_SQL.RUN_AND_NEXT(
     context_id IN INT
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
)
RETURNS INTEGER;
```

Table 2-341 DBE_SQL.RUN_AND_NEXT API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor whose query statement is to be parsed.

- DBE_SQL.SQL_BIND_VARIABLE

  Binds a parameter to an SQL statement. When the SQL statement is executed, the SQL statement is executed based on the bound value.

  The prototype of DBE_SQL.SQL_BIND_VARIABLE is as follows:

```
DBE_SQL.SQL_BIND_VARIABLE(
    context_id     IN int,
    query_string   IN text,
    language_flag  IN anyelement,
    out_value_size IN int default null
)
RETURNS void;
```

Table 2-342 DBE_SQL.SQL_BIND_VARIABLE API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
query_string	TEXT	IN	No	Name of the bind variable.
language_flag	ANYELEMENT	IN	No	Bound value.
out_value_size	INT	IN	Yes	Size of the return value. Default value: **NULL**.

- DBE_SQL.SQL_BIND_ARRAY

  Binds a group of parameters to an SQL statement. When the SQL statement is executed, the SQL statement is executed based on the bound array.

  The prototype of the **DBE_SQL.SQL_BIND_ARRAY** function is as follows:

```
DBE_SQL.SQL_BIND_ARRAY(
    context_id   IN int,
    query_string IN text,
    value        IN anyarray
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
)
RETURNS void;
DBE_SQL.SQL_BIND_ARRAY(
    context_id   IN int,
    query_string IN text,
    value        IN anyarray,
    lower_index  IN int,
    higher_index IN int
)
RETURNS void;
```

Table 2-343 DBE_SQL.SQL_BIND_ARRAY API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
query_string	TEXT	IN	No	Name of the bind variable.
value	ANYARRAY	IN	No	Bound array.
lower_index	INT	IN	No	Minimum index of the bound array.
higher_index	INT	IN	No	Maximum index of the bound array.

- DBE_SQL.SET_RESULT_TYPE_INTS

  Defines columns of the INT array type returned from a specified cursor. This API applies only to cursors defined by SELECT. The defined columns are identified by the relative positions in the query list. The data type of an input variable determines the corresponding column type.

  The prototype of DBE_SQL.SET_RESULT_TYPE_INTS is as follows:

```
DBE_SQL.SET_RESULT_TYPE_INTS(
    context_id IN int,
    pos        IN int,
    column_ref IN anyarray,
    cnt        IN int,
    lower_bnd  IN int
)
RETURNS integer;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-344 DBE_SQL.SET_RESULT_TYPE_INTS API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_ref	ANYARRAY	IN	No	Type of the returned array.
cnt	INT	IN	No	Number of values obtained at a time.
lower_bnd	INT	IN	No	Start index when an array is returned.

- DBE_SQL.SET_RESULT_TYPE_TEXTS

  Defines columns of the TEXT array type returned from a specified cursor. This API applies only to cursors defined by SELECT. The defined columns are identified by the relative positions in the query list. The data type of an input variable determines the corresponding column type.

  The prototype of DBE_SQL.SET_RESULT_TYPE_TEXTS is as follows:

```
DBE_SQL.SET_RESULT_TYPE_TEXTS(
    context_id IN int,
    pos        IN int,
    column_ref IN anyarray,
    cnt        IN int,
    lower_bnd  IN int,
    maxsize    IN int
)
RETURNS VOID;
```

Table 2-345 DBE_SQL.SET_RESULT_TYPE_TEXTS API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Position of a dynamically defined column in the query.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input / Output Parameter	Can Be Null	Description
column_ref	ANYARRAY	IN	No	Type of the returned array.
cnt	INT	IN	No	Number of values obtained at a time.
lower_bnd	INT	IN	No	Start index when an array is returned.
maxsize	INT	IN	No	Maximum length of the defined TEXT type.

- DBE_SQL.SET_RESULT_TYPE_RAWS

  Defines columns of the RAW array type returned from a specified cursor. This API applies only to cursors defined by SELECT. The defined columns are identified by the relative positions in the query list. The data type of an input variable determines the corresponding column type.

  The prototype of DBE_SQL.SET_RESULT_TYPE_RAWS is as follows:

```
DBE_SQL.set_result_type_raws(
    context_id  IN int,
    pos         IN int,
    column_ref  IN anyarray,
    cnt         IN int,
    lower_bnd   IN int,
    column_size IN int
)
RETURNS integer;
```

Table 2-346 DBE_SQL.SET_RESULT_TYPE_RAWS API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_ref	ANYARRAY	IN	No	Type of the returned array.
cnt	INT	IN	No	Number of values obtained at a time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input / Output Parameter	Can Be Null	Description
lower_bnd	INT	IN	No	Start index when an array is returned.
column_size	INT	IN	No	Column length.

- DBE_SQL.SET_RESULT_TYPE_BYTEAS

  Defines columns of the BYTEA array type returned from a specified cursor. This API applies only to cursors defined by SELECT. The defined columns are identified by the relative positions in the query list. The data type of an input variable determines the corresponding column type.

  The prototype of DBE_SQL.SET_RESULT_TYPE_BYTEAS is as follows:

```
DBE_SQL.set_result_type_byteas(
    context_id  IN int,
    pos         IN int,
    column_ref  IN anyarray,
    cnt         IN int,
    lower_bnd   IN int,
    column_size IN int
)
RETURNS integer;
```

Table 2-347 DBE_SQL.SET_RESULT_TYPE_BYTEAS API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_ref	ANYARRAY	IN	No	Type of the returned array.
cnt	INT	IN	No	Number of values obtained at a time.
lower_bnd	INT	IN	No	Start index when an array is returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input / Output Parameter	Can Be Null	Description
column_size	INT	IN	No	Column length.

- DBE_SQL.SET_RESULT_TYPE_CHARS

  Defines columns of the CHAR array type returned from a specified cursor. This API applies only to cursors defined by SELECT. The defined columns are identified by the relative positions in the query list. The data type of an input variable determines the corresponding column type.

  The prototype of DBE_SQL.SET_RESULT_TYPE_CHARS is as follows:

```
DBE_SQL.SET_RESULT_TYPE_CHARS(
   context_id  IN int,
   pos         IN int,
   column_ref  IN anyarray,
   cnt         IN int,
   lower_bnd   IN int,
   column_size IN int
)
RETURNS integer;
```

  Table 2-348 DBE_SQL.SET_RESULT_TYPE_CHARS API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_ref	ANYARRAY	IN	No	Type of the returned array.
cnt	INT	IN	No	Number of values obtained at a time.
lower_bnd	INT	IN	No	Start index when an array is returned.
column_size	INT	IN	No	Column length.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_SQL.SET_RESULTS_TYPE

  Defines columns returned from a specified cursor. This API applies only to cursors defined by SELECT. The defined columns are identified by the relative positions in the query list. The data type of an input variable determines the corresponding column type.

  The prototype of DBE_SQL.SET_RESULTS_TYPE is as follows:

```
DBE_SQL.SET_RESULTS_TYPE(
    context_id IN int,
    pos        IN int,
    column_ref IN anyarray,
    cnt        IN int,
    lower_bnd  IN int,
    maxsize    IN int DEFAULT 1024
) returns void;
```

  Table 2-349 DBE_SQL.SET_RESULTS_TYPE API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_ref	ANYARRAY	IN	No	Type of the returned array.
cnt	INT	IN	No	Number of values obtained at a time.
lower_bnd	INT	IN	No	Start index when an array is returned.
maxsize	INT	IN	Yes	Maximum length of the defined type.

- DBE_SQL.GET_RESULTS_INT

  Returns a value of the INT array type at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW.

  The prototype of DBE_SQL.GET_RESULTS_INT is as follows:

```
DBE_SQL.GET_RESULTS_INT(
    context_id   IN    int,
    pos          IN    int,
    column_value INOUT anyarray
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-350 DBE_SQL.GET_RESULTS_INT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_value	ANYARRAY	INOUT	No	Return value.

- DBE_SQL.GET_RESULTS_TEXT

  Returns a value of the TEXT array type at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW.

  The prototype of DBE_SQL.GET_RESULTS_TEXT is as follows:

```
DBE_SQL.GET_RESULTS_TEXT(
    context_id   IN    int,
    pos          IN    int,
    column_value INOUT anyarray
);
```

Table 2-351 DBE_SQL.GET_RESULTS_TEXT API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_value	ANYARRAY	INOUT	No	Return value.

- DBE_SQL.GET_RESULTS_RAW

  Returns a value of the RAW array type at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW.

  The prototype of DBE_SQL.GET_RESULTS_RAW is as follows:

```
DBE_SQL.GET_RESULTS_RAW(
    context_id   IN    int,
    pos          IN    int,
    column_value INOUT anyarray
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-352 DBE_SQL.GET_RESULTS_RAW API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_value	ANYARRAY	INOUT	No	Return value.

- DBE_SQL.GET_RESULTS_BYTEA

  Returns a value of the BYTEA array type at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW.

  The prototype of DBE_SQL.GET_RESULTS_BYTEA is as follows:

```
DBE_SQL.GET_RESULTS_BYTEA(
    context_id   IN    int,
    pos          IN    int,
    column_value INOUT anyarray
);
```

Table 2-353 DBE_SQL.GET_RESULTS_BYTEA API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_value	ANYARRAY	INOUT	No	Return value.

- DBE_SQL.GET_RESULTS_CHAR

  Returns a value of the CHAR array type at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW.

  The prototype of DBE_SQL.GET_RESULTS_CHAR is as follows:

```
DBE_SQL.GET_RESULTS_CHAR(
    context_id   IN    int,
    pos          IN    int,
    column_value INOUT anyarray
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-354 DBE_SQL.GET_RESULTS_CHAR API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_value	ANYARRAY	INOUT	No	Return value.

- DBE_SQL.GET_RESULTS

  Returns a value of the array type at a specified position from a cursor. This API accesses the data obtained by DBE_SQL.NEXT_ROW.

  NOTE

  The bottom-layer mechanism of DBE_SQL.GET_RESULTS is implemented through arrays. When different arrays are used to obtain the return value of the same column, **NULL** values are filled in the array due to discontinuous internal indexes to ensure the continuity of array indexes. As a result, the length of the returned array is different from that of the ORA Database.

  The prototype of DBE_SQL.GET_RESULTS is as follows:

```
DBE_SQL.GET_RESULTS(
   context_id   IN    int,
   pos          IN    int,
   column_value INOUT anyarray
);
```

Table 2-355 DBE_SQL.GET_RESULTS API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_value	ANYARRAY	INOUT	No	Return value.

- DBE_SQL.SQL_DESCRIBE_COLUMNS

  Describes column information. This API applies only to cursors defined by SELECT.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The prototype of DBE_SQL.SQL_DESCRIBE_COLUMNS is as follows:

```
DBE_SQL.SQL_DESCRIBE_COLUMNS(
   context_id IN    int,
   col_cnt    INOUT int,
   desc_t     INOUT dbe_sql.desc_tab
)RETURNS record;
```

Table 2-356 DBE_SQL.SQL_DESCRIBE_COLUMNS API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
col_cnt	INT	INOUT	No	Number of columns returned.
desc_t	DBE_SQL.DESC_TAB	INOUT	No	Description of the returned column.

- DBE_SQL.DESCRIBE_COLUMNS

  Describes column information. This API is compatible and can be used only for cursors defined by SELECT.

  The prototype of DBE_SQL.DESCRIBE_COLUMNS is as follows:

  ```
  DBE_SQL.DESCRIBE_COLUMNS(
     context_id IN  int,
     col_cnt    OUT int,
     desc_t     OUT dbe_sql.desc_tab
  )
  ```

  Table 2-357 DBE_SQL.DESCRIBE_COLUMNS API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
col_cnt	INT	OUT	No	Number of columns returned.
desc_t	DBE_SQL.DESC_TAB	OUT	No	Description of the returned column.

- DBE_SQL.BIND_VARIABLE

  Binds parameters. You are advised to use DBE_SQL.SQL_BIND_VARIABLE.

- DBE_SQL.SQL_SET_RESULTS_TYPE_C

  Dynamically defines a column of the array type. You are advised not to use it.

  The prototype of DBE_SQL.SQL_SET_RESULTS_TYPE_C is as follows:

```
DBE_SQL.sql_set_results_type_c(
   context_id IN int,
   pos        IN int,
   column_ref IN anyarray,
   cnt        IN int,
   lower_bnd  IN int,
   col_type   IN anyelement,
   maxsize    IN int
)return integer;
```

Table 2-358 DBE_SQL.SQL_SET_RESULTS_TYPE_C API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Position of a dynamically defined column in the query.
column_ref	ANYARRAY	IN	No	Type of the returned array.
cnt	INT	IN	No	Number of values obtained at a time.
lower_bnd	INT	IN	No	Start index when an array is returned.
col_type	ANYELEMENT	IN	No	Variable type corresponding to the returned array type.
maxsize	INT	IN	No	Maximum length of the defined type.

- DBE_SQL.SQL_GET_VALUES_C

  Reads a dynamically defined column value. You are advised not to use it.

  The prototype of DBE_SQL.SQL_GET_VALUES_C is as follows:

```
DBE_SQL.sql_get_values_c(
   context_id   IN    int,
   pos          IN    int,
   results_type INOUT anyarray,
   result_type  IN    anyelement
)return anyarray;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-359 DBE_SQL.SQL_GET_VALUES_C API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	INT	IN	No	Parameter position.
results_type	ANYARRAY	INOUT	No	Obtained result.
result_type	ANYELEMENT	IN	No	Type of the obtained result.

- DBE_SQL.GET_VARIABLE_RESULT

  Returns the value of the bound OUT parameter and obtains the OUT parameter in a stored procedure.

  The prototype of DBE_SQL.GET_VARIABLE_RESULT is as follows:

```
DBE_SQL.get_variable_result(
    context_id   IN    int,
    pos          IN    VARCHAR2,
    column_value INOUT anyelement
);
```

Table 2-360 DBE_SQL.GET_VARIABLE_RESULT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	VARCHAR2	IN	No	Name of the bound parameter.
column_value	ANYELEMENT	INOUT	No	Return value.

- DBE_SQL.GET_VARIABLE_RESULT_CHAR

  Returns the value of the bound OUT parameter of the CHAR type and obtains the OUT parameter in a stored procedure.

  The prototype of DBE_SQL.GET_VARIABLE_RESULT_CHAR is as follows:

```
DBE_SQL.get_variable_result_char(
    context_id IN int,
    pos        IN VARCHAR2
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
)
RETURNS char
```

Table 2-361 DBE_SQL.GET_VARIABLE_RESULT_CHAR API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	VARCHAR2	IN	No	Name of the bound parameter.

- DBE_SQL.GET_VARIABLE_RESULT_RAW

  Returns the value of the bound OUT parameter of the RAW type and obtains the OUT parameter in a stored procedure.

  The prototype of DBE_SQL.GET_VARIABLE_RESULT_RAW is as follows:

```
DBE_SQL.get_variable_result_raw(
    context_id IN     int,
    pos        IN     VARCHAR2,
    value      INOUT anyelement
);
```

Table 2-362 DBE_SQL.GET_VARIABLE_RESULT_RAW API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	VARCHAR2	IN	No	Name of the bound parameter.
value	ANYELEMENT	INOUT	No	Return value.

- DBE_SQL.GET_VARIABLE_RESULT_TEXT

  Returns the value of the bound OUT parameter of the TEXT type and obtains the OUT parameter in a stored procedure.

  The prototype of DBE_SQL.GET_VARIABLE_RESULT_TEXT is as follows:

```
CREATE OR REPLACE FUNCTION DBE_SQL.get_variable_result_text(
    context_id IN int,
    pos        IN VARCHAR2
)
RETURNS text
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-363 DBE_SQL.GET_VARIABLE_RESULT_TEXT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	VARCHAR2	IN	No	Name of the bound parameter.

- DBE_SQL.GET_VARIABLE_RESULT_INT

  Returns the value of the bound OUT parameter of the INT type and obtains the OUT parameter in a stored procedure.

  The prototype of DBE_SQL.GET_VARIABLE_RESULT_INT is as follows:

```
DBE_SQL.get_variable_result_int(
   context_id IN    int,
   pos        IN    VARCHAR2,
   value      INOUT anyelement
);
```

Table 2-364 DBE_SQL.GET_VARIABLE_RESULT_INT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	VARCHAR2	IN	No	Name of the bound parameter.
value	ANYELEMENT	INOUT	No	Return value.

- DBE_SQL.GET_ARRAY_RESULT_TEXT

  Returns the value of the bound OUT parameter of the TEXT array type and obtains the OUT parameter in a stored procedure.

  The prototype of DBE_SQL.GET_ARRAY_RESULT_TEXT is as follows:

```
DBE_SQL.get_array_result_text(
   context_id   IN    int,
   pos          IN    VARCHAR2,
   column_value INOUT anyarray
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-365 DBE_SQL.GET_ARRAY_RESULT_TEXT API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	VARCHAR2	IN	No	Name of the bound parameter.
column_value	ANYARRAY	INOUT	No	Return value.

- DBE_SQL.GET_ARRAY_RESULT_RAW

  Returns the value of the bound OUT parameter of the RAW array type and obtains the OUT parameter in a stored procedure.

  The prototype of DBE_SQL.GET_ARRAY_RESULT_RAW is as follows:

```
DBE_SQL.get_array_result_raw(
    context_id   IN    int,
    pos          IN    VARCHAR2,
    column_value INOUT anyarray
);
```

Table 2-366 DBE_SQL.GET_ARRAY_RESULT_RAW API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	VARCHAR2	IN	No	Name of the bound parameter.
column_value	ANYARRAY	INOUT	No	Return value.

- DBE_SQL.GET_ARRAY_RESULT_CHAR

  Returns the value of the bound OUT parameter of the CHAR array type and obtains the OUT parameter in a stored procedure.

  The prototype of DBE_SQL.GET_ARRAY_RESULT_CHAR is as follows:

```
DBE_SQL.get_array_result_char(
    context_id   IN    int,
    pos          IN    VARCHAR2,
    column_value INOUT anyarray
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-367 DBE_SQL.GET_ARRAY_RESULT_CHAR API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	VARCHAR2	IN	No	Name of the bound parameter.
column_value	ANYARRAY	INOUT	No	Return value.

- DBE_SQL.GET_ARRAY_RESULT_INT

  Returns the value of the bound OUT parameter of the INT array type and obtains the OUT parameter in a stored procedure.

  The prototype of DBE_SQL.GET_ARRAY_RESULT_INT is as follows:

```
DBE_SQL.get_array_result_int(
    context_id   IN    int,
    pos          IN    VARCHAR2,
    column_value INOUT anyarray
);
```

Table 2-368 DBE_SQL.GET_ARRAY_RESULT_INT API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
context_id	INT	IN	No	ID of the cursor to be queried.
pos	VARCHAR2	IN	No	Name of the bound parameter.
column_value	ANYARRAY	INOUT	No	Return value.

Examples

```
-- Example 1
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE test_desc_cols(
    id NUMBER,
    name VARCHAR2(50)
);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'id' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# INSERT INTO test_desc_cols(id, name) VALUES (1, 'xiaoming');
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
INSERT 0 1
gaussdb=# INSERT INTO test_desc_cols(id, name) VALUES (2, 'xiaohong');
INSERT 0 1
gaussdb=# INSERT INTO test_desc_cols(id, name) VALUES (3, 'xiaolan');
INSERT 0 1

gaussdb=# DECLARE
context_id  INTEGER;
col_cnt     INTEGER;
v_id int;
v_name varchar2;
execute_ret  INTEGER;
BEGIN
-- Open a cursor.
context_id := DBE_SQL.REGISTER_CONTEXT();
-- Compile the cursor.
DBE_SQL.SQL_SET_SQL(context_id, 'SELECT * FROM test_desc_cols', 2);
-- Set the return value type of a column.
DBE_SQL.SET_RESULT_TYPE(context_id, 1, v_id);
DBE_SQL.SET_RESULT_TYPE(context_id, 2, v_name);
execute_ret := DBE_SQL.SQL_RUN(context_id);
loop
exit when (DBE_SQL.NEXT_ROW(context_id) <= 0);
-- Obtain values.
DBE_SQL.GET_RESULT(context_id, 1, v_id);
DBE_SQL.GET_RESULT(context_id, 2, v_name);
-- Output the result.
dbe_output.print_line('id:'|| v_id || ' name:' || v_name);
end loop;
DBE_SQL.SQL_UNREGISTER_CONTEXT(context_id);
END;
/
id:1 name:xiaoming
id:2 name:xiaohong
id:3 name:xiaolan
ANONYMOUS BLOCK EXECUTE

-- Clean up the environment.
gaussdb=# drop table if exists test_desc_cols;
DROP TABLE

-- Example 2
gaussdb=# CREATE OR REPLACE PROCEDURE test_square(n NUMBER, square OUT NUMBER) IS
BEGIN
  square := n * n;
END;
/
CREATE PROCEDURE

gaussdb=# DECLARE
cur NUMBER;
query varchar(2000);
ret integer;
n NUMBER;
square Integer;
BEGIN
  n := 2;
  cur := DBE_SQL.REGISTER_CONTEXT();
  query := 'BEGIN test_square(:n_bnd, :square_bnd); END;';
  DBE_SQL.SQL_SET_SQL(cur, query, 2);
  DBE_SQL.SQL_BIND_VARIABLE(cur, 'n_bnd', n);
  DBE_SQL.SQL_BIND_VARIABLE(cur, 'square_bnd', square);
  ret := DBE_SQL.SQL_RUN(cur);
  DBE_SQL.GET_VARIABLE_RESULT(cur, 'square_bnd', square);
  DBE_OUTPUT.PRINT_LINE('square = ' || square);
  DBE_SQL.SQL_UNREGISTER_CONTEXT(cur);
END;
/
square = 4
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ANONYMOUS BLOCK EXECUTE

-- Clean up the environment.
gaussdb=# drop PROCEDURE test_square;
DROP PROCEDURE

-- Example 3
-- Examples of executing DESCRIBE_COLUMNS, RUN_AND_NEXT, and LAST_ROW_COUNT
-- Create a table and insert data into the table.
gaussdb=# CREATE TABLE test_desc_cols(
    id NUMBER,
    name VARCHAR2(50)
);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'id' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# INSERT INTO test_desc_cols(id, name) VALUES (1, 'xiaoming');
INSERT 0 1
gaussdb=# INSERT INTO test_desc_cols(id, name) VALUES (2, 'xiaohong');
INSERT 0 1
gaussdb=# INSERT INTO test_desc_cols(id, name) VALUES (3, 'xiaolan');
INSERT 0 1
-- Create a stored procedure for printing column description information.
gaussdb=# CREATE OR REPLACE PROCEDURE print_rec(
    rec in DBE_SQL.DESC_REC
)package AS
BEGIN
    raise INFO 'col_type            =    %', rec.col_type;
    raise INFO 'col_name              =    %', rec.col_name;
    raise INFO 'col_name_len        =    %', rec.col_name_len;
END;
/
CREATE PROCEDURE

-- Verify functions.
gaussdb=# DECLARE
context_id  INTEGER;
col_cnt     INTEGER;
rec_tab     DBE_SQL.DESC_TAB;
execute_ret  INTEGER;
nextrow_ret INTEGER;
last_row_count INTEGER;
BEGIN
-- Open a cursor.
context_id := DBE_SQL.REGISTER_CONTEXT();
-- Compile the cursor.
DBE_SQL.SQL_SET_SQL(context_id, 'SELECT * FROM test_desc_cols', 2);
-- Print the column description information.
DBE_SQL.DESCRIBE_COLUMNS(context_id, col_cnt, rec_tab);
FOR var IN 1..col_cnt LOOP
    print_rec(rec_tab(var));
END LOOP;
-- Execute and obtain a row of data.
execute_ret := DBE_SQL.RUN_AND_NEXT(context_id);
-- Obtain a row of data.
nextrow_ret := DBE_SQL.NEXT_ROW(context_id);
-- Obtain the number of obtained data rows.
last_row_count := DBE_SQL.LAST_ROW_COUNT();
DBE_OUTPUT.PRINT_LINE('last_row_count =  '|| last_row_count);
DBE_SQL.SQL_UNREGISTER_CONTEXT(context_id);
END;
/
INFO:  col_type            =    1700
CONTEXT:  SQL statement "CALL print_rec(rec_tab[var])"
PL/pgSQL function inline_code_block line 16 at PERFORM
INFO:  col_name              =    id
CONTEXT:  SQL statement "CALL print_rec(rec_tab[var])"
PL/pgSQL function inline_code_block line 16 at PERFORM
INFO:  col_name_len        =    2
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
CONTEXT:  SQL statement "CALL print_rec(rec_tab[var])"
PL/pgSQL function inline_code_block line 16 at PERFORM
INFO:  col_type           =    1043
CONTEXT:  SQL statement "CALL print_rec(rec_tab[var])"
PL/pgSQL function inline_code_block line 16 at PERFORM
INFO:  col_name           =    name
CONTEXT:  SQL statement "CALL print_rec(rec_tab[var])"
PL/pgSQL function inline_code_block line 16 at PERFORM
INFO:  col_name_len       =    4
CONTEXT:  SQL statement "CALL print_rec(rec_tab[var])"
PL/pgSQL function inline_code_block line 16 at PERFORM
last_row_count =  2
ANONYMOUS BLOCK EXECUTE

-- Clean up the environment.
gaussdb=# drop table test_desc_cols;
DROP TABLE

-- Example 4
gaussdb=# DROP TABLE if exists dbe_sql_tab;
NOTICE:  table "dbe_sql_tab" does not exist, skipping
DROP TABLE
gaussdb=# create table dbe_sql_tab(a char(30), b int, c text, d bytea, e text, f bool);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# insert into dbe_sql_tab values('aaa', 10, 'abcdefghijklmn', 'a', 'abcdefghijklmn', true);
INSERT 0 1

gaussdb=# DECLARE
cursorid int;
execute_ret int;
query varchar(2000);
err numeric;
v_char char(30);
v_int int;
v_long text;
v_long_len int;
v_bytea bytea;
v_text text;
BEGIN
query := 'select * from dbe_sql_tab';
cursorid := DBE_SQL.register_context();
DBE_SQL.sql_set_sql(cursorid, query, 1);
DBE_SQL.SET_RESULT_TYPE_CHAR(cursorid, 1, v_char, 30);
DBE_SQL.SET_RESULT_TYPE_INT(cursorid, 2);
DBE_SQL.SET_RESULT_TYPE_LONG(cursorid, 3);
DBE_SQL.SET_RESULT_TYPE_RAW(cursorid, 4, v_bytea, 8);
DBE_SQL.SET_RESULT_TYPE_TEXT(cursorid, 5, 1024);
execute_ret := DBE_SQL.sql_run(cursorid);
loop
exit when (DBE_SQL.next_row(cursorid) <= 0);
DBE_SQL.GET_RESULT_CHAR(cursorid, 1, v_char);
v_int := DBE_SQL.GET_RESULT_INT(cursorid, 2);
DBE_SQL.GET_RESULT_LONG(cursorid, 3, 3, 3, v_long, v_long_len);
DBE_SQL.GET_RESULT_RAW(cursorid, 4, v_bytea);
v_text := DBE_SQL.GET_RESULT_TEXT(cursorid, 5);
dbe_output.print_line('a:'|| v_char);
dbe_output.print_line('b:'|| v_int);
dbe_output.print_line('c:'|| v_long);
raise info 'd:%', v_bytea;
dbe_output.print_line('e:'|| v_text);
end loop;
DBE_SQL.sql_unregister_context(cursorid);
END;
/
a:aaa
b:10
c:cde
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
INFO:  d:\x61
e:abcdefghijklmn
ANONYMOUS BLOCK EXECUTE

gaussdb=# DECLARE
cursorid int;
execute_ret int;
query varchar(2000);
BEGIN
query := 'select * from dbe_sql_tab';
cursorid := DBE_SQL.register_context();
DBE_SQL.sql_set_sql(cursorid, query, 1);
DBE_SQL.SET_RESULT_TYPE_UNKNOWN(cursorid, 7, 'boolean');
execute_ret := DBE_SQL.sql_run(cursorid);
loop
exit when (DBE_SQL.next_row(cursorid) <= 0);
DBE_SQL.GET_RESULT_UNKNOWN(cursorid, 7, 'boolean');
end loop;
DBE_SQL.sql_unregister_context(cursorid);
END;
/
ERROR:  UnSupport data type for set_result_type(context: 8, pos: 7, 'boolean')
CONTEXT:  SQL statement "CALL pg_catalog.report_application_error('UnSupport data type for
set_result_type(context: '||context_id||', pos: '||pos||', '||PG_CATALOG.QUOTE_LITERAL(col_type)||')')"
PL/pgSQL function dbe_sql.set_result_type_unknown(integer,integer,text) line 8 at PERFORM
SQL statement "CALL dbe_sql.set_result_type_unknown(cursorid,7,'boolean')"
PL/pgSQL function inline_code_block line 10 at PERFORM

-- Clean up the environment.
gaussdb=# drop table dbe_sql_tab;
DROP TABLE

-- Example 5
gaussdb=# drop table if exists dbe_sql_tab;
NOTICE:  table "dbe_sql_tab" does not exist, skipping
DROP TABLE
gaussdb=# create table dbe_sql_tab(a char(30), b raw);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# insert into dbe_sql_tab values('aaa', HEXTORAW('DEADBEEF'));
INSERT 0 1

gaussdb=# DECLARE
cursorid int;
execute_ret int;
query varchar(2000);
v_char char(30);
v_raw bytea;
BEGIN
query := 'select * from dbe_sql_tab';
cursorid := DBE_SQL.register_context();
DBE_SQL.sql_set_sql(cursorid, query, 2);
DBE_SQL.SET_RESULT_TYPE(cursorid, 1, v_char);
DBE_SQL.SET_RESULT_TYPE_RAW(cursorid, 2, v_raw, 1024);
execute_ret := DBE_SQL.sql_run(cursorid);
loop
exit when (DBE_SQL.next_row(cursorid) <= 0);
v_char := DBE_SQL.DBE_SQL_GET_RESULT_CHAR(cursorid, 1);
v_raw := DBE_SQL.DBE_SQL_GET_RESULT_RAW(cursorid, 2);
dbe_output.print_line('a:'|| v_char);
raise info 'b:%', v_raw;
end loop;
DBE_SQL.sql_unregister_context(cursorid);
END;
/
a:aaa
INFO:  b:\x4445414442454546
ANONYMOUS BLOCK EXECUTE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Clean up the environment.
gaussdb=# drop table dbe_sql_tab;
DROP TABLE

-- Example 6
gaussdb=# DECLARE
cursorid int;
execute_ret int;
is_open boolean;
BEGIN
cursorid := DBE_SQL.register_context();
is_open := DBE_SQL.IS_ACTIVE(cursorid);
dbe_output.print_line('is_open:' ||is_open);
DBE_SQL.sql_unregister_context(cursorid);
is_open := DBE_SQL.IS_ACTIVE(cursorid);
dbe_output.print_line('is_open:' ||is_open);
END;
/
is_open:true
is_open:false
ANONYMOUS BLOCK EXECUTE

-- Example 7
gaussdb=# create table tbl(a integer ,b varchar(100));
OTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE

gaussdb=# DECLARE
c integer;
v1 integer[];
v2 varchar2[];
query varchar(2000);
ret integer;
begin
c := dbe_sql.register_context();
query := 'insert into tbl(a,b) values(:v_1, :v_2);';
dbe_sql.sql_set_sql(c, query, 2);
v1(1) := 1;
v1(2) := 2;
v2(1) := '1';
v2(2) := '2';
dbe_sql.sql_bind_array(c, 'v_1', v1);
dbe_sql.sql_bind_array(c, 'v_2', v2);
ret := dbe_sql.sql_run(c);
dbe_sql.sql_unregister_context(c);
end;
/
ANONYMOUS BLOCK EXECUTE

gaussdb=# select * from tbl order by a;
 a | b
---+---
 1 | 1
 2 | 2
(2 rows)

-- Clean up the environment.
gaussdb=# drop table tbl;
DROP TABLE

-- Example 8
-- Prerequisites
gaussdb=# drop table if exists dbe_sql_tab;
NOTICE:  table "dbe_sql_tab" does not exist, skipping
DROP TABLE
gaussdb=# create table dbe_sql_tab(a int, b text, c raw, d text, e char, f int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# insert into dbe_sql_tab values(1, '9', '5', '13', 'a', 1);
INSERT 0 1
gaussdb=# insert into dbe_sql_tab values(2, '9', '6', '14', 'b', 2);
INSERT 0 1
gaussdb=# insert into dbe_sql_tab values(3, '7', '7', '15', 'c', 3);
INSERT 0 1
gaussdb=# insert into dbe_sql_tab values(4, '6', '8', '16', 'd', 4);
INSERT 0 1

gaussdb=# DECLARE
query varchar(2000);
context_id int;
execute_ret int;
v_id int;
v_ints int[];
v_texts text[];
v_raws raw[];
v_byteas bytea[];
v_chars character[];
v_type int[];
BEGIN
query := ' select * from dbe_sql_tab order by 1';
context_id := dbe_sql.register_context();
dbe_sql.sql_set_sql(context_id, query, 1);
DBE_SQL.SET_RESULT_TYPE_INTS(context_id, 1, v_ints, 3, 1);
DBE_SQL.SET_RESULT_TYPE_TEXTS(context_id, 2, v_texts, 3, 1, 1024);
DBE_SQL.SET_RESULT_TYPE_RAWS(context_id, 3, v_raws, 3, 1, 1024);
DBE_SQL.SET_RESULT_TYPE_BYTEAS(context_id, 4, v_byteas, 3, 1, 1024);
DBE_SQL.SET_RESULT_TYPE_CHARS(context_id, 5, v_chars, 3, 1, 1024);
DBE_SQL.SET_RESULTS_TYPE(context_id, 6, v_type, 3, 1);
execute_ret := dbe_sql.sql_run(context_id);
loop
v_id := dbe_sql.next_row(context_id);
v_ints := DBE_SQL.GET_RESULTS_INT(context_id, 1, v_ints);
v_texts := DBE_SQL.GET_RESULTS_TEXT(context_id, 2, v_texts);
v_raws := DBE_SQL.GET_RESULTS_RAW(context_id, 3, v_raws);
v_byteas := DBE_SQL.GET_RESULTS_BYTEA(context_id, 4, v_byteas);
v_chars := DBE_SQL.GET_RESULTS_CHAR(context_id, 5, v_chars);
v_type := DBE_SQL.GET_RESULTS(context_id, 6, v_type);
exit when(v_id != 3);
end loop;
FOR i IN v_ints.FIRST .. v_ints.LAST  LOOP
    dbe_output.print_line('int' || i || ' = ' || v_ints[i]);
END LOOP;
FOR i IN v_texts.FIRST .. v_texts.LAST  LOOP
    dbe_output.print_line('text' || i || ' = ' || v_texts[i]);
END LOOP;
FOR i IN v_raws.FIRST .. v_raws.LAST  LOOP
    dbe_output.print_line('raw' || i || ' = ' || v_raws[i]);
END LOOP;
FOR i IN v_byteas.FIRST .. v_byteas.LAST  LOOP
    dbe_output.print_line('bytea' || i || ' = ' || v_byteas[i]);
END LOOP;
FOR i IN v_chars.FIRST .. v_chars.LAST  LOOP
    dbe_output.print_line('char' || i || ' = ' || v_chars[i]);
END LOOP;
FOR i IN v_type.FIRST .. v_type.LAST  LOOP
    dbe_output.print_line('type' || i || ' = ' || v_type[i]);
END LOOP;
dbe_sql.sql_unregister_context(context_id);
END;
/
int1 = 1
int2 = 2
int3 = 3
int4 = 4
text1 = 9
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
text2 = 9
text3 = 7
text4 = 6
raw1 = 05
raw2 = 06
raw3 = 07
raw4 = 08
bytea1 = \x3133
bytea2 = \x3134
bytea3 = \x3135
bytea4 = \x3136
char1 = a
char2 = b
char3 = c
char4 = d
type1 = 1
type2 = 2
type3 = 3
type4 = 4
ANONYMOUS BLOCK EXECUTE

-- Clean up the environment.
gaussdb=# drop table if exists dbe_sql_tab;
DROP TABLE

-- Example 9
-- Prerequisites
gaussdb=# drop table if exists dbe_sql_tab;
NOTICE:  table "dbe_sql_tab" does not exist, skipping
DROP TABLE
gaussdb=# create table dbe_sql_tab(a int ,b int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# insert into dbe_sql_tab values(1,3);
INSERT 0 1

gaussdb=# DECLARE
context_id int;
type re_rssc is record (col_num int, desc_col dbe_sql.desc_tab);
employer re_rssc;
res re_rssc;
d int;
dd dbe_sql.desc_tab;
query varchar(2000);
BEGIN
query := 'select * from dbe_sql_tab';
-- Open a cursor.
context_id := dbe_sql.register_context();
-- Compile the cursor.
dbe_sql.sql_set_sql(context_id, query, 1);
-- Execute the cursor.
res := dbe_sql.sql_describe_columns(context_id, d,dd);
-- Output the result.
dbe_output.print_line('col_num:' || res.col_num);
dbe_output.print_line('col_type:' || res.desc_col[1].col_type);
dbe_output.print_line('col_max_len:' || res.desc_col[1].col_max_len);
dbe_output.print_line('col_name:' || res.desc_col[1].col_name);
dbe_output.print_line('col_name_len:' || res.desc_col[1].col_name_len);
dbe_output.print_line('col_schema_name:' || res.desc_col[1].col_schema_name);
dbe_output.print_line('col_schema_name_len:' || res.desc_col[1].col_schema_name_len);
dbe_output.print_line('col_precision:' || res.desc_col[1].col_precision);
dbe_output.print_line('col_scale:' || res.desc_col[1].col_scale);
dbe_output.print_line('col_charsetid:' || res.desc_col[1].col_charsetid);
dbe_output.print_line('col_charsetform:' || res.desc_col[1].col_charsetform);
dbe_output.print_line('col_null_ok:' || res.desc_col[1].col_null_ok);
-- Close the cursor.
dbe_sql.sql_unregister_context(context_id);
END;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
/
col_num:2
col_type:23
col_max_len:4
col_name:a
col_name_len:1
col_schema_name:
col_schema_name_len:0
col_precision:0
col_scale:0
col_charsetid:0
col_charsetform:0
col_null_ok:true
ANONYMOUS BLOCK EXECUTE

-- Clean up the environment.
gaussdb=# drop table if exists dbe_sql_tab;
DROP TABLE

-- Example 10
gaussdb=# drop table if exists dbe_sql_tab;
NOTICE:  table "dbe_sql_tab" does not exist, skipping
DROP TABLE
gaussdb=# create table dbe_sql_tab(a int);
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE
gaussdb=# insert into dbe_sql_tab values(1);
INSERT 0 1
gaussdb=# insert into dbe_sql_tab values(2);
INSERT 0 1
gaussdb=# insert into dbe_sql_tab values(3);
INSERT 0 1

gaussdb=# DECLARE
query varchar(2000);
context_id int;
execute_ret int;
v_id int;
v_ints int[];
i1 integer;
BEGIN
query := 'select * from dbe_sql_tab order by 1';
context_id := dbe_sql.register_context();
dbe_sql.sql_set_sql(context_id, query, 1);
DBE_SQL.SQL_SET_RESULTS_TYPE_C(context_id, 1, v_ints, 3, 1, i1, 0);
execute_ret := dbe_sql.sql_run(context_id);
loop
v_id := dbe_sql.next_row(context_id);
v_ints := DBE_SQL.SQL_GET_VALUES_C(context_id, 1, v_ints, i1);
exit when(v_id != 3);
end loop;
FOR i IN v_ints.FIRST .. v_ints.LAST  LOOP
        dbe_output.print_line('int' || i || ' = ' || v_ints[i]);
END LOOP;
dbe_sql.sql_unregister_context(context_id);
END;
/
int1 = 1
int2 = 2
int3 = 3
ANONYMOUS BLOCK EXECUTE

-- Clean up the environment.
gaussdb=# drop table if exists dbe_sql_tab;
DROP TABLE

-- Example 11
gaussdb=# CREATE OR REPLACE PROCEDURE test_proc(out_int out Integer, out_char out char, out_raw out
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
raw, out_text out text) IS
BEGIN
  out_int := 1;
  out_char := 'a';
  out_raw := 'b';
  out_text := 'c';
END;
/
CREATE PROCEDURE

gaussdb=# DECLARE
cur NUMBER;
query varchar(2000);
ret integer;
v_int Integer;
v_char char;
v_raw raw;
v_text text;
BEGIN
  cur := DBE_SQL.REGISTER_CONTEXT();
  query := 'BEGIN test_proc(:v_int, :v_char, :v_raw, :v_text); END;';
  DBE_SQL.SQL_SET_SQL(cur, query, 2);
  DBE_SQL.SQL_BIND_VARIABLE(cur, 'v_int', v_int);
  DBE_SQL.SQL_BIND_VARIABLE(cur, 'v_char', v_char, 1024);
  DBE_SQL.SQL_BIND_VARIABLE(cur, 'v_raw', v_raw, 1024);
  DBE_SQL.SQL_BIND_VARIABLE(cur, 'v_text', v_text, 1024);
  ret := DBE_SQL.SQL_RUN(cur);
  DBE_SQL.GET_VARIABLE_RESULT_INT(cur, 'v_int', v_int);
  v_char := DBE_SQL.GET_VARIABLE_RESULT_CHAR(cur, 'v_char');
  DBE_SQL.GET_VARIABLE_RESULT_RAW(cur, 'v_raw', v_raw);
  v_text := DBE_SQL.GET_VARIABLE_RESULT_TEXT(cur, 'v_text');
  DBE_OUTPUT.PRINT_LINE('v_int = ' || v_int);
  DBE_OUTPUT.PRINT_LINE('v_char = ' || v_char);
  DBE_OUTPUT.PRINT_LINE('v_raw = ' || v_raw);
  DBE_OUTPUT.PRINT_LINE('v_text = ' || v_text);
  DBE_SQL.SQL_UNREGISTER_CONTEXT(cur);
END;
/
v_int = 1
v_char = a
v_raw = 0B
v_text = c
ANONYMOUS BLOCK EXECUTE

-- Clean up the environment.
gaussdb=# drop procedure test_proc;
DROP PROCEDURE

-- Example 12
gaussdb=# CREATE OR REPLACE PROCEDURE test_proc(out_int out Integer[], out_char out char[], out_raw
out raw[], out_text out text[]) IS
BEGIN
  out_int(0) := 1;
  out_char(0) := 'a';
  out_raw(0) := 'b';
  out_text(0) := 'c';
END;
/
CREATE PROCEDURE

gaussdb=# DECLARE
cur NUMBER;
query varchar(2000);
ret integer;
v_int Integer[];
v_char char[];
v_raw raw[];
v_text text[];
BEGIN
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  cur := DBE_SQL.REGISTER_CONTEXT();
  query := 'call test_proc(:v_int, :v_char, :v_raw, :v_text);';
  DBE_SQL.SQL_SET_SQL(cur, query, 1);
  DBE_SQL.SQL_BIND_ARRAY(cur, 'v_int', v_int);
  DBE_SQL.SQL_BIND_ARRAY(cur, 'v_char', v_char);
  DBE_SQL.SQL_BIND_ARRAY(cur, 'v_raw', v_raw);
  DBE_SQL.SQL_BIND_ARRAY(cur, 'v_text', v_text);
  ret := DBE_SQL.SQL_RUN(cur);
  DBE_SQL.GET_ARRAY_RESULT_INT(cur, 'v_int', v_int);
  DBE_SQL.GET_ARRAY_RESULT_CHAR(cur, 'v_char', v_char);
  DBE_SQL.GET_ARRAY_RESULT_RAW(cur, 'v_raw', v_raw);
  DBE_SQL.GET_ARRAY_RESULT_TEXT(cur, 'v_text', v_text);
  DBE_OUTPUT.PRINT_LINE('v_int = ' || v_int(0));
  DBE_OUTPUT.PRINT_LINE('v_char = ' || v_char(0));
  DBE_OUTPUT.PRINT_LINE('v_raw = ' || v_raw(0));
  DBE_OUTPUT.PRINT_LINE('v_text = ' || v_text(0));
  DBE_SQL.SQL_UNREGISTER_CONTEXT(cur);
END;
/
v_int = 1
v_char = a
v_raw = 0B
v_text = c
ANONYMOUS BLOCK EXECUTE

-- Clean up the environment.
gaussdb=# drop PROCEDURE test_proc;
DROP PROCEDURE
```

2.12.2.18 DBE_STATS

The advanced package DBE_STATS provides the function of managing statistics information, including locking, unlocking, exporting, importing, setting, and rolling back historical statistics. **Table 2-369** lists APIs supported by DBE_STATS.

NOTE

- DBE_STATS does not support temporary tables.
- The rules for managing DBE_STATS statistics are the same as those for managing existing statistics. Only the functions related to existing statistics in the kernel are supported.
- DBE_STATS and multi-version statistics management do not support the M-compatible mode.
- DBE_STATS does not support the API for deleting statistics.
- The multi-version statistics management function requires that the GUC parameters be consistent on all nodes.
- The values of **stats_history_retention_time** (time for retaining historical statistics) and **stats_history_record_limit** (maximum number of historical statistics records) must be the same for all nodes. If the GUC parameter values are different on different nodes, the historical statistics of the same object on different nodes are inconsistent. During the rollback, the historical statistics of a specified timestamp may not be found on a node, causing the rollback failure.
- After a node is added, if the rollback API is used to specify the timestamp before the scale-out for rollback, the new node reuses the CN metadata. Because the historical statistics are also metadata, the historical statistics of the node are used for rollback.
- Modifying statistics (importing, rolling back, and setting statistics) will invalidate the plan cache. The execution plan in the cache records the version of the statistics on which it depends. If the current statistics version is different from the version recorded in the cached plan, the cached plan is invalid and needs to be regenerated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

APIs

Table 2-369 DBE_STATS

API	Description	Permission Requirement
DBE_STATS.LOCK_TABLE_STATS	Locks table-level statistics.	When using this API, you must have the same permission on the specified table as that of the ANALYZE statement. For permission details, see **ANALYZE | ANALYSE**.
DBE_STATS.LOCK_PARTITION_STATS	Locks partition-level statistics.	
DBE_STATS.LOCK_COLUMN_STATS	Locks column-level statistics.	
DBE_STATS.LOCK_SCHEMA_STATS	Locks statistics about all tables in a specified schema.	• When this API is called, all tables on which the current user has the ANALYZE permission in the schema are locked. Tables on which the current user does not have the ANALYZE permission are skipped. • To call this API, you must have the USAGE permission on the specified schema.
DBE_STATS.UNLOCK_TABLE_STATS	Unlocks table-level statistics.	When using this API, you must have the same permission on the specified table as that of the ANALYZE statement. For permission details, see **ANALYZE | ANALYSE**.
DBE_STATS.UNLOCK_PARTITION_STATS	Unlocks partition-level statistics.	
DBE_STATS.UNLOCK_COLUMN_STATS	Unlocks column-level statistics.	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description	Permission Requirement
DBE_STATS.UNLOCK_SCHEMA_STATS	Unlocks statistics about all tables in a specified schema.	• When this API is called, all tables on which the current user has the ANALYZE permission in the schema are unlocked. Tables on which the current user does not have the ANALYZE permission are skipped. • To call this API, you must have the USAGE permission on the specified schema.
DBE_STATS.RESTORE_TABLE_STATS	Rolls back table-level statistics to a specified time point.	When using this API, you must have the same permission on the specified table as that of the ANALYZE statement. For permission details, see ANALYZE | ANALYSE.
DBE_STATS.RESTORE_PARTITION_STATS	Rolls back partition-level statistics to a specified time point.	
DBE_STATS.RESTORE_COLUMN_STATS	Rolls back column-level statistics to a specified time point.	
DBE_STATS.RESTORE_SCHEMA_STATS	Rolls back statistics about all tables in a specified schema to a specified time point.	• When this API is called, all tables on which the current user has the ANALYZE permission in the schema are rolled back. Tables on which the current user does not have the ANALYZE permission are skipped. • To call this API, you must have the USAGE permission on the specified schema.
DBE_STATS.PURGE_STATS	Clears all historical statistics before a specified time point.	Only the initial user can call this API.
DBE_STATS.GET_STATS_HISTORY_RETENTION	Obtains the retention period of historical statistics.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description	Permission Requirement
DBE_STATS.GET_STATS_HISTORY_AVAILABILITY	Obtains the earliest available time of the current historical statistics.	-
DBE_STATS.CREATE_STAT_TABLE	Creates a statistics table for storing statistics.	To call this procedure to create a statistics table, you must have the permission to create tables and indexes in the specified schema and tablespace.
DBE_STATS.DROP_STAT_TABLE	Deletes a statistics table for storing statistics.	To call this procedure to delete a statistics table, you must have the permission to delete the table and index in the specified schema.
DBE_STATS.EXPORT_INDEX_STATS	Retrieves statistics for the specified index and stores them in the user statistics table.	• When using this API, you must have the same permission on the specified table as that of the ANALYZE statement. For permission details, see ANALYZE | ANALYSE. • To call this procedure to export statistics to a user statistics table, you must have the SELECT, INSERT, DELETE, and UPDATE permissions on the user statistics table.
DBE_STATS.EXPORT_TABLE_STATS	Retrieves statistics for the specified table and stores them in the user statistics table.	
DBE_STATS.EXPORT_COLUMN_STATS	Retrieves statistics for the specified column and stores them in the user statistics table.	
DBE_STATS.EXPORT_SCHEMA_STATS	Retrieves statistics for the specified schema and stores them in the user statistics table.	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description	Permission Requirement
DBE_STATS.EXPORT_DATABASE_STATS	Retrieves statistics for the current database and stores them in the user statistics table.	• When this API is called, all tables on which the current user has the ANALYZE permission in the current database are exported. Tables on which the current user does not have the ANALYZE permission are skipped. • To call this procedure to export statistics to a user statistics table, you must have the SELECT, INSERT, DELETE, and UPDATE permissions on the user statistics table.
DBE_STATS.IMPORT_INDEX_STATS	Retrieves statistics for the specified index from the user statistics table and writes it back to the system catalog.	• When using this API, you must have the same permission on the specified table as that of the ANALYZE statement. For permission details, see ANALYZE | ANALYSE. • To call this procedure to import statistics from a user statistics table to a system catalog, you must have the SELECT, INSERT, DELETE, and UPDATE permissions on the user statistics table.
DBE_STATS.IMPORT_TABLE_STATS	Retrieves statistics for the specified table from the user statistics table and writes it back to the system catalog.	
DBE_STATS.IMPORT_COLUMN_STATS	Retrieves statistics for the specified column from the user statistics table and writes it back to the system catalog.	
DBE_STATS.IMPORT_SCHEMA_STATS	Retrieves statistics for the specified schema from the user statistics table and writes it back to the system catalog.	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description	Permission Requirement
DBE_STATS.IMPORT_DATABASE_STATS	Retrieves statistics for the specified database from the user statistics table and writes it back to the system catalog.	• When this API is called, all tables on which the current user has the ANALYZE permission in the current database are imported. Tables on which the current user does not have the ANALYZE permission are skipped. • To call this procedure to export statistics to a user statistics table, you must have the SELECT, INSERT, DELETE, and UPDATE permissions on the user statistics table.
DBE_STATS.SET_COLUMN_STATS	Sets column-related statistics, including single-column, multi-column, and expression statistics.	When using this API, you must have the same permission on the specified table as that of the ANALYZE statement. For permission details, see ANALYZE | ANALYSE.
DBE_STATS.SET_INDEX_STATS	Sets index-related statistics.	
DBE_STATS.SET_TABLE_STATS	Sets table-related statistics.	
DBE_STATS.GET_TABLES_BY_SQLID	Obtains all tables involved in the statement corresponding to the unique SQL ID. If multiple tables are involved, multiple rows of records are returned.	When using this function, the user must meet one of the following conditions: • The user is an initial user. • The user has the MONADMIN access permission.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description	Permission Requirement
DBE_STATS.ANALYZE_SQLID	Obtains all tables involved in the statement corresponding to the unique SQL ID and collects statistics about the tables. If multiple tables are successfully collected, multiple rows of records are returned.	When using this function, the user must meet one of the following conditions: • The user is an initial user. • The user has the MONADMIN access permission and has the same permissions on the corresponding table as the **ANALYZE** command. For permission details, see **ANALYZE | ANALYSE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description	Permission Requirement
dbe_stats_tools.py	Exports and imports database-level statistics, and exports the metadata of the statements for which the execution plan is optimized.	• To export statistics, you must have the permission on the corresponding API or command (DBE_STATS.CREATE_STAT_TABLE, DBE_STATS.EXPORT_DATABASE_STATS, \COPY TO or DBE_STATS.DROP_STAT_TABLE) and the read and write permissions on the specified export directory. • To import statistics, you must have the permission on the corresponding API or command (DBE_STATS.CREATE_STAT_TABLE, DBE_STATS.EXPORT_DATABASE_STATS, \COPY FROM or DBE_STATS.DROP_STAT_TABLE) and the read permission on the files in the specified export directory. • To export statement metadata, you must have the system administrator permission on the database and the read and write permissions on the specified directory.

- DBE_STATS.LOCK_TABLE_STATS

  Locks the statistics of a table. After the table is locked, the statistics of the table cannot be updated, and the indexes of the table are locked as well. This API does not return any value.

  The prototype of DBE_STATS.LOCK_TABLE_STATS is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_STATS.LOCK_TABLE_STATS(
  ownname   VARCHAR2,
  tabname   VARCHAR2
);
```

Table 2-370 DBE_STATS.LOCK_TABLE_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	varchar2	Yes	Name of the schema to which the table to be locked belongs. When it is null, the current schema is used by default.
tabname	varchar2	No	Name of the table to be locked.

NOTE

- You can view **stat_state** in **reloptions** of the table to obtain the lock status of the table.
- When a table is locked, its statistics cannot be updated.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
-- Create a table.
gaussdb=# CREATE SCHEMA dbe_stats_lock;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA=dbe_stats_lock;
SET
gaussdb=# CREATE TABLE t1(a int,b int) distribute by hash(a);
CREATE TABLE

-- Lock the table and check its locking status.
gaussdb=# CALL DBE_STATS.LOCK_TABLE_STATS(ownname=>'dbe_stats_lock',tabname=>'t1');
 lock_table_stats
------------------

(1 row)
gaussdb=# SELECT relname,instr(reloptions::text,'stat_state=locked',1,1) <> 0  AS exist_lock FROM
PG_CLASS WHERE relname='t1' AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_lock');
 relname | exist_lock
---------+------------
 t1      | t
(1 row)

-- An error is reported after the ANALYZE operation is performed.
gaussdb=# ANALYZE t1;
ERROR:  The statistics is locked, cannot be updated.

-- Drop the table and namespace.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_lock;
DROP SCHEMA
```

- DBE_STATS.LOCK_PARTITION_STATS

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Locks the statistics of a partition. After the partition is locked, the statistics of the partition cannot be updated, and the indexes of the partition are locked as well. This API does not return any value.

The prototype of DBE_STATS.LOCK_PARTITION_STATS is as follows:

```
DBE_STATS.LOCK_PARTITION_STATS(
   ownname     VARCHAR2,
   tabname     VARCHAR2,
   partname    VARCHAR2
);
```

Table 2-371 DBE_STATS.LOCK_PARTITION_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	varchar2	Yes	Name of the schema to which the partition to be locked belongs. When it is null, the current schema is used by default.
tabname	varchar2	No	Name of the table to which the partition to be locked belongs.
partname	varchar2	No	Name of the partition to be locked.

NOTE

- You can view **stat_state** in **reloptions** of the partition to obtain the lock status of the table.
- When the partition or the table to which the table belongs is locked, its statistics cannot be updated.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
-- Create a table.
gaussdb=# CREATE SCHEMA dbe_stats_lock;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA=dbe_stats_lock;
SET
gaussdb=# CREATE TABLE upart_table(a int, b int, c int) distribute by hash(a) PARTITION BY
RANGE(a)
 (
   PARTITION p1 VALUES LESS THAN(1200),
   PARTITION p2 VALUES LESS THAN(2400),
   PARTITION p3 VALUES LESS THAN(MAXVALUE)
 );
CREATE TABLE

-- Lock a partition. Other partitions and tables are not affected.
gaussdb=# CALL
DBE_STATS.LOCK_PARTITION_STATS(ownname=>'dbe_stats_lock',tabname=>'upart_table',partname=>'
p1');
 lock_partition_stats
----------------------

(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT relname,instr(reloptions::text,'stat_state=locked',1,1) <> 0 AS exist_lock FROM
PG_CLASS WHERE relname='upart_table';
   relname   | exist_lock
-------------+------------
 upart_table | f
(1 row)

gaussdb=# SELECT relname,instr(reloptions::text,'stat_state=locked',1,1) <> 0 AS exist_lock FROM
PG_PARTITION WHERE parentid='upart_table'::REGCLASS ORDER BY oid;
   relname   | exist_lock
-------------+------------
 upart_table | f
 p1          | t
 p2          | f
 p3          | f
(4 rows)
-- Drop the table and namespace.
gaussdb=# DROP TABLE upart_table;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_lock;
DROP SCHEMA
```

- DBE_STATS.LOCK_COLUMN_STATS

  Locks the statistics of a column. After the statistics are locked, they cannot be updated. This API does not return any value.

  The prototype of DBE_STATS.LOCK_COLUMN_STATS is as follows:

```
DBE_STATS.LOCK_COLUMN_STATS(
    ownname     VARCHAR2,
    tabname     VARCHAR2,
    colname     VARCHAR2,
    partname    VARCHAR2 DEFAULT NULL
);
```

Table 2-372 DBE_STATS.LOCK_COLUMN_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	varchar2	Yes	Name of the schema to which the column to be locked belongs. When it is null, the current schema is used by default.
tabname	varchar2	No	Name of the table to which the column to be locked belongs.
colname	varchar2	No	Name of the column to be locked.
partname	varchar2	Yes	Name of the partition to which the column to be locked belongs. The default value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If **partname** is **NULL**, the list-level statistics are locked by default.
- If the column-level statistics of the table do not exist, the locking can be executed successfully but does not take effect.
- You can view **stastate** in **PG_STATISTIC** or **PG_STATISTIC_EXT** to obtain the lock status.
- Before locking multiple columns, query the **staextname** column in **PG_STATISTIC_EXT** to obtain the aliases of multiple columns and pass them as the input parameter **colname**.
- If a column is locked or the table or partition to which the column belongs is locked, the statistics cannot be updated. The locked information about the column is not displayed. You need to view the locked information by using logging_module of DBE_STATS at the DEBUG2 level in logs.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
-- Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_lock;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA=dbe_stats_lock;
SET
gaussdb=# CREATE TABLE t1(a int,b int) distribute by hash(a);
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES(generate_series(1,100),1);
INSERT 0 100
gaussdb=# ANALYZE t1;
ANALYZE

-- After locking a column, check the locking status of the column.
gaussdb=# CALL
DBE_STATS.LOCK_COLUMN_STATS(ownname=>'dbe_stats_lock',tabname=>'t1',colname=>'a');
 lock_column_stats
-------------------

(1 row)

gaussdb=# SELECT staattnum,stastate FROM PG_STATISTIC WHERE starelid='t1'::REGCLASS ORDER BY
1,2;
 staattnum | stastate
-----------+----------
         1 | l
         2 | u
(2 rows)

-- Drop the table and namespace.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_lock;
DROP SCHEMA
```

- DBE_STATS.LOCK_SCHEMA_STATS

  Locks statistics about all tables and their indexes in a specified schema. This API does not return any value.

  The prototype of DBE_STATS.LOCK_SCHEMA_STATS is as follows:

```
DBE_STATS.LOCK_SCHEMA_STATS(
   ownname    VARCHAR2
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-373 DBE_STATS.LOCK_SCHEMA_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	varchar2	Yes	Name of the specified schema. When it is null, the current schema is used by default.

NOTE

If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_lock_schema CASCADE;
NOTICE:  schema "dbe_stats_lock_schema" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table.
gaussdb=# CREATE SCHEMA dbe_stats_lock_schema;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_lock_schema;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by hash(a);
CREATE TABLE
gaussdb=# CREATE TABLE t2(a int, b int) distribute by hash(a);
CREATE TABLE
gaussdb=# -- Lock the schema.
gaussdb=# CALL DBE_STATS.LOCK_SCHEMA_STATS(ownname=>'dbe_stats_lock_schema');
 lock_schema_stats
-------------------

(1 row)

gaussdb=# -- Verification: Both t1 and t2 should be locked (exist_lock set to t).
gaussdb=# SELECT relname, instr(reloptions::text,'stat_state=locked',1,1) <> 0 AS exist_lock
gaussdb-# FROM PG_CLASS
gaussdb-# WHERE relname IN ('t1','t2')
gaussdb-#   AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_lock_schema')
gaussdb-# ORDER BY 1;
 relname | exist_lock
---------+------------
 t1      | t
 t2      | t
(2 rows)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP TABLE t2;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_lock_schema;
DROP SCHEMA
```

- DBE_STATS.UNLOCK_TABLE_STATS

  Unlocks the statistics of a table. After the statistics are unlocked, they can be updated. The indexes of the table will be unlocked as well. This API does not return any value.

  The prototype of DBE_STATS.UNLOCK_TABLE_STATS is as follows:

```
DBE_STATS.UNLOCK_TABLE_STATS(
   ownname   VARCHAR2,
   tabname   VARCHAR2
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-374 DBE_STATS.UNLOCK_TABLE_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	varchar2	Yes	Name of the schema to which the table to be unlocked belongs. When it is null, the current schema is used by default.
tabname	varchar2	No	Name of the table to be unlocked.

NOTE

- You can view **stat_state** in **reloptions** of the table to obtain the lock status of the table.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_lock_unlock_table CASCADE;
NOTICE:  schema "dbe_stats_lock_unlock_table" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table.
gaussdb=# CREATE SCHEMA dbe_stats_lock_unlock_table;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_lock_unlock_table;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by hash(a);
CREATE TABLE
gaussdb=# -- Lock the table first.
gaussdb=# CALL DBE_STATS.LOCK_TABLE_STATS(ownname=>'dbe_stats_lock_unlock_table',
tabname=>'t1');
 lock_table_stats
------------------

(1 row)

gaussdb=# -- Verify that the table has been locked.
gaussdb=# SELECT relname, instr(reloptions::text,'stat_state=locked',1,1) <> 0 AS exist_lock
gaussdb-# FROM PG_CLASS
gaussdb-# WHERE relname='t1'
gaussdb-#   AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_lock_unlock_table');
 relname | exist_lock
---------+------------
 t1      | t
(1 row)

gaussdb=# -- Unlock the table.
gaussdb=# CALL DBE_STATS.UNLOCK_TABLE_STATS(ownname=>'dbe_stats_lock_unlock_table',
tabname=>'t1');
 unlock_table_stats
--------------------

(1 row)

gaussdb=# -- Verify that the table has been unlocked (exist_lock set to f).
gaussdb=# SELECT relname, instr(reloptions::text,'stat_state=locked',1,1) <> 0 AS exist_lock
gaussdb-# FROM PG_CLASS
gaussdb-# WHERE relname='t1'
gaussdb-#   AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
nspname='dbe_stats_lock_unlock_table');
 relname | exist_lock
---------+------------
 t1      | f
(1 row)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_lock_unlock_table;
DROP SCHEMA
```

- DBE_STATS.UNLOCK_PARTITION_STATS

  Unlocks the statistics of a partition. After the partition is unlocked, the statistics of the partition can be updated. The indexes of the partition are unlocked as well. This API does not return any value.

  The prototype of DBE_STATS.UNLOCK_PARTITION_STATS is as follows:

```
DBE_STATS.UNLOCK_PARTITION_STATS(
   ownname     VARCHAR2,
   tabname     VARCHAR2,
   partname    VARCHAR2
);
```

Table 2-375 DBE_STATS.UNLOCK_PARTITION_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	varchar2	Yes	Name of the schema to which the partition to be unlocked belongs. When it is null, the current schema is used by default.
tabname	varchar2	No	Name of the table to which the partition to be unlocked belongs.
partname	varchar2	No	Name of the partition to be unlocked.

NOTE

- You can view **stat_state** in **reloptions** of the partition to obtain the lock status of the table.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_lock_unlock_partition CASCADE;
NOTICE:  schema "dbe_stats_lock_unlock_partition" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a partitioned table.
gaussdb=# CREATE SCHEMA dbe_stats_lock_unlock_partition;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_lock_unlock_partition;
SET
gaussdb=# CREATE TABLE upart_table(a int, b int, c int) distribute by hash(a) PARTITION BY
RANGE(a)
gaussdb-# (
gaussdb(#     PARTITION p1 VALUES LESS THAN(1200),
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb(#     PARTITION p2 VALUES LESS THAN(2400),
gaussdb(#     PARTITION p3 VALUES LESS THAN(MAXVALUE)
gaussdb(# );
CREATE TABLE
gaussdb=# -- Lock p1 first.
gaussdb=# CALL DBE_STATS.LOCK_PARTITION_STATS(ownname=>'dbe_stats_lock_unlock_partition',
tabname=>'upart_table', partname=>'p1');
 lock_partition_stats
----------------------


(1 row)

gaussdb=# -- Verify that p1 has been locked.
gaussdb=# SELECT relname, instr(reloptions::text,'stat_state=locked',1,1) <> 0 AS exist_lock
gaussdb-# FROM PG_PARTITION
gaussdb-# WHERE parentid='upart_table'::REGCLASS
gaussdb-# ORDER BY 1;
   relname   | exist_lock
-------------+------------
 p1          | t
 p2          | f
 p3          | f
 upart_table | f
(4 rows)

gaussdb=# -- Unlock p1.
gaussdb=# CALL
DBE_STATS.UNLOCK_PARTITION_STATS(ownname=>'dbe_stats_lock_unlock_partition',
tabname=>'upart_table', partname=>'p1');
 unlock_partition_stats
------------------------


(1 row)

gaussdb=# -- Verify that p1 has been unlocked (exist_lock set to f).
gaussdb=# SELECT relname, instr(reloptions::text,'stat_state=locked',1,1) <> 0 AS exist_lock
gaussdb-# FROM PG_PARTITION
gaussdb-# WHERE parentid='upart_table'::REGCLASS
gaussdb-# ORDER BY 1;
   relname   | exist_lock
-------------+------------
 p1          | f
 p2          | f
 p3          | f
 upart_table | f
(4 rows)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE upart_table;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_lock_unlock_partition;
DROP SCHEMA
```

- DBE_STATS.UNLOCK_COLUMN_STATS

  Unlocks the statistics of a column. After the column is unlocked, the statistics of the column can be updated. This API does not return any value.

  The prototype of DBE_STATS.UNLOCK_COLUMN_STATS is as follows:

```
DBE_STATS.UNLOCK_COLUMN_STATS(
   ownname     VARCHAR2,
   tabname     VARCHAR2,
   colname     VARCHAR2,
   partname    VARCHAR2 DEFAULT NULL
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-376 DBE_STATS.UNLOCK_COLUMN_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	varchar2	Yes	Name of the schema to which the column to be unlocked belongs. When it is null, the current schema is used by default.
tabname	varchar2	No	Name of the table to which the column to be unlocked belongs.
colname	varchar2	No	Name of the column to be unlocked.
partname	varchar2	Yes	Name of the partition to which the column to be unlocked belongs. The default value is **NULL**.

NOTE

- If **partname** is **NULL**, the list-level statistics and column-level statistics of the cascaded partition are unlocked by default.
- You can view **stastate** in **PG_STATISTIC** or **PG_STATISTIC_EXT** to obtain the lock status.
- Before unlocking multiple columns, query the **staextname** column in **PG_STATISTIC_EXT** to obtain the aliases of multiple columns and pass them as the input parameter **colname**.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_lock_unlock_column CASCADE;
NOTICE:  schema "dbe_stats_lock_unlock_column" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_lock_unlock_column;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_lock_unlock_column;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by hash(a);
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES(generate_series(1,100), 1);
INSERT 0 100
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- Lock column a first.
gaussdb=# CALL DBE_STATS.LOCK_COLUMN_STATS(ownname=>'dbe_stats_lock_unlock_column',
tabname=>'t1', colname=>'a');
 lock_column_stats
-------------------

(1 row)

gaussdb=# -- Verify that column a is locked (stastate set to 'l').
gaussdb=# SELECT staattnum, stastate
gaussdb-# FROM PG_STATISTIC
gaussdb-# WHERE starelid='t1'::REGCLASS
gaussdb-# ORDER BY 1, 2;
 staattnum | stastate
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-----------+----------
         1 | l
         2 | u
(2 rows)

gaussdb=# -- Unlock column a.
gaussdb=# CALL DBE_STATS.UNLOCK_COLUMN_STATS(ownname=>'dbe_stats_lock_unlock_column',
tabname=>'t1', colname=>'a');
 unlock_column_stats
---------------------


(1 row)

gaussdb=# -- Verify that column a has been unlocked (stastate set to 'u').
gaussdb=# SELECT staattnum, stastate
gaussdb-# FROM PG_STATISTIC
gaussdb-# WHERE starelid='t1'::REGCLASS
gaussdb-# ORDER BY 1, 2;
 staattnum | stastate
-----------+----------
         1 | u
         2 | u
(2 rows)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_lock_unlock_column;
DROP SCHEMA
```

- DBE_STATS.UNLOCK_SCHEMA_STATS

  Unlocks statistics about all tables and their indexes in a specified schema. This API does not return any value.

  The prototype of DBE_STATS.UNLOCK_SCHEMA_STATS is as follows:

```
DBE_STATS.UNLOCK_SCHEMA_STATS(
   ownname    VARCHAR2
);
```

Table 2-377 DBE_STATS.UNLOCK_SCHEMA_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	varchar2	Yes	Name of the specified schema. When it is null, the current schema is used by default.

NOTE

If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_lock_unlock_schema CASCADE;
NOTICE:  schema "dbe_stats_lock_unlock_schema" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table.
gaussdb=# CREATE SCHEMA dbe_stats_lock_unlock_schema;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_lock_unlock_schema;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by hash(a);
CREATE TABLE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CREATE TABLE t2(a int, b int) distribute by hash(a);
CREATE TABLE
gaussdb=# -- Lock the schema first.
gaussdb=# CALL DBE_STATS.LOCK_SCHEMA_STATS(ownname=>'dbe_stats_lock_unlock_schema');
 lock_schema_stats
-------------------

(1 row)

gaussdb=# -- Verify that the schema has been locked (exist_lock set to t).
gaussdb=# SELECT relname, instr(reloptions::text,'stat_state=locked',1,1) <> 0 AS exist_lock
gaussdb-# FROM PG_CLASS
gaussdb-# WHERE relname IN ('t1','t2')
gaussdb-#   AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_lock_unlock_schema')
gaussdb-# ORDER BY 1;
 relname | exist_lock
---------+------------
 t1      | t
 t2      | t
(2 rows)

gaussdb=# -- Unlock the schema.
gaussdb=# CALL DBE_STATS.UNLOCK_SCHEMA_STATS(ownname=>'dbe_stats_lock_unlock_schema');
 unlock_schema_stats
---------------------

(1 row)

gaussdb=# -- Verify that the table has been unlocked (exist_lock set to f).
gaussdb=# SELECT relname, instr(reloptions::text,'stat_state=locked',1,1) <> 0 AS exist_lock
gaussdb-# FROM PG_CLASS
gaussdb-# WHERE relname IN ('t1','t2')
gaussdb-#   AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_lock_unlock_schema')
gaussdb-# ORDER BY 1;
 relname | exist_lock
---------+------------
 t1      | f
 t2      | f
(2 rows)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP TABLE t2;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_lock_unlock_schema;
DROP SCHEMA
```

- DBE_STATS.RESTORE_TABLE_STATS

  Rolls back table statistics to a specified time point. The last statistics collected before the time point are loaded to the system catalog, and the statistics of the lower-level partitions, indexes, and columns that are not locked are cascaded. This API does not return any value.

  The prototype of DBE_STATS.RESTORE_TABLE_STATS is as follows:

```
DBE_STATS.RESTORE_TABLE_STATS(
   ownname                 VARCHAR2,
   tabname                 VARCHAR2,
   as_of_timestamp         TIMESTAMPTZ,
   restore_cluster_index   BOOLEAN DEFAULT NULL,
   force                   BOOLEAN DEFAULT FALSE,
   no_invalidate           BOOLEAN DEFAULT NULL
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-378 DBE_STATS.RESTORE_TABLE_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	Yes	Name of the schema to which the table to be rolled back belongs. When it is null, the current schema is used by default.
tabname	VARCHAR2	No	Name of the table to be rolled back.
as_of_timestamp	timestamp with time zone	No	Target time point to be rolled back.
restore_cluster_index	BOOLEAN	Yes	Not supported currently.
force	BOOLEAN	Yes	Determines whether to forcibly roll back the statistics when the statistics are locked. The default value is **false**.
no_invalidate	BOOLEAN	Yes	Not supported currently.

NOTE

- When this API is called, the statistics of its lower-level partitions, indexes, and columns are rolled back as well. If the lower-level partitions or columns are locked, the statistics are skipped.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_restore_table CASCADE;
NOTICE:  schema "dbe_stats_restore_table" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table and collect statistics twice.
gaussdb=# CREATE SCHEMA dbe_stats_restore_table;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_restore_table;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by replication;
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- View the historical table (there should be two records: reltuples=3 and reltuples=6).
gaussdb=# SELECT relname, reltuples FROM GS_TABLESTATS_HISTORY WHERE relid='t1'::regclass
ORDER BY 1, 2;
 relname | reltuples
---------+-----------
 t1      |         3
 t1      |         6
(2 rows)

gaussdb=# -- View the current system catalog (reltuples=6).
gaussdb=# SELECT relname, reltuples FROM PG_CLASS
gaussdb-# WHERE relname='t1'
gaussdb-#   AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_restore_table');
 relname | reltuples
---------+-----------
 t1      |         6
(1 row)

gaussdb=# -- Roll back to the earliest statistics.
gaussdb=# CALL DBE_STATS.RESTORE_TABLE_STATS(
gaussdb(#     ownname=>'dbe_stats_restore_table',
gaussdb(#     tabname=>'t1',
gaussdb(#     as_of_timestamp=>(SELECT MIN(reltimestamp) FROM GS_TABLESTATS_HISTORY
WHERE relname='t1'
gaussdb(#     AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_restore_table'))
gaussdb(# );
 restore_table_stats
---------------------

(1 row)

gaussdb=# -- Verification: reltuples = 3 after the rollback.
gaussdb=# SELECT relname, reltuples FROM PG_CLASS
gaussdb-# WHERE relname='t1'
gaussdb-#   AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_restore_table');
 relname | reltuples
---------+-----------
 t1      |         3
(1 row)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_restore_table;
DROP SCHEMA
```

- DBE_STATS.RESTORE_PARTITION_STATS

  Rolls back partition statistics to a specified time point. The last statistics collected before the time point are loaded to the system catalog, and the statistics of the lower-level partitions, indexes, and columns that are not locked are cascaded. This API does not return any value.

  The prototype of DBE_STATS.RESTORE_PARTITION_STATS is as follows:

```
DBE_STATS.RESTORE_PARTITION_STATS(
    ownname                 VARCHAR2,
    tabname                 VARCHAR2,
    partname                VARCHAR2,
    as_of_timestamp           TIMESTAMPTZ,
    force                   BOOLEAN DEFAULT FALSE,
    no_invalidate           BOOLEAN DEFAULT NULL
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-379 DBE_STATS.RESTORE_PARTITION_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	Yes	Name of the schema to which the partition to be rolled back belongs.
tabname	VARCHAR2	No	Name of the table to which the partition to be rolled back belongs.
partname	VARCHAR2	No	Name of the partition to be rolled back.
as_of_timestamp	timestamp with time zone	No	Target time point to be rolled back.
force	BOOLEAN	Yes	Determines whether to forcibly roll back the statistics when the statistics are locked.
no_invalidate	BOOLEAN	Yes	Not supported currently.

NOTE

- When this API is called, the statistics of its lower-level partitions, indexes, and columns are rolled back.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_restore_partition CASCADE;
NOTICE:  schema "dbe_stats_restore_partition" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a partitioned table and collect partition statistics twice.
gaussdb=# CREATE SCHEMA dbe_stats_restore_partition;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_restore_partition;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by replication PARTITION BY RANGE(a)
gaussdb-# (
gaussdb(#     PARTITION p1 VALUES LESS THAN(10),
gaussdb(#     PARTITION p2 VALUES LESS THAN(MAXVALUE)
gaussdb(# );
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# ANALYZE t1 PARTITION (p1);
ANALYZE
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
gaussdb=# ANALYZE t1 PARTITION (p1);
ANALYZE
gaussdb=# -- View the historical table (there should be historical records of partitions).
gaussdb=# SELECT relname, relkind, reltuples FROM GS_TABLESTATS_HISTORY
gaussdb-# WHERE relname='p1' AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_restore_partition')
gaussdb-# ORDER BY reltuples;
 relname | relkind | reltuples
---------+---------+-----------
 p1      | p       |         3
 p1      | p       |         6
(2 rows)

gaussdb=# -- View the statistics of the current partition (reltuples=6).
gaussdb=# SELECT relname, reltuples FROM PG_PARTITION
gaussdb-# WHERE relname='p1' AND parentid = (SELECT oid FROM PG_CLASS WHERE relname='t1'
AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_restore_partition'));
 relname | reltuples
---------+-----------
 p1      |         6
(1 row)

gaussdb=# -- Roll back partition p1 to the earliest statistics.
gaussdb=# CALL DBE_STATS.RESTORE_PARTITION_STATS(
gaussdb(#     ownname=>'dbe_stats_restore_partition',
gaussdb(#     tabname=>'t1',
gaussdb(#     partname=>'p1',
gaussdb(#     as_of_timestamp=>(SELECT MIN(reltimestamp) FROM GS_TABLESTATS_HISTORY
WHERE relname='p1' AND
gaussdb(#     relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_restore_partition'))
gaussdb(# );
 restore_partition_stats
-------------------------

(1 row)

gaussdb=# -- Verify the rollback result (reltuples should be rolled back to 3).
gaussdb=# SELECT relname, reltuples FROM PG_PARTITION
gaussdb-# WHERE relname='p1' AND parentid = (SELECT oid FROM PG_CLASS WHERE relname='t1'
AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_restore_partition'));
 relname | reltuples
---------+-----------
 p1      |         3
(1 row)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_restore_partition;
DROP SCHEMA
```

- DBE_STATS.RESTORE_COLUMN_STATS

  Rolls back the column statistics to a specified time point. The last statistics collected before the time point to the system catalog are loaded. This API does not return any value.

  The prototype of DBE_STATS.RESTORE_COLUMN_STATS is as follows:

```
DBE_STATS.RESTORE_COLUMN_STATS(
    ownname                 VARCHAR2,
    tabname                 VARCHAR2,
    colname                 VARCHAR2,
    partname                VARCHAR2 DEFAULT NULL,
    as_of_timestamp           TIMESTAMPTZ,
    force                   BOOLEAN DEFAULT FALSE,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    no_invalidate           BOOLEAN DEFAULT NULL
);
```

Table 2-380 DBE_STATS.RESTORE_COLUMN_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	Yes	Name of the schema to which the column to be rolled back belongs.
tabname	VARCHAR2	No	Name of the table to which the column to be rolled back belongs.
colname	VARCHAR2	No	Name of the column to be rolled back.
partname	VARCHAR2	Yes	Name of the partition to which the column to be rolled back belongs.
as_of_timestamp	timestamp with time zone	No	Target time point to be rolled back.
force	BOOLEAN	Yes	Determines whether to forcibly roll back the statistics when the statistics are locked.
no_invalidate	BOOLEAN	Yes	Not supported currently.

NOTE

- If **partname** is **NULL**, the list-level statistics are rolled back by default.
- Before rolling back multiple columns, query the **staextname** column in **PG_STATISTIC_EXT** to obtain the aliases of multiple columns and pass them as the input parameter **colname**.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_restore_column CASCADE;
NOTICE:  schema "dbe_stats_restore_column" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table and collect statistics twice.
gaussdb=# CREATE SCHEMA dbe_stats_restore_column;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_restore_column;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by replication;
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- Ensure that the interval between two ANALYZE timestamps is greater than 1 second.
gaussdb=# SELECT pg_sleep(1.1);
 pg_sleep
----------

(1 row)

gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- View the statistics in the history table.
gaussdb=# SELECT staattnum, stadistinct, stakind1, stanumbers1, stavalues1
gaussdb-# FROM GS_STATISTIC_HISTORY
gaussdb-# WHERE starelid='t1'::REGCLASS
gaussdb-# ORDER BY statimestamp;
 staattnum | stadistinct | stakind1 | stanumbers1 | stavalues1
-----------+-------------+----------+-------------+------------
         1 |    -.333333 |        1 | {1}         | {1}
         2 |    -.333333 |        1 | {1}         | {1}
         1 |    -.333333 |        1 | {.5,.5}     | {1,2}
         2 |    -.333333 |        1 | {.5,.5}     | {1,2}
(4 rows)

gaussdb=# SELECT staattnum, stadistinct, stakind1, stanumbers1, stavalues1
gaussdb-# FROM PG_STATISTIC
gaussdb-# WHERE starelid='t1'::REGCLASS
gaussdb-# ORDER BY 1;
 staattnum | stadistinct | stakind1 | stanumbers1 | stavalues1
-----------+-------------+----------+-------------+------------
         1 |    -.333333 |        1 | {.5,.5}     | {1,2}
         2 |    -.333333 |        1 | {.5,.5}     | {1,2}
(2 rows)

gaussdb=# -- Roll back column a to an earlier time point.
gaussdb=# CALL DBE_STATS.RESTORE_COLUMN_STATS(
gaussdb(#     ownname=>'dbe_stats_restore_column',
gaussdb(#     tabname=>'t1',
gaussdb(#     colname=>'a',
gaussdb(#     as_of_timestamp=>((SELECT MIN(reltimestamp) FROM GS_TABLESTATS_HISTORY
WHERE relname='t1'
gaussdb(#     AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_restore_column')) + INTERVAL '1 second')
gaussdb(# );
 restore_column_stats
----------------------

(1 row)

gaussdb=# -- Verify the statistics of column a after the rollback (stanumbers1={1}, stavalues1={1}).
gaussdb=# SELECT staattnum, stadistinct, stakind1, stanumbers1, stavalues1
gaussdb-# FROM PG_STATISTIC
gaussdb-# WHERE starelid='t1'::REGCLASS
gaussdb-# ORDER BY 1;
 staattnum | stadistinct | stakind1 | stanumbers1 | stavalues1
-----------+-------------+----------+-------------+------------
         1 |    -.333333 |        1 | {1}         | {1}
         2 |    -.333333 |        1 | {.5,.5}     | {1,2}
(2 rows)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# DROP SCHEMA dbe_stats_restore_column;
DROP SCHEMA
```

- DBE_STATS.RESTORE_SCHEMA_STATS

  Rolls back statistics of all tables in a specified schema to a specified time point. This API does not return any value.

  The prototype of DBE_STATS.RESTORE_SCHEMA_STATS is as follows:

```
DBE_STATS.RESTORE_SCHEMA_STATS(
   ownname                VARCHAR2,
   as_of_timestamp          TIMESTAMPTZ,
   force                BOOLEAN DEFAULT FALSE,
   no_invalidate            BOOLEAN DEFAULT NULL
);
```

Table 2-381 DBE_STATS.RESTORE_SCHEMA_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	Yes	Specified schema name.
as_of_timestamp	timestamp with time zone	No	Target time point to be rolled back.
force	BOOLEAN	Yes	Determines whether to forcibly roll back the statistics when the statistics are locked.
no_invalidate	BOOLEAN	Yes	Not supported currently.

NOTE

If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_restore_schema CASCADE;
NOTICE:  schema "dbe_stats_restore_schema" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table, insert data, and collect statistics twice.
gaussdb=# CREATE SCHEMA dbe_stats_restore_schema;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_restore_schema;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by replication;
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- View the history tables.
gaussdb=# SELECT relname, reltuples FROM GS_TABLESTATS_HISTORY WHERE relid='t1'::regclass
ORDER BY 1, 2;
 relname | reltuples
---------+-----------
 t1      |         3
 t1      |         6
(2 rows)

gaussdb=# -- View the current system catalog (reltuples=6).
gaussdb=# SELECT relname, reltuples FROM PG_CLASS
gaussdb-# WHERE relname='t1'
gaussdb-#   AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_restore_schema');
 relname | reltuples
---------+-----------
 t1      |         6
(1 row)

gaussdb=# -- Roll back the schema.
gaussdb=# CALL DBE_STATS.RESTORE_SCHEMA_STATS(
gaussdb(#     ownname=>'dbe_stats_restore_schema',
gaussdb(#     as_of_timestamp=>(SELECT MIN(reltimestamp) FROM GS_TABLESTATS_HISTORY
WHERE relname='t1'
gaussdb(#     AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_restore_schema'))
gaussdb(# );
 restore_schema_stats
----------------------

(1 row)

gaussdb=# -- Verification: reltuples = 3 after the rollback.
gaussdb=# SELECT relname, reltuples FROM PG_CLASS
gaussdb-# WHERE relname='t1'
gaussdb-#   AND relnamespace = (SELECT oid FROM PG_NAMESPACE WHERE
nspname='dbe_stats_restore_schema');
 relname | reltuples
---------+-----------
 t1      |         3
(1 row)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_restore_schema;
DROP SCHEMA
```

- DBE_STATS.PURGE_STATS

  Clears all historical statistics before a specified time point. This API does not return any value.

  The prototype of DBE_STATS.PURGE_STATS is as follows:

  ```
  DBE_STATS.PURGE_STATS(
     before_timestamp       TIMESTAMPTZ
  );
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-382 DBE_STATS.PURGE_STATS API parameters

Parameter	Type	Can Be Null	Description
before_timestamp	timestamp with time zone	No	Time node for clearing.

Example:

```
-- Create a table and collect statistics twice.
gaussdb=# CREATE SCHEMA dbe_stats_purge;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA=dbe_stats_purge;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by replication;
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(1,1);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES(2,2);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE

-- View history tables.
gaussdb=# SELECT relname,reltuples FROM GS_TABLESTATS_HISTORY WHERE relname='t1' ORDER
BY 1,2;
 relname | reltuples
---------+-----------
 t1      |         3
 t1      |         6
(2 rows)

-- Clear the earlier history statistics and view the history table.
gaussdb=# CALL DBE_STATS.PURGE_STATS(before_timestamp=>((SELECT MIN(reltimestamp) FROM
GS_TABLESTATS_HISTORY WHERE relname='t1') + INTERVAL '1 second'));
 purge_stats
-------------

(1 row)

gaussdb=# SELECT relname,reltuples FROM GS_TABLESTATS_HISTORY WHERE relname='t1';
 relname | reltuples
---------+-----------
 t1      |         6
(1 row)

-- Drop the table and namespace.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_purge;
DROP SCHEMA
```

- DBE_STATS.GET_STATS_HISTORY_RETENTION

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Obtains the retention time of historical statistics. The return value is the value of the GUC parameter **stats_history_retention_time**, indicating the retention time of historical statistics. This API does not have input parameters.

The prototype of DBE_STATS.GET_STATS_HISTORY_RETENTION is as follows:

```
DBE_STATS.GET_STATS_HISTORY_RETENTION()
returns NUMBER;
```

Example:

```
gaussdb=# CALL DBE_STATS.GET_STATS_HISTORY_RETENTION();
 get_stats_history_retention
-----------------------------
                          31
(1 row)
```

- DBE_STATS.GET_STATS_HISTORY_AVAILABILITY

  Obtains the earliest available time of the current historical statistics. The earliest time of all historical statistics in the current database is returned. This API does not have input parameters.

  The prototype of DBE_STATS.GET_STATS_HISTORY_AVAILABILITY is as follows:

```
DBE_STATS.GET_STATS_HISTORY_AVAILABILITY()
returns TIMESTAMPTZ;
```

  Example:

```
gaussdb=# ANALYZE;
ANALYZE
gaussdb=# CALL DBE_STATS.GET_STATS_HISTORY_AVAILABILITY();
 get_stats_history_availability
--------------------------------
 2025-05-07 05:11:43.584349-04
(1 row)
```

  NOTE

  The DBE_STATS advanced package API is valid only for ordinary tables.

- DBE_STATS.CREATE_STAT_TABLE

  Creates a statistics table for storing user statistics. This API does not return any value.

  The prototype of DBE_STATS.CREATE_STAT_TABLE is as follows:

```
DBE_STATS.CREATE_STAT_TABLE (
   ownname          VARCHAR2,
   stattab          VARCHAR2,
   tblspace         VARCHAR2 DEFAULT NULL,
   global_temporary BOOLEAN DEFAULT NULL
);
```

Table 2-383 DBE_STATS.CREATE_STAT_TABLE API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	No	Schema where the created statistics table is located. When it is null, the current schema is used by default.
stattab	VARCHAR2	No	Name of a statistics table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Can Be Null	Description
tblspace	VARCHAR2	Yes	Tablespace where the created statistics table is located. If not specified, the default tablespace is used.
global_temporary	BOOLEAN	Yes	Not supported currently.

NOTE

- To distinguish ordinary tables from statistics tables, the **statstable** column in **reloptions** of a statistics table is not an empty string after the statistics table is created. Considering that the structure of the statistics table may be upgraded in the future, to implement compatibility with earlier versions, **statstable** is assigned the version information of the current statistics table.
- When this table is created, the **gs_stattab_***<name_hash_value>***_statid_type_nameinfo_index** index is automatically generated for index scan. *<name_hash_value>* indicates the hash value of the statistics table name. If the index is deleted, the export and import APIs are unavailable.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
-- Create a stattab statistics table in dbe_stats_create_drop_schema and manually insert a data record. You can query the data record in the new user statistics table.
gaussdb=# CREATE SCHEMA dbe_stats_create_drop_schema;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_create_drop_schema;
SET
gaussdb=# CALL DBE_STATS.CREATE_STAT_TABLE(ownname => 'dbe_stats_create_drop_schema', stattab => 'stat_tab');
 create_stat_table
-------------------

(1 row)
gaussdb=# INSERT INTO stat_tab(nodeid, statid, type, relname) VALUES(0, '123', 't', 't1');
INSERT 0 1
gaussdb=# SELECT statid, type, relname FROM stat_tab;
 statid | type | relname
--------+------+---------
 123    | t    | t1
(1 row)
```

- DBE_STATS.DROP_STAT_TABLE

  Deletes the user statistics table. This API does not return any value.

  The prototype of DBE_STATS.DROP_STAT_TABLE is as follows:

```
DBE_STATS.DROP_STAT_TABLE (
   ownname VARCHAR2,
   stattab VARCHAR2
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-384 DBE_STATS.DROP_STAT_TABLE API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	No	Schema where the statistics table is located. When it is null, the current schema is used by default.
stattab	VARCHAR2	No	Name of a statistics table.

NOTE

- Ordinary tables whose **statstable** attribute is empty cannot be deleted using this function.
- If an index is created on the user statistics table or other objects that depend on the table are created, all objects that depend on the table are deleted when this function is called.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
-- Delete the created user statistics table and the new schema.
gaussdb=# CALL DBE_STATS.DROP_STAT_TABLE(ownname => 'dbe_stats_create_drop_schema',
stattab => 'stat_tab');
 drop_stat_table
-----------------
(1 row)
gaussdb=# DROP SCHEMA dbe_stats_create_drop_schema CASCADE;
DROP SCHEMA
```

- DBE_STATS.EXPORT_INDEX_STATS

  Searches the system catalog for the statistics of the specified index and stores the result in the user statistics table. This API does not return any value.

  The prototype of DBE_STATS.EXPORT_INDEX_STATS is as follows:

```
DBE_STATS.EXPORT_INDEX_STATS (
   ownname  VARCHAR2,
   indname  VARCHAR2,
   partname VARCHAR2 DEFAULT NULL,
   stattab  VARCHAR2,
   statid   VARCHAR2 DEFAULT NULL,
   statown  VARCHAR2 DEFAULT NULL
);
```

Table 2-385 DBE_STATS.EXPORT_INDEX_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	No	Name of the schema where the index is located. When it is null, the current schema is used by default.
indname	VARCHAR2	No	Name of the index whose statistics are to be exported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Can Be Null	Description
partname	VARCHAR2	Yes	Name of a partitioned index. If the index has been partitioned but **partname** is **null**, global and partitioned index statistics are exported.
stattab	VARCHAR2	No	Name of the user table that stores statistics.
statid	VARCHAR2	Yes	(Optional) Identifier that points to statistics in **stattab**.
statown	VARCHAR2	Yes	Name of the schema where the user table that stores statistics is located. If this parameter is set to **null**, the value of **ownname** is used.

NOTE

- You need to export the reltuples (numrows), relpages (numlblks), and relallvisible (relallvisible) statistics of indexes.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
-- Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_export;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_export;
SET
gaussdb=# CREATE TABLE t2(a int, b int, c int) distribute by replication PARTITION BY RANGE(a)
SUBPARTITION BY RANGE(b)
 (
   PARTITION p1 VALUES LESS THAN(200)
   (SUBPARTITION s1 VALUES LESS THAN (500),
    SUBPARTITION s2 VALUES LESS THAN (MAXVALUE)
   ),
   PARTITION p2 VALUES LESS THAN(500)
   (SUBPARTITION s3 VALUES LESS THAN (500),
    SUBPARTITION s4 VALUES LESS THAN (MAXVALUE)
   ),
   PARTITION p3 VALUES LESS THAN(MAXVALUE)
   ( SUBPARTITION s5 VALUES LESS THAN (500),
     SUBPARTITION s6 VALUES LESS THAN (MAXVALUE)
   )
 );
CREATE TABLE
gaussdb=# CREATE INDEX idx_t2_local_a ON t2(a) LOCAL
 (
    PARTITION p1_idx_a
    (
       SUBPARTITION s1_idx_a,
       SUBPARTITION s2_idx_a
    ),
    PARTITION p2_idx_a
    (
       SUBPARTITION s3_idx_a,
       SUBPARTITION s4_idx_a
    ),
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    PARTITION p3_idx_a
    (
        SUBPARTITION s5_idx_a,
        SUBPARTITION s6_idx_a
    ));
CREATE INDEX
gaussdb=# INSERT INTO t2 VALUES (100, 300, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (100, 600, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (300, 300, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (300, 600, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (600, 400, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (600, 600, 1);
INSERT 0 1
gaussdb=# ANALYZE t2 WITH ALL;
ANALYZE
gaussdb=# ANALYZE t2((a,b)) WITH ALL;
ANALYZE
-- Create a user statistics table as the target table for exporting statistics.
gaussdb=# CALL DBE_STATS.CREATE_STAT_TABLE('dbe_stats_export', 'export_table');
 create_stat_table
-------------------
(1 row)

-- Export index statistics together with partitioned index statistics.
-- View the current statistics.
gaussdb=# SELECT reltuples FROM PG_CLASS WHERE relname='idx_t2_local_a';
 reltuples
-----------
         6
(1 row)
gaussdb=# SELECT relname,relpages,reltuples FROM PG_PARTITION WHERE
parentid='idx_t2_local_a'::REGCLASS ORDER BY oid;
 relname  | relpages | reltuples
----------+----------+-----------
 s1_idx_a |        1 |         0
 s2_idx_a |        1 |         0
 s3_idx_a |        1 |         0
 s4_idx_a |        1 |         0
 s5_idx_a |        1 |         0
 s6_idx_a |        1 |         0
(6 rows)
-- Export the statistics and view the data in the user statistics table.
gaussdb=# CALL DBE_STATS.EXPORT_INDEX_STATS(ownname => 'dbe_stats_export', indname =>
'idx_t2_local_a', partname=> null, stattab => 'export_table', statid => 'idx_s1');
 export_index_stats
--------------------
(1 row)
gaussdb=# SELECT relname,partname,numrows,numblocks FROM export_table WHERE nodeid = 1
ORDER BY relname, partname desc;
    relname     | partname | numrows | numblocks
----------------+----------+---------+-----------
 idx_t2_local_a |          |       6 |         6
 idx_t2_local_a | s6_idx_a |       0 |         1
 idx_t2_local_a | s5_idx_a |       0 |         1
 idx_t2_local_a | s4_idx_a |       0 |         1
 idx_t2_local_a | s3_idx_a |       0 |         1
 idx_t2_local_a | s2_idx_a |       0 |         1
 idx_t2_local_a | s1_idx_a |       0 |         1
(7 rows)
-- Drop the table and namespace.
gaussdb=# DROP INDEX idx_t2_local_a;
DROP INDEX
gaussdb=# DROP TABLE t2;
DROP TABLE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# DROP TABLE export_table;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_export;
DROP SCHEMA
```

- DBE_STATS.EXPORT_TABLE_STATS

  Retrieves statistics for the specified table and stores them in the user statistics table. This API does not return any value.

  The prototype of DBE_STATS.EXPORT_TABLE_STATS is as follows:

```
DBE_STATS.EXPORT_TABLE_STATS (
  ownname          VARCHAR2,
  tabname          VARCHAR2,
  partname         VARCHAR2 DEFAULT NULL,
  stattab          VARCHAR2,
  statid           VARCHAR2 DEFAULT NULL,
  cascade          BOOLEAN  DEFAULT TRUE,
  statown          VARCHAR2 DEFAULT NULL,
  stat_category    VARCHAR2 DEFAULT NULL
);
```

Table 2-386 DBE_STATS.EXPORT_TABLE_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	No	Name of the schema. When it is null, the current schema is used by default.
tabname	VARCHAR2	No	Name of the table whose statistics are to be exported.
partname	VARCHAR2	Yes	Table partition name. If the table has been partitioned but **partname** is **null**, global and partitioned table statistics are exported.
stattab	VARCHAR2	No	Name of the user table that stores statistics.
statid	VARCHAR2	Yes	(Optional) Identifier that points to statistics in **stattab**.
cascade	BOOLEAN	Yes	Specifies whether to export statistics about columns, indexes, multiple columns, and index expressions. The default value is **TRUE**.
statown	VARCHAR2	Yes	Name of the schema where the user table that stores statistics is located. If this parameter is set to **null**, the value of **ownname** is used.
stat_category	VARCHAR2	Yes	Not supported currently.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- You need to export the reltuples (numrows), relpages (numblks), and relallvisible (relallvisible) statistics of tables. For details about how to export column-level and index-level statistics in cascading mode, see the statistics of the **EXPORT_INDEX_STATS** and **EXPORT_COLUMN_STATS** APIs.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_export_table CASCADE;
NOTICE:  schema "dbe_stats_export_table" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_export_table;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_export_table;
SET
gaussdb=# CREATE TABLE t1(a int, b int, c int) distribute by replication;
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES (100, 300, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (150, 550, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (250, 150, 1);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- Create a user statistics table.
gaussdb=# CALL DBE_STATS.CREATE_STAT_TABLE('dbe_stats_export_table', 'export_table');
 create_stat_table
-------------------

(1 row)

gaussdb=# -- Export table statistics (including column and index statistics).
gaussdb=# CALL DBE_STATS.EXPORT_TABLE_STATS(
gaussdb(#     ownname => 'dbe_stats_export_table',
gaussdb(#     tabname => 't1',
gaussdb(#     partname => null,
gaussdb(#     stattab => 'export_table',
gaussdb(#     statid => 't1_t',
gaussdb(#     cascade => true
gaussdb(# );
 export_table_stats
--------------------

(1 row)

gaussdb=# -- Verification: The user statistics table should contain data.
gaussdb=# SELECT count(*) AS row_count FROM export_table WHERE relname='t1' and nodeid=0;
 row_count
-----------
         4
(1 row)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP TABLE export_table;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_export_table;
DROP SCHEMA
```

- DBE_STATS.EXPORT_COLUMN_STATS

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Retrieves statistics for the specified columns (multiple columns or expressions) and stores them in the user statistics table. This API does not return any value.

The prototype of DBE_STATS.EXPORT_COLUMN_STATS is as follows:

```
DBE_STATS.EXPORT_COLUMN_STATS (
   ownname  VARCHAR2,
   tabname  VARCHAR2,
   colname  VARCHAR2,
   partname VARCHAR2 DEFAULT NULL,
   stattab  VARCHAR2,
   statid   VARCHAR2 DEFAULT NULL,
   statown  VARCHAR2 DEFAULT NULL
);
```

Table 2-387 DBE_STATS.EXPORT_COLUMN_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	No	Name of the schema. When it is null, the current schema is used by default.
tabname	VARCHAR2	No	Name of the table (index name corresponding to the expression) to which the column whose statistics are to be exported belongs.
colname	VARCHAR2	No	Column name or multiple column names (If tabname is an index, **colname** is an expression name.)
partname	VARCHAR2	Yes	Partition name. If the table or index has been partitioned but **partname** is **null**, global and partitioned key, multiple columns, and index statistics are exported.
stattab	VARCHAR2	No	Name of the user table that stores statistics.
statid	VARCHAR2	Yes	(Optional) Identifier that points to statistics in **stattab**.
statown	VARCHAR2	Yes	Name of the schema where the user table that stores statistics is located. If this parameter is set to **null**, the value of **ownname** is used.

NOTE

- You can query **PG_STATISTIC** or **PG_STATISTIC_EXT** to export column-level statistics.
- Before exporting the statistics of multiple columns, query the **staextname** column in **PG_STATISTIC_EXT** to obtain the aliases of multiple columns and pass them as the input parameter **colname**.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_export;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA=dbe_stats_export;
SET
gaussdb=# CREATE TABLE IF NOT EXISTS t1 (a int, b int, c int) distribute by replication;
CREATE TABLE
gaussdb=# CREATE INDEX idx_t1_expr on t1((a*a), (a+a), (a*b));
CREATE INDEX
gaussdb=# INSERT INTO t1 VALUES (100, 300, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (150, 550, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (250, 150, 1);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# ANALYZE t1((a,b));
ANALYZE
-- Create a user statistics table.
gaussdb=# CALL DBE_STATS.CREATE_STAT_TABLE('dbe_stats_export', 'export_table');
 create_stat_table
-------------------
(1 row)

-- View column-level statistics.
gaussdb=# SELECT staattnum,stadistinct,stakind1,stanumbers1,stavalues1 FROM PG_STATISTIC
WHERE starelid='t1'::REGCLASS ORDER BY 1;
 staattnum | stadistinct | stakind1 | stanumbers1 |  stavalues1
-----------+-------------+----------+-------------+---------------
         1 |          -1 |        2 |             | {100,150,250}
         2 |          -1 |        2 |             | {150,300,550}
         3 |    -.333333 |        1 | {1}         | {1}
(3 rows)
-- Export column-level statistics and view the user statistics table.
gaussdb=# CALL DBE_STATS.EXPORT_COLUMN_STATS(ownname => 'dbe_stats_export', tabname =>
't1', colname=> 'a', partname => null, stattab => 'export_table', statid => 't1_c');
 export_column_stats
---------------------
(1 row)
gaussdb=# SELECT relname,columnname,stadistinct,stakind1,stanumbers1,stavalues1 FROM
export_table WHERE nodeid=0;
 relname | columnname | stadistinct | stakind1 | stanumbers1 |  stavalues1
---------+------------+-------------+----------+-------------+---------------
 t1      | a          |          -1 |        2 |             | {100,150,250}
(1 row)

-- View expression statistics.
gaussdb=# SELECT staattnum,stadistinct,stakind1,stanumbers1,stavalues1 FROM PG_STATISTIC
WHERE starelid='idx_t1_expr'::REGCLASS ORDER BY 1;
 staattnum | stadistinct | stakind1 | stanumbers1 |     stavalues1
-----------+-------------+----------+-------------+--------------------
         1 |          -1 |        2 |             | {10000,22500,62500}
         2 |          -1 |        2 |             | {200,300,500}
         3 |          -1 |        2 |             | {30000,37500,82500}
(3 rows)
-- Export expression statistics and view the user statistics table.
gaussdb=# CALL DBE_STATS.EXPORT_COLUMN_STATS(ownname => 'dbe_stats_export', tabname =>
'idx_t1_expr', colname => 'expr', stattab => 'export_table', statid => 't1_expr');
 export_column_stats
---------------------
(1 row)
gaussdb=# SELECT relname,columnname,stadistinct,stakind1,stanumbers1,stavalues1 FROM
export_table where statid='t1_expr' AND nodeid=0;
   relname   | columnname | stadistinct | stakind1 | stanumbers1 |     stavalues1
-------------+------------+-------------+----------+-------------+--------------------
 idx_t1_expr | expr       |          -1 |        2 |             | {10000,22500,62500}
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Export the statistics of multiple columns.
-- View the statistics of multiple columns.
gaussdb=# SELECT stakey,staextname,stadistinct,stakind1,stanumbers1,stavalues1 FROM
PG_STATISTIC_EXT WHERE starelid='t1'::REGCLASS;
 stakey |     staextname      | stadistinct | stakind1 | stanumbers1 | stavalues1
--------+---------------------+-------------+----------+-------------+------------
 1 2    | extname_-1308656218 |          -1 |        8 |             |
(1 row)
-- Export and view the user statistics table.
gaussdb=# CALL DBE_STATS.EXPORT_COLUMN_STATS(ownname => 'dbe_stats_export', tabname =>
't1', colname=> 'extname_-1308656218', partname => null, stattab => 'export_table', statid => 't1_c');
 export_column_stats
---------------------
(1 row)
gaussdb=# SELECT relname,columnname,stadistinct,stakind1,stanumbers1,stavalues1 FROM
export_table WHERE nodeid=0 AND relname='t1';
 relname |     columnname      | stadistinct | stakind1 | stanumbers1 |  stavalues1
---------+---------------------+-------------+----------+-------------+---------------
 t1      | a                   |          -1 |        2 |             | {100,150,250}
 t1      | extname_-1308656218 |          -1 |        8 |             |
(2 rows)
-- Drop the table and namespace.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP TABLE export_table;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_export;
DROP SCHEMA
```

- DBE_STATS.EXPORT_SCHEMA_STATS

  Retrieves statistics for the specified schema and stores them in the user statistics table. This API does not return any value.

  The prototype of DBE_STATS.EXPORT_SCHEMA_STATS is as follows:

```
DBE_STATS.EXPORT_SCHEMA_STATS (
   ownname         VARCHAR2,
   stattab         VARCHAR2,
   statid          VARCHAR2 DEFAULT NULL,
   statown         VARCHAR2 DEFAULT NULL,
   stat_category   VARCHAR2 DEFAULT NULL
);
```

Table 2-388 DBE_STATS.EXPORT_SCHEMA_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	No	Name of the schema. When it is null, the current schema is used by default.
stattab	VARCHAR2	No	Name of the user table that stores statistics.
statid	VARCHAR2	Yes	(Optional) Identifier that points to statistics in **stattab**.
statown	VARCHAR2	Yes	Name of the schema where the user table that stores statistics is located. If this parameter is set to **null**, the value of **ownname** is used.
stat_category	VARCHAR2	Yes	Not supported currently.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- You need to export statistics about all tables on which the current user has permission in the specified schema. Tables on which the current user does not have permission are skipped and no error is reported.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_export_schema CASCADE;
NOTICE:  schema "dbe_stats_export_schema" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_export_schema;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_export_schema;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by replication;
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES (1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (2,2);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- Create a user statistics table.
gaussdb=# CALL DBE_STATS.CREATE_STAT_TABLE('dbe_stats_export_schema', 'export_table');
 create_stat_table
-------------------

(1 row)

gaussdb=# -- Export schema statistics.
gaussdb=# CALL DBE_STATS.EXPORT_SCHEMA_STATS(
gaussdb(#     ownname => 'dbe_stats_export_schema',
gaussdb(#     stattab => 'export_table',
gaussdb(#     statid => 's1'
gaussdb(# );
 export_schema_stats
---------------------

(1 row)

gaussdb=# -- Verification: The user statistics table should contain data.
gaussdb=# SELECT relname,numrows FROM export_table where nodeid=0 and numrows != 0;
 relname | numrows
---------+---------
 t1      |       2
(1 row)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP TABLE export_table;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_export_schema;
DROP SCHEMA
```

- DBE_STATS.EXPORT_DATABASE_STATS

  Retrieves statistics for the specified database and stores them in the user statistics table. This API does not return any value.

  The prototype of DBE_STATS.EXPORT_DATABASE_STATS is as follows:

```
DBE_STATS.EXPORT_DATABASE_STATS (
   stattab         VARCHAR2,
   statid          VARCHAR2 DEFAULT NULL,
   statown          VARCHAR2 DEFAULT NULL,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  stat_category   VARCHAR2 DEFAULT NULL
);
```

Table 2-389 DBE_STATS.EXPORT_DATABASE_STATS API parameters

Parameter	Type	Can Be Null	Description
stattab	VARCHAR2	No	Name of the user table that stores statistics.
statid	VARCHAR2	Yes	(Optional) Identifier that points to statistics in **stattab**.
statown	VARCHAR2	Yes	Name of the schema where the user table that stores statistics is located. If this parameter is set to **null**, the current schema is used.
stat_category	VARCHAR2	Yes	Not supported currently.

NOTE

- You can export statistics about all tables on which the current user has the ANALYZE permission in the current database. Tables on which the current user does not have permission are skipped and no error is reported.
- If **statown** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_export_database CASCADE;
NOTICE:  schema "dbe_stats_export_database" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_export_database;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_export_database;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by replication;
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES (1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (2,2);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- Create a user statistics table.
gaussdb=# CALL DBE_STATS.CREATE_STAT_TABLE('dbe_stats_export_database', 'export_table');
 create_stat_table
-------------------

(1 row)

gaussdb=# -- Export database statistics.
gaussdb=# CALL DBE_STATS.EXPORT_DATABASE_STATS(
gaussdb(#     statown => 'dbe_stats_export_database',
gaussdb(#     stattab => 'export_table',
gaussdb(#     statid => 's1'
gaussdb(# );
 export_database_stats
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-----------------------

(1 row)

gaussdb=# -- Verification: The user statistics table should contain data.
gaussdb=# SELECT count(*) > 0 FROM export_table;
 ?column?
----------
 t
(1 row)

gaussdb=# SELECT count(*) > 0 FROM export_table WHERE relname='t1';
 ?column?
----------
 t
(1 row)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP TABLE export_table;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_export_database;
DROP SCHEMA
```

- DBE_STATS.IMPORT_INDEX_STATS

  Retrieves statistics for the specified index from the user statistics table of the **stattab** identifier and writes it back to the system catalog. This API does not return any value.

  The prototype of DBE_STATS.IMPORT_INDEX_STATS is as follows:

```
DBE_STATS.IMPORT_INDEX_STATS (
   ownname       VARCHAR2,
   indname       VARCHAR2,
   partname      VARCHAR2 DEFAULT NULL,
   stattab       VARCHAR2,
   statid        VARCHAR2 DEFAULT NULL,
   statown       VARCHAR2 DEFAULT NULL,
   no_invalidate BOOLEAN DEFAULT NULL,
   force         BOOLEAN DEFAULT FALSE
);
```

Table 2-390 DBE_STATS.IMPORT_INDEX_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	No	Name of the schema. When it is null, the current schema is used by default.
indname	VARCHAR2	No	Name of the index whose statistics are to be imported.
partname	VARCHAR2	Yes	Name of the index partition. If the index has been partitioned but **partname** is **null**, global and partitioned index statistics are imported.
stattab	VARCHAR2	No	Name of the user table that stores statistics.
statid	VARCHAR2	Yes	(Optional) Identifier that points to statistics in **stattab**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Can Be Null	Description
statown	VARCHAR2	Yes	Name of the schema where the user table that stores statistics is located. If this parameter is set to **null**, the value of **ownname** is used.
no_invalidate	BOOLEAN	Yes	Not supported currently.
force	BOOLEAN	Yes	Specifies whether to forcibly import data. The lock status is ignored. The default value is **false**.

NOTE

- You need to import the reltuples (numrows), relpages (numlblks), and relallvisible (relallvisible) statistics of indexes.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
-- Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_import;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_import;
SET
gaussdb=# CREATE TABLE t2(a int, b int, c int)distribute by replication PARTITION BY RANGE(a)
SUBPARTITION BY RANGE(b)
 (
   PARTITION p1 VALUES LESS THAN(200)
   (SUBPARTITION s1 VALUES LESS THAN (500),
    SUBPARTITION s2 VALUES LESS THAN (MAXVALUE)
   ),
   PARTITION p2 VALUES LESS THAN(500)
   (SUBPARTITION s3 VALUES LESS THAN (500),
    SUBPARTITION s4 VALUES LESS THAN (MAXVALUE)
   ),
   PARTITION p3 VALUES LESS THAN(MAXVALUE)
   ( SUBPARTITION s5 VALUES LESS THAN (500),
     SUBPARTITION s6 VALUES LESS THAN (MAXVALUE)
   )
 );
CREATE TABLE
gaussdb=# CREATE INDEX idx_t2_local_a ON t2(a) LOCAL
 (
    PARTITION p1_idx_a
    (
       SUBPARTITION s1_idx_a,
       SUBPARTITION s2_idx_a
    ),
    PARTITION p2_idx_a
    (
       SUBPARTITION s3_idx_a,
       SUBPARTITION s4_idx_a
    ),
    PARTITION p3_idx_a
    (
       SUBPARTITION s5_idx_a,
       SUBPARTITION s6_idx_a
    ));
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
CREATE INDEX
gaussdb=# INSERT INTO t2 VALUES (100, 300, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (100, 600, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (300, 300, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (300, 600, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (600, 400, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (600, 600, 1);
INSERT 0 1
gaussdb=# ANALYZE t2 WITH ALL;
ANALYZE
-- Create a user statistics table and export the data.
gaussdb=# CALL DBE_STATS.CREATE_STAT_TABLE('dbe_stats_import', 'export_table');
 create_stat_table
-------------------
(1 row)
gaussdb=# CALL DBE_STATS.EXPORT_INDEX_STATS(ownname => 'dbe_stats_import', indname =>
'idx_t2_local_a', partname=> null, stattab => 'export_table', statid => 'idx_s1');
 export_index_stats
--------------------
(1 row)
gaussdb=# SELECT relname,partname,numrows,numblocks FROM export_table WHERE nodeid=1
ORDER BY relname,partname desc;
    relname     | partname | numrows | numblocks
----------------+----------+---------+-----------
 idx_t2_local_a |          |       6 |         6
 idx_t2_local_a | s6_idx_a |       0 |         1
 idx_t2_local_a | s5_idx_a |       0 |         1
 idx_t2_local_a | s4_idx_a |       0 |         1
 idx_t2_local_a | s3_idx_a |       0 |         1
 idx_t2_local_a | s2_idx_a |       0 |         1
 idx_t2_local_a | s1_idx_a |       0 |         1
(7 rows)
-- Insert data again, collect statistics, and view the statistics.
gaussdb=# INSERT INTO t2 VALUES (100, 300, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (100, 600, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (100, 600, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (300, 600, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (600, 400, 1);
INSERT 0 1
gaussdb=# INSERT INTO t2 VALUES (600, 600, 1);
INSERT 0 1
gaussdb=# ANALYZE t2 WITH ALL;
ANALYZE
gaussdb=#  SELECT reltuples FROM PG_CLASS WHERE relname='idx_t2_local_a';
 reltuples
-----------
        12
(1 row)
gaussdb=# SELECT relname,relpages,reltuples FROM PG_PARTITION WHERE
parentid='idx_t2_local_a'::REGCLASS ORDER BY oid;
 relname  | relpages | reltuples
----------+----------+-----------
 s1_idx_a |        1 |         0
 s2_idx_a |        1 |         0
 s3_idx_a |        1 |         0
 s4_idx_a |        1 |         0
 s5_idx_a |        1 |         0
 s6_idx_a |        1 |         0
(6 rows)
-- Import the statistics in the user statistics table to the statistics information system catalog.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# CALL DBE_STATS.IMPORT_INDEX_STATS(ownname => 'dbe_stats_import', indname =>
'idx_t2_local_a', stattab => 'export_table', statid => 'idx_s1');
 import_index_stats
--------------------
(1 row)
gaussdb=# SELECT reltuples FROM PG_CLASS WHERE relname='idx_t2_local_a';
 reltuples
-----------
         6
(1 row)
gaussdb=# SELECT relname,relpages,reltuples FROM PG_PARTITION WHERE
parentid='idx_t2_local_a'::REGCLASS ORDER BY oid;
 relname  | relpages | reltuples
----------+----------+-----------
 s1_idx_a |        1 |         0
 s2_idx_a |        1 |         0
 s3_idx_a |        1 |         0
 s4_idx_a |        1 |         0
 s5_idx_a |        1 |         0
 s6_idx_a |        1 |         0
(6 rows)
-- Drop the table and namespace.
gaussdb=# DROP TABLE t2;
DROP TABLE
gaussdb=# DROP TABLE export_table;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_import;
DROP SCHEMA
```

- DBE_STATS.IMPORT_TABLE_STATS

  Retrieves statistics for the specified table from the user statistics table of the **stattab** identifier and writes it back to the system catalog. This API does not return any value.

  The prototype of DBE_STATS.IMPORT_TABLE_STATS is as follows:

```
DBE_STATS.IMPORT_TABLE_STATS (
   ownname        VARCHAR2,
   tabname        VARCHAR2,
   partname       VARCHAR2 DEFAULT NULL,
   stattab        VARCHAR2,
   statid         VARCHAR2 DEFAULT NULL,
   cascade        BOOLEAN  DEFAULT TRUE,
   statown        VARCHAR2 DEFAULT NULL,
   no_invalidate BOOLEAN  DEFAULT NULL,
   force          BOOLEAN  DEFAULT FALSE,
   stat_category VARCHAR2 DEFAULT NULL
);
```

Table 2-391 DBE_STATS.IMPORT_TABLE_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	varchar2	No	Name of the schema. When it is null, the current schema is used by default.
tabname	varchar2	No	Name of the table whose statistics are to be imported.
partname	varchar2	Yes	Table partition name. If the table has been partitioned but **partname** is **null**, global and partitioned table statistics are imported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Can Be Null	Description
stattab	varchar2	No	Name of the user table that stores statistics.
statid	varchar2	Yes	(Optional) Identifier that points to statistics in **stattab**.
cascade	Boolean	Yes	Specifies whether to import statistics about columns, indexes, multiple columns, and index expressions. The default value is **true**.
statown	varchar2	Yes	Name of the schema where the user table that stores statistics is located. If this parameter is set to **null**, the value of **ownname** is used.
no_invalidate	Boolean	Yes	Not supported currently.
force	Boolean	Yes	Specifies whether to forcibly import data. The lock status is ignored. The default value is **false**.
stat_category	varchar2	Yes	Not supported currently.

NOTE

- You need to import the reltuples (numrows), relpages (numblks), and relallvisible (relallvisible) statistics of tables. For details about column-level and index-level statistics imported in cascading mode, see the statistics of the **IMPORT_INDEX_STATS** and **IMPORT_COLUMN_STATS** APIs.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_import_table CASCADE;
NOTICE:  schema "dbe_stats_import_table" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_import_table;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_import_table;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by replication;
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES (1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (2,2);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (3,3);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- Create a user statistics table and export table statistics.
gaussdb=# CALL DBE_STATS.CREATE_STAT_TABLE('dbe_stats_import_table', 'export_table');
 create_stat_table
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------------------

(1 row)

gaussdb=# CALL DBE_STATS.EXPORT_TABLE_STATS(
gaussdb(#     ownname => 'dbe_stats_import_table',
gaussdb(#     tabname => 't1',
gaussdb(#     partname => null,
gaussdb(#     stattab => 'export_table',
gaussdb(#     statid => 't1_t',
gaussdb(#     cascade => true
gaussdb(# );
 export_table_stats
--------------------

(1 row)

gaussdb=# -- Verification: reltuples=3 before the export.
gaussdb=# SELECT relname, reltuples FROM PG_CLASS WHERE oid='t1'::regclass;
 relname | reltuples
---------+-----------
 t1      |         3
(1 row)

gaussdb=# INSERT INTO t1 VALUES (4,4);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (5,5);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (6,6);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# SELECT relname, reltuples FROM PG_CLASS WHERE oid='t1'::regclass;
 relname | reltuples
---------+-----------
 t1      |         6
(1 row)

gaussdb=# CALL DBE_STATS.IMPORT_TABLE_STATS(
gaussdb(#     ownname => 'dbe_stats_import_table',
gaussdb(#     tabname => 't1',
gaussdb(#     partname => null,
gaussdb(#     stattab => 'export_table',
gaussdb(#     statid => 't1_t',
gaussdb(#     cascade => true,
gaussdb(#     force => false
gaussdb(# );
 import_table_stats
--------------------

(1 row)

gaussdb=# -- Verification: reltuples is restored to 3 after the import.
gaussdb=# SELECT relname, reltuples FROM PG_CLASS WHERE oid='t1'::regclass;
 relname | reltuples
---------+-----------
 t1      |         3
(1 row)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP TABLE export_table;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_import_table;
DROP SCHEMA
```

- DBE_STATS.IMPORT_COLUMN_STATS

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Retrieves statistics for the specified columns (multiple columns or expressions) from the user statistics table of the **stattab** identifier and writes it back to the system catalog. This API does not return any value.

The prototype of DBE_STATS.IMPORT_COLUMN_STATS is as follows:

```
DBE_STATS.IMPORT_COLUMN_STATS (
   ownname       VARCHAR2,
   tabname       VARCHAR2,
   colname       VARCHAR2,
   partname      VARCHAR2 DEFAULT NULL,
   stattab       VARCHAR2,
   statid        VARCHAR2 DEFAULT NULL,
   statown       VARCHAR2 DEFAULT NULL,
   no_invalidate BOOLEAN DEFAULT NULL,
   force         BOOLEAN DEFAULT FALSE
);
```

Table 2-392 DBE_STATS.IMPORT_COLUMN_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	No	Name of the schema. When it is null, the current schema is used by default.
tabname	VARCHAR2	No	Name of the table (index name corresponding to the expression) to which the column whose statistics are to be imported belongs.
colname	VARCHAR2	No	Column name or multiple column names (If tabname is an index, **colname** is an expression name.)
partname	VARCHAR2	Yes	Partition name. If the table or index has been partitioned but **partname** is **null**, global and partitioned key, multiple columns, and index statistics are imported.
stattab	VARCHAR2	No	Name of the user table that stores statistics.
statid	VARCHAR2	Yes	(Optional) Identifier that points to statistics in **stattab**.
statown	VARCHAR2	Yes	Name of the schema where the user table that stores statistics is located. If this parameter is set to **null**, the value of **ownname** is used.
no_invalidate	BOOLEAN	Yes	Not supported currently.
force	BOOLEAN	Yes	Specifies whether to forcibly import data. The lock status is ignored. The default value is **false**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- You can query **PG_STATISTIC** or **PG_STATISTIC_EXT** to import column-level statistics.
- Before importing multiple columns, query the **staextname** column in **PG_STATISTIC_EXT** to obtain the aliases of multiple columns and pass them as the input parameter **colname**.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_import_column CASCADE;
NOTICE:  schema "dbe_stats_import_column" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_import_column;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_import_column;
SET
gaussdb=# CREATE TABLE IF NOT EXISTS t1 (a int, b int, c int) distribute by replication;
CREATE TABLE
gaussdb=# CREATE INDEX idx_t1_expr on t1((a*a), (a+a), (a*b));
CREATE INDEX
gaussdb=# INSERT INTO t1 VALUES (100, 300, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (150, 550, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (250, 150, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (250, 150, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (250, 150, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (250, 150, 1);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# ANALYZE t1((a,b));
ANALYZE
gaussdb=# -- Create a user statistics table and export column statistics.
gaussdb=# CALL DBE_STATS.CREATE_STAT_TABLE('dbe_stats_import_column', 'export_table');
 create_stat_table
-------------------

(1 row)

gaussdb=# CALL DBE_STATS.EXPORT_COLUMN_STATS(
gaussdb(#     ownname => 'dbe_stats_import_column',
gaussdb(#     tabname => 't1',
gaussdb(#     colname => 'a',
gaussdb(#     partname => null,
gaussdb(#     stattab => 'export_table',
gaussdb(#     statid => 't1_c'
gaussdb(# );
 export_column_stats
---------------------

(1 row)

gaussdb=# -- Check the export result.
gaussdb=# SELECT relname, columnname, stadistinct, stakind1, stanumbers1, stavalues1
gaussdb-# FROM export_table order by 1,2,3,4;
 relname | columnname | stadistinct | stakind1 | stanumbers1 | stavalues1
---------+------------+-------------+----------+-------------+------------
 t1      | a          |        -.5 |        1 | {.666667}   | {250}
 t1      | a          |        -.5 |        1 | {.666667}   | {250}
 t1      | a          |        -.5 |        1 | {.666667}   | {250}
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(3 rows)

gaussdb=# -- Insert data again and collect statistics.
gaussdb=# INSERT INTO t1 VALUES (100, 300, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (150, 550, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (250, 150, 1);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- View the column statistics before import (stadistinct changes).
gaussdb=# SELECT staattnum, stadistinct, stakind1, stanumbers1, stavalues1
gaussdb-# FROM PG_STATISTIC WHERE starelid='t1'::REGCLASS ORDER BY 1;
 staattnum | stadistinct | stakind1 |        stanumbers1         |  stavalues1
-----------+-------------+----------+----------------------------+---------------
         1 |    -.333333 |        1 | {.555556,.222222,.222222} | {250,100,150}
         2 |    -.333333 |        1 | {.555556,.222222,.222222} | {150,300,550}
         3 |    -.111111 |        1 | {1}                       | {1}
(3 rows)

gaussdb=# CALL DBE_STATS.IMPORT_COLUMN_STATS(
gaussdb(#     ownname => 'dbe_stats_import_column',
gaussdb(#     tabname => 't1',
gaussdb(#     colname => 'a',
gaussdb(#     partname => null,
gaussdb(#     stattab => 'export_table',
gaussdb(#     statid => 't1_c',
gaussdb(#     force => false
gaussdb(# );
 import_column_stats
---------------------

(1 row)

gaussdb=# -- Verification: Statistics about column a are restored after the import.
gaussdb=# SELECT staattnum, stadistinct, stakind1, stanumbers1, stavalues1
gaussdb-# FROM PG_STATISTIC WHERE starelid='t1'::REGCLASS ORDER BY 1;
 staattnum | stadistinct | stakind1 |        stanumbers1         |  stavalues1
-----------+-------------+----------+----------------------------+---------------
         1 |          -.5 |        1 | {.666667}                 | {250}
         2 |    -.333333 |        1 | {.555556,.222222,.222222} | {150,300,550}
         3 |    -.111111 |        1 | {1}                       | {1}
(3 rows)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP TABLE export_table;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_import_column;
DROP SCHEMA
```

- DBE_STATS.IMPORT_SCHEMA_STATS

  Retrieves statistics for the specified schema from the user statistics table of the **stattab** identifier and writes it back to the system catalog. This API does not return any value.

  The prototype of DBE_STATS.IMPORT_SCHEMA_STATS is as follows:

```
DBE_STATS.IMPORT_SCHEMA_STATS (
   ownname       VARCHAR2,
   stattab       VARCHAR2,
   statid        VARCHAR2 DEFAULT NULL,
   statown       VARCHAR2 DEFAULT NULL,
   no_invalidate BOOLEAN  DEFAULT NULL,
   force         BOOLEAN DEFAULT FALSE,
   stat_category VARCHAR2 DEFAULT NULL
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-393 DBE_STATS.IMPORT_SCHEMA_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	No	Name of the schema. When it is null, the current schema is used by default.
stattab	VARCHAR2	No	Name of the user table that stores statistics.
statid	VARCHAR2	Yes	(Optional) Identifier that points to statistics in **stattab**.
statown	VARCHAR2	Yes	Name of the schema where the user table that stores statistics is located. If this parameter is set to **null**, the value of **ownname** is used.
no_invalidate	BOOLEAN	Yes	Not supported currently.
force	BOOLEAN	Yes	Specifies whether to forcibly import data. The lock status is ignored. The default value is **false**.
stat_category	VARCHAR2	Yes	Not supported currently.

NOTE

- You need to import statistics about all tables on which the current user has permission in the specified schema. Tables on which the current user does not have permission are skipped and no error is reported.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_import_schema CASCADE;
NOTICE:  schema "dbe_stats_import_schema" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_import_schema;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_import_schema;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by replication;
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES (1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (2,2);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (3,3);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- Create a user statistics table and export schema statistics.
gaussdb=# CALL DBE_STATS.CREATE_STAT_TABLE('dbe_stats_import_schema', 'export_table');
 create_stat_table
-------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)

gaussdb=# CALL DBE_STATS.EXPORT_SCHEMA_STATS(
gaussdb(#     ownname => 'dbe_stats_import_schema',
gaussdb(#     stattab => 'export_table',
gaussdb(#     statid => 's1'
gaussdb(# );
 export_schema_stats
---------------------

(1 row)

gaussdb=# -- Verification: reltuples=3 before the export.
gaussdb=# SELECT relname, reltuples FROM PG_CLASS WHERE oid='t1'::regclass;
 relname | reltuples
---------+-----------
 t1      |         3
(1 row)

gaussdb=# -- Insert data again and collect statistics.
gaussdb=# INSERT INTO t1 VALUES (4,4);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (5,5);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (6,6);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- Verification: Statistics have been updated (reltuples=6).
gaussdb=# SELECT relname, reltuples FROM PG_CLASS WHERE oid='t1'::regclass;
 relname | reltuples
---------+-----------
 t1      |         6
(1 row)

gaussdb=# -- Import the schema statistics.
gaussdb=# CALL DBE_STATS.IMPORT_SCHEMA_STATS(
gaussdb(#     ownname => 'dbe_stats_import_schema',
gaussdb(#     stattab => 'export_table',
gaussdb(#     statid => 's1',
gaussdb(#     force => false
gaussdb(# );
 import_schema_stats
---------------------

(1 row)

gaussdb=# -- Verification: reltuples is restored to 3 after the import.
gaussdb=# SELECT relname, reltuples FROM PG_CLASS WHERE oid='t1'::regclass;
 relname | reltuples
---------+-----------
 t1      |         3
(1 row)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP TABLE export_table;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_import_schema;
DROP SCHEMA
```

- DBE_STATS.IMPORT_DATABASE_STATS

  Retrieves statistics for user tables in the current database from the user statistics table of the **stattab** identifier and writes it back to the system catalog. This API does not return any value.

  The prototype of DBE_STATS.IMPORT_DATABASE_STATS is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_STATS.IMPORT_DATABASE_STATS (
   stattab         VARCHAR2,
   statid          VARCHAR2 DEFAULT NULL,
   statown         VARCHAR2 DEFAULT NULL,
   no_invalidate   BOOLEAN DEFAULT NULL,
   force           BOOLEAN DEFAULT FALSE,
   stat_category   VARCHAR2 DEFAULT NULL
);
```

Table 2-394 DBE_STATS.IMPORT_DATABASE_STATS API parameters

Parameter	Type	Can Be Null	Description
stattab	VARCHAR2	No	Name of the user table that stores statistics.
statid	VARCHAR2	Yes	(Optional) Identifier that points to statistics in **stattab**.
statown	VARCHAR2	Yes	Name of the schema where the user table that stores statistics is located. If this parameter is set to **null**, the current schema is used.
no_invalidate	BOOLEAN	Yes	Not supported currently.
force	BOOLEAN	Yes	Specifies whether to forcibly import data. The lock status is ignored. The default value is **false**.
stat_category	VARCHAR2	Yes	Not supported currently.

NOTE

- You can import statistics about all tables on which the current user has the ANALYZE permission in the specified database. Tables on which the current user does not have permission are skipped and no error is reported.
- If **statown** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_import_database CASCADE;
NOTICE:  schema "dbe_stats_import_database" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_import_database;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_import_database;
SET
gaussdb=# CREATE TABLE t1(a int, b int) distribute by replication;
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES (1,1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (2,2);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (3,3);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- Create a user statistics table and export database statistics.
gaussdb=# CALL DBE_STATS.CREATE_STAT_TABLE('dbe_stats_import_database', 'export_table');
 create_stat_table
-------------------

(1 row)

gaussdb=# CALL DBE_STATS.EXPORT_DATABASE_STATS(
gaussdb(#     statown => 'dbe_stats_import_database',
gaussdb(#     stattab => 'export_table',
gaussdb(#     statid => 's1'
gaussdb(# );
 export_database_stats
-----------------------

(1 row)

gaussdb=# -- Verification: reltuples=3 before the export.
gaussdb=# SELECT relname, reltuples FROM PG_CLASS WHERE oid='t1'::regclass;
 relname | reltuples
---------+-----------
 t1      |         3
(1 row)

gaussdb=# -- Insert data again and collect statistics.
gaussdb=# INSERT INTO t1 VALUES (4,4);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (5,5);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (6,6);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- Verification: Statistics have been updated (reltuples=6).
gaussdb=# SELECT relname, reltuples FROM PG_CLASS WHERE oid='t1'::regclass;
 relname | reltuples
---------+-----------
 t1      |         6
(1 row)

gaussdb=# -- Import the database statistics.
gaussdb=# CALL DBE_STATS.IMPORT_DATABASE_STATS(
gaussdb(#     statown => 'dbe_stats_import_database',
gaussdb(#     stattab => 'export_table',
gaussdb(#     statid => 's1',
gaussdb(#     force => false
gaussdb(# );
 import_database_stats
-----------------------

(1 row)

gaussdb=# -- Verification: reltuples is restored to 3 after the import.
gaussdb=# SELECT relname, reltuples FROM PG_CLASS WHERE oid='t1'::regclass;
 relname | reltuples
---------+-----------
 t1      |         3
(1 row)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP TABLE export_table;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_import_database;
DROP SCHEMA
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_STATS.SET_COLUMN_STATS

  Sets column-related statistics, including single-column, multi-column, and expression statistics. This API does not return any value.

  The prototype of DBE_STATS.SET_COLUMN_STATS is as follows:

```
DBE_STATS.SET_COLUMN_STATS (
   ownname         VARCHAR2,
   tabname         VARCHAR2,
   colname         VARCHAR2,
   partname        VARCHAR2   DEFAULT NULL,
   stattab         VARCHAR2   DEFAULT NULL,
   statid          VARCHAR2   DEFAULT NULL,
   distcnt         NUMBER     DEFAULT NULL,
   density         NUMBER     DEFAULT NULL,
   nullcnt         NUMBER     DEFAULT NULL,
   srec            VARCHAR2   DEFAULT NULL,
   avgclen         NUMBER     DEFAULT NULL,
   flags           NUMBER     DEFAULT NULL,
   statown         VARCHAR2   DEFAULT NULL,
   no_invalidate   BOOLEAN    DEFAULT NULL,
   force           BOOLEAN    DEFAULT FALSE,
   node_name       VARCHAR2   DEFAULT NULL,
   option          VARCHAR2   DEFAULT NULL
);
```

Table 2-395 DBE_STATS.SET_COLUMN_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	No	Name of the schema where the table is located. When it is null, the current schema is used by default.
tabname	VARCHAR2	No	Table name.
colname	VARCHAR2	No	Column name.
partname	VARCHAR2	Yes	Name of the table partition whose statistics are to be set. If the table has been partitioned and **partname** is null, global column–level statistics are set.
stattab	VARCHAR2	Yes	Not supported currently.
statid	VARCHAR2	Yes	Not supported currently.
distcnt	NUMBER	Yes	Number of NDVs in the column.
density	NUMBER	Yes	Not supported currently.
nullcnt	NUMBER	Yes	Number of null values in the column.
srec	VARCHAR2	Yes	Not supported currently.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Can Be Null	Description
avgclen	NUMBER	Yes	Average length (in bytes) of the fields in the column.
flags	NUMBER	Yes	Not supported currently.
statown	VARCHAR2	Yes	Not supported currently.
no_invalidate	BOOLEAN	Yes	Not supported currently.
force	BOOLEAN	Yes	Behavior when the statistics are locked. If the value is **true**, the procedure sets the value even if the column-level statistics are locked. The default value is **false**.
node_name	VARCHAR2	Yes	If **node_name** is **null**, the statistics of all CNs and DNs are updated. If **node_name** is the DN name, only the statistics of the specified DN are updated. If **node_name** is the name of a CN, statistics of all CNs will be updated.
option	VARCHAR2	Yes	NDV calculation method of the DN. This parameter is valid only when **node_name** is **null**. Currently, only **"p"** and **"s"** are allowed. **"p"**: The NDV of a DN is calculated using Poisson distribution. **"s"**: The DN and CN use the same NDV.

NOTE

- This API can be used to set expression statistics. Expression creation depends on indexes. Therefore, when this API is called to set expression statistics, **tabname** must be set to the index name.
- Currently, only NDV (distcnt), nullfrac (nullcnt) and width (avgclen) statistics of columns can be set.
- After the statistics of a single column or expression are deleted, this API can still be called to set the statistics of the corresponding single column or expression. In this case, new data is inserted into the **PG_STATISTIC** system catalog. After the multi-column statistics are deleted, this API cannot be used to reset the statistics.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.
- If the number of rows (**reltuples**) of the current column is **0**, **nullfrac** will be set to **0** and **distcnt** will be set to the input parameter value.

Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Prerequisites: Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_set_schema;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_set_schema;
SET
gaussdb=# DROP TABLE IF EXISTS t1;
NOTICE:  table "t1" does not exist, skipping
DROP TABLE
gaussdb=# CREATE TABLE IF NOT EXISTS t1 (a int, b int, c int) distribute by replication;
CREATE TABLE
gaussdb=# CREATE INDEX idx_t1_expr ON t1((a*a), (a+a), (a*b));
CREATE INDEX
gaussdb=# INSERT INTO t1 VALUES (100, 300, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (150, 550, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (250, 150, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (250, 150, 1);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# ANALYZE t1((a, b));
ANALYZE
-- Set single-column statistics.
-- Query the original statistics.
gaussdb=# SELECT staattnum,stanullfrac,stawidth,stadistinct FROM PG_STATISTIC WHERE
starelid='t1'::REGCLASS ORDER BY 1;
 staattnum | stanullfrac | stawidth | stadistinct
-----------+-------------+----------+-------------
         1 |           0 |        4 |        -.75
         2 |           0 |        4 |        -.75
         3 |           0 |        4 |        -.25
(3 rows)
-- Set the column-level statistics and query again.
gaussdb=# CALL DBE_STATS.SET_COLUMN_STATS(ownname => 'dbe_stats_set_schema', tabname =>
't1', colname => 'a', distcnt => 6, nullcnt => 5, avgclen => 42);
 set_column_stats
------------------
(1 row)
gaussdb=# SELECT staattnum,stanullfrac,stawidth,stadistinct FROM PG_STATISTIC WHERE
starelid='t1'::REGCLASS ORDER BY 1;
 staattnum | stanullfrac | stawidth | stadistinct
-----------+-------------+----------+-------------
         1 |           1 |       42 |           6
         2 |           0 |        4 |        -.75
         3 |           0 |        4 |        -.25
(3 rows)
-- Set multi-column statistics.
-- Query the original statistics.
gaussdb=# SELECT stakey,staextname,stanullfrac,stawidth,stadistinct FROM PG_STATISTIC_EXT
WHERE starelid='t1'::REGCLASS;
 stakey |     staextname      | stanullfrac | stawidth | stadistinct
--------+---------------------+-------------+----------+-------------
 1 2    | extname_-1308656218 |           0 |        8 |        -.75
(1 row)
-- Set the statistics and query again.
gaussdb=# CALL DBE_STATS.SET_COLUMN_STATS(ownname => 'dbe_stats_set_schema', tabname =>
't1', colname => 'extname_-1308656218', distcnt => 5, nullcnt => 5, avgclen => 2);
 set_column_stats
------------------
(1 row)
gaussdb=# SELECT stakey,staextname,stanullfrac,stawidth,stadistinct FROM PG_STATISTIC_EXT
WHERE starelid='t1'::REGCLASS;
 stakey |     staextname      | stanullfrac | stawidth | stadistinct
--------+---------------------+-------------+----------+-------------
 1 2    | extname_-1308656218 |           1 |        2 |           5
(1 row)
-- Set expression statistics.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Query the original statistics.
gaussdb=# SELECT staattnum,stanullfrac,stawidth,stadistinct FROM PG_STATISTIC WHERE
starelid='idx_t1_expr'::REGCLASS ORDER BY 1;
 staattnum | stanullfrac | stawidth | stadistinct
-----------+-------------+----------+-------------
         1 |           0 |        4 |       -.75
         2 |           0 |        4 |       -.75
         3 |           0 |        4 |       -.75
(3 rows)
-- Query an expression column name.
gaussdb=# SELECT attnum, attname FROM PG_ATTRIBUTE WHERE attrelid='idx_t1_expr'::REGCLASS
ORDER BY 1;
 attnum | attname
--------+---------
      1 | expr
      2 | expr1
      3 | expr2
(3 rows)
-- Set the statistics and query again.
gaussdb=# CALL DBE_STATS.SET_COLUMN_STATS(ownname => 'dbe_stats_set_schema', tabname =>
'idx_t1_expr', colname => 'expr', distcnt => 13, nullcnt => 5, avgclen => 22);
 set_column_stats
------------------

(1 row)
gaussdb=# SELECT staattnum,stanullfrac,stawidth,stadistinct FROM PG_STATISTIC WHERE
starelid='idx_t1_expr'::REGCLASS ORDER BY 1;
 staattnum | stanullfrac | stawidth | stadistinct
-----------+-------------+----------+-------------
         1 |           1 |       22 |          13
         2 |           0 |        4 |       -.75
         3 |           0 |        4 |       -.75
(3 rows)
-- Set the statistics of a specified DN.
gaussdb=# CALL DBE_STATS.SET_COLUMN_STATS(ownname => 'dbe_stats_set_schema', tabname =>
't1', colname => 'a', distcnt => 3, nullcnt => 3,avgclen => 3, node_name=>'dn_6007_6008_6009');
 set_column_stats
------------------

(1 row)
gaussdb=# EXECUTE DIRECT ON (dn_6007_6008_6009) $$SELECT
staattnum,stanullfrac,stawidth,stadistinct FROM PG_STATISTIC WHERE starelid='t1'::REGCLASS AND
staattnum = 1;$$;
staattnum | stanullfrac | stawidth | stadistinct
-----------+-------------+----------+-------------
         1 |         .75 |        3 |           3
(1 row)
-- Set the statistics of all CNs.
gaussdb=# CALL DBE_STATS.SET_COLUMN_STATS(ownname => 'dbe_stats_set_schema', tabname =>
't1', colname => 'a', distcnt => 6, nullcnt => 6,avgclen => 6, node_name=>'cn_5002');
 set_column_stats
------------------

(1 row)
gaussdb=# EXECUTE DIRECT ON (cn_5002) $$SELECT staattnum,stanullfrac,stawidth,stadistinct FROM
PG_STATISTIC WHERE starelid='t1'::REGCLASS AND staattnum = 1;$$;
 staattnum | stanullfrac | stawidth | stadistinct
-----------+-------------+----------+-------------
         1 |           1 |        6 |           6
(1 row)
-- Set the statistics of the entire cluster. The value of distcnt of the DN is the same as that of the CN.
gaussdb=# CALL DBE_STATS.SET_COLUMN_STATS(ownname => 'dbe_stats_set_schema', tabname =>
't1', colname => 'a', distcnt => 100, nullcnt => 100,avgclen => 100, option=>'s');
 set_column_stats
------------------

(1 row)
gaussdb=# EXECUTE DIRECT ON (cn_5002) $$SELECT staattnum,stanullfrac,stawidth,stadistinct FROM
PG_STATISTIC WHERE starelid='t1'::REGCLASS AND staattnum = 1;$$;
 staattnum | stanullfrac | stawidth | stadistinct
```

```
-----------+-------------+----------+-------------
         1 |           1 |      100 |         100
(1 row)
gaussdb=# EXECUTE DIRECT ON (dn_6007_6008_6009) $$SELECT
staattnum,stanullfrac,stawidth,stadistinct FROM PG_STATISTIC WHERE starelid='t1'::REGCLASS AND
staattnum = 1;$$;
 staattnum | stanullfrac | stawidth | stadistinct
-----------+-------------+----------+-------------
         1 |           1 |      100 |         100
(1 row)
-- Set the number of rows and pages of a specified DN.
gaussdb=# CALL DBE_STATS.SET_TABLE_STATS(tabname => 't1', ownname => 'dbe_stats_set_schema',
numrows => 3, numblks => 3, relallvisible => 1, node_name=>'dn_6007_6008_6009');
 set_table_stats
-----------------

(1 row)
-- Set the statistics of the entire cluster. The distcnt value of the DN is calculated using the Poisson
distribution.
gaussdb=# CALL DBE_STATS.SET_COLUMN_STATS(ownname => 'dbe_stats_set_schema', tabname =>
't1', colname => 'a', distcnt => 100, nullcnt => 100,avgclen => 100, option=>'p');
 set_column_stats
------------------


(1 row)
-- Drop the table and namespace.
gaussdb=# DROP INDEX idx_t1_expr;
DROP INDEX
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_set_schema cascade;
DROP SCHEMA
```

- DBE_STATS.SET_INDEX_STATS

  Sets index-related statistics. This API does not return any value.

  The prototype of DBE_STATS.SET_INDEX_STATS is as follows:

```
DBE_STATS.SET_INDEX_STATS (
   ownname       VARCHAR2,
   indname       VARCHAR2,
   partname      VARCHAR2  DEFAULT NULL,
   stattab       VARCHAR2  DEFAULT NULL,
   statid        VARCHAR2  DEFAULT NULL,
   numrows       NUMBER    DEFAULT NULL,
   numlblks      NUMBER    DEFAULT NULL,
   relallvisible NUMBER    DEFAULT NULL,
   numdist       NUMBER    DEFAULT NULL,
   avglblk       NUMBER    DEFAULT NULL,
   avgdblk       NUMBER    DEFAULT NULL,
   clstfct       NUMBER    DEFAULT NULL,
   indlevel      NUMBER    DEFAULT NULL,
   flags         NUMBER    DEFAULT NULL,
   statown       VARCHAR2  DEFAULT NULL,
   no_invalidate BOOLEAN   DEFAULT NULL,
   guessq        NUMBER    DEFAULT NULL,
   cachedblk     NUMBER    DEFAULT NULL,
   cachehit      NUMBER    DEFAULT NULL,
   force         BOOLEAN   DEFAULT FALSE,
   node_name     VARCHAR2  DEFAULT NULL
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-396 DBE_STATS.SET_INDEX_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	No	Schema to which the index belongs. When it is null, the current schema is used by default.
indname	VARCHAR2	No	Index name.
partname	VARCHAR2	Yes	Name of the index partition whose statistics are to be set. If the index has been partitioned and **partname** is null, global index–level statistics are set.
stattab	VARCHAR2	Yes	Not supported currently.
statid	VARCHAR2	Yes	Not supported currently.
numrows	NUMBER	Yes	Number of rows in an index.
numlblks	NUMBER	Yes	Number of pages in an index.
relallvisible	NUMBER	Yes	Number of pages marked as all visible in the index.
numdist	NUMBER	Yes	Not supported currently.
avglblk	NUMBER	Yes	Not supported currently.
avgdblk	NUMBER	Yes	Not supported currently.
clstfct	NUMBER	Yes	Not supported currently.
indlevel	NUMBER	Yes	Not supported currently.
flags	NUMBER	Yes	Not supported currently.
statown	VARCHAR2	Yes	Not supported currently.
no_invalidate	BOOLEAN	Yes	Not supported currently.
guessq	NUMBER	Yes	Not supported currently.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Can Be Null	Description
cachedblk	NUMBER	Yes	Not supported currently.
cachehit	NUMBER	Yes	Not supported currently.
force	BOOLEAN	Yes	Behavior when the statistics are locked. If the value is **TRUE**, the procedure sets the value even if the index-level statistics are locked. The default value is **false**.
node_name	VARCHAR2	Yes	If **node_name** is **null**, the statistics of all CNs and DNs are updated. If **node_name** is the DN name, only the statistics of the specified DN are updated. If **node_name** is the name of a CN, statistics of all CNs will be updated.

NOTE

- Currently, only reltuples (numrows), relpages (numlblks) and relallvisible (relallvisible) statistics of indexes can be set.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
-- Prerequisites: Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_set_schema;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_set_schema;
SET
gaussdb=# DROP TABLE IF EXISTS t1;
NOTICE:  table "t1" does not exist, skipping
DROP TABLE
gaussdb=# CREATE TABLE IF NOT EXISTS t1 (a int, b int, c int) distribute by replication;
CREATE TABLE
gaussdb=# CREATE INDEX idx_t1_a on t1(a);
CREATE INDEX
gaussdb=# INSERT INTO t1 VALUES (100, 300, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (150, 550, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (250, 150, 1);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
-- Set index statistics.
-- View the current index statistics.
gaussdb=# SELECT reltuples FROM PG_CLASS WHERE relname='idx_t1_a';
 reltuples
-----------
         3
(1 row)
-- Set the statistics and query again.
gaussdb=# CALL DBE_STATS.SET_INDEX_STATS(indname => 'idx_t1_a', ownname =>
'dbe_stats_set_schema', numrows => 1, numlblks => 3, relallvisible => 6);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 set_index_stats
-----------------
(1 row)
gaussdb=# SELECT reltuples,relpages,relallvisible FROM PG_CLASS WHERE relname='idx_t1_a';
 reltuples | relpages | relallvisible
-----------+----------+---------------
         1 |        3 |             6
(1 row)
-- Set the statistics of a specified DN.
gaussdb=# CALL DBE_STATS.SET_INDEX_STATS(indname => 'idx_t1_a', ownname =>
'dbe_stats_set_schema', numrows => 2, numlblks => 2, relallvisible =>
2,node_name=>'dn_6007_6008_6009');
 set_index_stats
-----------------

(1 row)
-- Query the statistics of a specified DN.
gaussdb=# EXECUTE DIRECT ON (dn_6007_6008_6009) $$SELECT reltuples,relpages,relallvisible FROM
PG_CLASS WHERE relname='idx_t1_a'$$;
 reltuples | relpages | relallvisible
-----------+----------+---------------
         2 |        2 |             2
(1 row)
-- Set the statistics of a specified CN.
gaussdb=# CALL DBE_STATS.SET_INDEX_STATS(indname => 'idx_t1_a', ownname =>
'dbe_stats_set_schema', numrows => 10, numlblks => 100, relallvisible => 10,node_name=>'cn_5001');
 set_index_stats
-----------------

(1 row)
-- Query the statistics of a specified CN.
gaussdb=# EXECUTE DIRECT ON (cn_5001) $$SELECT reltuples,relpages,relallvisible FROM PG_CLASS
WHERE relname='idx_t1_a'$$;
 reltuples | relpages | relallvisible
-----------+----------+---------------
        10 |      100 |            10
(1 row)
-- Drop the table and namespace.
gaussdb=# DROP INDEX idx_t1_a;
DROP INDEX
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_set_schema CASCADE;
DROP SCHEMA
```

- DBE_STATS.SET_TABLE_STATS

  Sets table-level statistics. This API does not return any value.

  The prototype of DBE_STATS.SET_TABLE_STATS is as follows:

```
DBE_STATS.SET_TABLE_STATS (
    ownname         VARCHAR2,
    tabname         VARCHAR2,
    partname        VARCHAR2 DEFAULT NULL,
    stattab         VARCHAR2  DEFAULT NULL,
    statid          VARCHAR2  DEFAULT NULL,
    numrows         NUMBER    DEFAULT NULL,
    numblks         NUMBER    DEFAULT NULL,
    relallvisible   NUMBER    DEFAULT NULL,
    avgrlen         NUMBER    DEFAULT NULL,
    flags           NUMBER    DEFAULT NULL,
    statown         VARCHAR2  DEFAULT NULL,
    no_invalidate   BOOLEAN   DEFAULT NULL,
    cachedblk       NUMBER    DEFAULT NULL,
    cachehit        NUMBER    DEFAULT NULL,
    force           BOOLEAN   DEFAULT FALSE,
    im_imcu_count   NUMBER    DEFAULT NULL,
    im_block_count NUMBER     DEFAULT NULL,
    scanrate        NUMBER    DEFAULT NULL,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  node_name     VARCHAR2  DEFAULT NULL
);
```

Table 2-397 DBE_STATS.SET_TABLE_STATS API parameters

Parameter	Type	Can Be Null	Description
ownname	VARCHAR2	No	Name of the schema where the table is located. When it is null, the current schema is used by default.
tabname	VARCHAR2	No	Table name.
partname	VARCHAR2	Yes	Name of the table partition whose statistics are to be set. If the table has been partitioned and **partname** is null, global table–level statistics are set.
stattab	VARCHAR2	Yes	Not supported currently.
statid	VARCHAR2	Yes	Not supported currently.
numrows	NUMBER	Yes	Number of rows in a table.
numblks	NUMBER	Yes	Number of pages in a table.
relallvisible	NUMBER	Yes	Number of pages marked as all visible in a table.
avgrlen	NUMBER	Yes	Not supported currently.
flags	NUMBER	Yes	Not supported currently.
statown	VARCHAR2	Yes	Not supported currently.
no_invalidate	BOOLEAN	Yes	Not supported currently.
cachedblk	NUMBER	Yes	Not supported currently.
cachehit	NUMBER	Yes	Not supported currently.
force	BOOLEAN	Yes	Behavior when the statistics are locked. If the value is **TRUE**, the procedure sets the value even if the table statistics are locked. The default value is **false**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Can Be Null	Description
im_imcu_count	NUMBER	Yes	Not supported currently.
im_block_count	NUMBER	Yes	Not supported currently.
scanrate	NUMBER	Yes	Not supported currently.
node_name	VARCHAR2	Yes	If **node_name** is **null**, the statistics of all CNs and DNs are updated. If **node_name** is the DN name, only the statistics of the specified DN are updated. If **node_name** is the name of a CN, statistics of all CNs will be updated.

NOTE

- Currently, only reltuples (numrows), relpages (numblks) and relallvisible (relallvisible) statistics of tables can be set.
- If **ownname** is set to **null**, the current schema is used. In this case, you need to set **behavior_compat_options** to **bind_procedure_searchpath**.

Example:

```
gaussdb=# DROP SCHEMA IF EXISTS dbe_stats_set_schema_table CASCADE;
NOTICE:  schema "dbe_stats_set_schema_table" does not exist, skipping
DROP SCHEMA
gaussdb=# -- Prerequisite: Create a table and collect statistics.
gaussdb=# CREATE SCHEMA dbe_stats_set_schema_table;
CREATE SCHEMA
gaussdb=# SET CURRENT_SCHEMA = dbe_stats_set_schema_table;
SET
gaussdb=# DROP TABLE IF EXISTS t1;
NOTICE:  table "t1" does not exist, skipping
DROP TABLE
gaussdb=# CREATE TABLE IF NOT EXISTS t1 (a int, b int, c int) distribute by replication;
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES (100, 300, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (150, 550, 1);
INSERT 0 1
gaussdb=# INSERT INTO t1 VALUES (250, 150, 1);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# -- View statistics about the current table.
gaussdb=# SELECT reltuples, relpages, relallvisible FROM PG_CLASS WHERE oid='t1'::regclass;
 reltuples | relpages | relallvisible
-----------+----------+---------------
         3 |        1 |             0
(1 row)

gaussdb=# -- Set table-level statistics (numrows=3, numblks=3, relallvisible=3).
gaussdb=# CALL DBE_STATS.SET_TABLE_STATS(
gaussdb(#     tabname => 't1',
gaussdb(#     ownname => 'dbe_stats_set_schema_table',
gaussdb(#     numrows => 3,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb(#     numblks => 3,
gaussdb(#     relallvisible => 3
gaussdb(# );
 set_table_stats
-----------------

(1 row)

gaussdb=# -- Verification: The table statistics have been set (reltuples=3, relpages=3, relallvisible=3).
gaussdb=# SELECT reltuples, relpages, relallvisible FROM PG_CLASS WHERE oid='t1'::regclass;
 reltuples | relpages | relallvisible
-----------+----------+---------------
         3 |        3 |             3
(1 row)

gaussdb=# -- Drop.
gaussdb=# DROP TABLE t1;
DROP TABLE
gaussdb=# DROP SCHEMA dbe_stats_set_schema_table CASCADE;
DROP SCHEMA
```

- DBE_STATS.GET_TABLES_BY_SQLID

  Obtains all tables involved in the statement corresponding to the unique SQL ID. If multiple tables are involved, multiple rows of records are returned.

  The prototype of DBE_STATS.GET_TABLES_BY_SQLID is as follows:

```
DBE_STATS.GET_TABLES_BY_SQLID (
   unique_sqlid    BIGINT
)
RETURNS TABLE (
   nspname         TEXT,
   tabname         TEXT
);
```

Table 2-398 Parameters and return values of the DBE_STATS.GET_TABLES_BY_SQLID API

Column	Data Type	Parameter Type	Can Be Null	Description
unique_sqlid	BIGINT	IN	No	Unique SQL ID, which can be obtained from pg_catalog.pg_stat_activity (view for recording real-time session information), dbe_perf.statement (view for recording normalized SQL statements), or dbe_perf.statement_history (view for recording full or slow SQL statements).
nspname	TEXT	OUT	No	Schema name of the table involved in the statement.
tabname	TEXT	OUT	No	Name of the table involved in the statement.

Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Prerequisites
gaussdb=# DROP TABLE IF EXISTS t1;
DROP TABLE
gaussdb=# CREATE TABLE t1 (c1 int) DISTRIBUTE BY HASH (c1);
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES (1);
INSERT 0 1
gaussdb=# CREATE OR REPLACE PROCEDURE select_t1()
gaussdb-# AS
gaussdb$# BEGIN
gaussdb$#     PERFORM * FROM t1;
gaussdb$# END;
gaussdb$# /
CREATE PROCEDURE
gaussdb=# DROP TABLE IF EXISTS t2;
DROP TABLE
gaussdb=# CREATE TABLE t2 (c1 int) DISTRIBUTE BY HASH (c1) PARTITION BY LIST (c1) (
gaussdb(#     PARTITION t2_p1 VALUES (1),
gaussdb(#     PARTITION t2_p2 VALUES (2)
gaussdb(# );
CREATE TABLE
gaussdb=# INSERT INTO t2 VALUES (1), (2);
INSERT 0 2

-- Use the unique SQL ID queried from pgxc_stat_activity.
-- Session 1: Simulate a statement that cannot be executed.
gaussdb=# SET cursor_sharing = off; -- This avoids automatic parameterization without parsing.
SET
gaussdb=# SELECT pg_sleep(60) from t1;
 pg_sleep
----------

(1 row)

-- Session 2: Query the unique SQL ID and the corresponding table name during the execution of
session 1. The statement needs to be adjusted based on the actual SQL ID.
gaussdb=#  SELECT query, sessionid, unique_sql_id FROM pgxc_stat_activity WHERE query LIKE
'%pg_sleep%t1%' AND query NOT LIKE '%pgxc_stat_activity%';
            query              | sessionid | unique_sql_id
-------------------------------+-----------+---------------
 SELECT pg_sleep(60) FROM t1; |         1 |    3101300097
(1 row)

gaussdb=# SELECT * FROM dbe_stats.get_tables_by_sqlid(3101300097);
 nspname | tabname
---------+---------
 public  | t1
(1 row)

-- 2. Use the unique SQL ID queried from dbe_perf.statement.
gaussdb=# SHOW instr_unique_sql_count; -- View the maximum number of records in the statement
view.
 instr_unique_sql_count
------------------------
 200000
(1 row)

gaussdb=# SELECT count(*) FROM dbe_perf.summary_statement; -- Check whether the number of
records in the statement view exceeds the upper limit.
 count
-------
    46
(1 row)

gaussdb=# SELECT reset_unique_sql('GLOBAL','ALL',0); -- If the number of records reaches the upper
limit, subsequent statements will not be recorded. You can clear the records. (Note: This will cause
SQL records to be lost.)
 reset_unique_sql
------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 t
(1 row)

gaussdb=# SELECT * FROM t1;
 c1
----
  1
(1 row)

gaussdb=# SELECT unique_sql_id FROM dbe_perf.summary_statement WHERE query LIKE '%SELECT *
FROM t1%';
 unique_sql_id
---------------
    4115569441
(1 row)

gaussdb=# SELECT * FROM dbe_stats.get_tables_by_sqlid(4115569441);
 nspname | tabname
---------+---------
 public  | t1
(1 row)

-- 3. Use the unique SQL ID queried from dbe_stats.statement_history.
gaussdb=# SET log_min_duration_statement = '1s'; -- Slow SQL statements whose execution time is
greater than or equal to 1s will be recorded.
SET
gaussdb=# DELETE FROM pg_catalog.statement_history; -- Avoid being affected by residual slow SQL
records.
DELETE 2
gaussdb=# SELECT pg_sleep(1) FROM t1;
 pg_sleep
----------

(1 row)

gaussdb=# SELECT unique_query_id FROM dbe_perf.statement_history WHERE query LIKE '%pg_sleep
%t1%' LIMIT 1;
 unique_query_id
-----------------
      3101300097
(1 row)

gaussdb=# SELECT * FROM dbe_stats.get_tables_by_sqlid(3101300097);
 nspname | tabname
---------+---------
 public  | t1
(1 row)

-- 4. Use the unique SQL ID of the stored procedure queried from the dbe_stats.statement view.
gaussdb=# SHOW instr_unique_sql_track_type; -- Check whether the statement view records include
the sub-statements of the stored procedure.
 instr_unique_sql_track_type
-----------------------------
 all
(1 row)

gaussdb=# SELECT * FROM select_t1();
 select_t1
-----------

(1 row)

gaussdb=# SELECT s1.query, s1.unique_sql_id FROM dbe_perf.summary_statement s1 WHERE s1.query
= 'SELECT * FROM select_t1();'
gaussdb-# UNION ALL
gaussdb-# SELECT s2.query, s2.unique_sql_id FROM dbe_perf.summary_statement s1,
dbe_perf.summary_statement s2 WHERE s1.query = 'SELECT * FROM select_t1();' AND
s1.unique_sql_id = s2.parent_unique_sql_id;
           query            | unique_sql_id
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
----------------------------+---------------
 SELECT * FROM select_t1(); |    1861705880
 SELECT * FROM t1           |    4115569441
(2 rows)

gaussdb=# SELECT * FROM dbe_stats.get_tables_by_sqlid(1861705880); -- The top-level statement
does not contain the base table.
 nspname | tabname
---------+---------
(0 rows)

gaussdb=# SELECT * FROM dbe_stats.get_tables_by_sqlid(4115569441); -- The sub-statement
contains the base table.
 nspname | tabname
---------+---------
 public  | t1
(1 row)

-- 5. Query statements involving multiple tables in dbe_perf.statement.
gaussdb=# SELECT * FROM t1, t2 LIMIT 1;
 c1 | c1
----+----
  1 |  1
(1 row)

gaussdb=# SELECT query, unique_sql_id FROM dbe_perf.summary_statement WHERE query LIKE
'%SELECT * FROM t1, t2 LIMIT%';
             query             | unique_sql_id
-------------------------------+---------------
 SELECT * FROM t1, t2 LIMIT ?; |    2763456878
(1 row)

gaussdb=# SELECT * FROM dbe_stats.get_tables_by_sqlid(2763456878);
 nspname | tabname
---------+---------
 public  | t1
 public  | t2
(2 rows)

-- Follow-up operations
gaussdb=# DROP PROCEDURE IF EXISTS select_t1;
DROP PROCEDURE
gaussdb=# DROP TABLE IF EXISTS t1;
DROP TABLE
gaussdb=# DROP TABLE IF EXISTS t2;
DROP TABLE
```

- DBE_STATS.ANALYZE_SQLID

  Obtains all tables involved in the statement corresponding to the unique SQL ID and collects statistics about the tables. If multiple tables are successfully collected, multiple rows of records are returned.

  The prototype of DBE_STATS.ANALYZE_SQLID is as follows:

```
DBE_STATS.ANALYZE_SQLID (
    unique_sqlid    BIGINT,
    all_complete    BOOLEAN DEFAULT FALSE,
    node_name       NAME DEFAULT NULL
)
RETURNS TABLE (
    nspname         TEXT,
    tabname         TEXT
);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-399 Parameters and return values of the DBE_STATS.ANALYZE_SQLID API

Column	Data Type	Parameter Type	Can Be Null	Description
unique_sqlid	BIGINT	IN	No	Unique SQL ID, which can be obtained from pg_catalog.pg_stat_activity (view for recording real-time session information), dbe_perf.statement (view for recording normalized SQL statements), or dbe_perf.statement_history (view for recording full or slow SQL statements).
all_complete	BOOLEAN	IN	Yes	Specifies whether to use the **ALL COMPLETE** mode to collect statistics. The behavior is the same as that of **ANALYZE** *tabname* **WITH ALL COMPLETE**. If this parameter is not specified, the default mode is used, which corresponds to **ANALYZE** *tabname*.
node_name	NAME	IN	Yes	CN name, corresponding to **pgxc_node.node_name**. If a CN name is specified, only the SQL statement records on the corresponding CN are queried. If no CN name is specified, the records on all CNs are queried.
nspname	TEXT	OUT	No	Schema name of the table involved in the statement.
tabname	TEXT	OUT	No	Name of the table involved in the statement. Only tables whose statistics are successfully collected are included.

Example:

```
-- The usage of ANALYZE_SQLID is similar to that of GET_TABLES_SQLID. The difference is that this API collects statistics and provides the all_complete option and node_name parameter for filtering.

-- Prerequisites
gaussdb=# DROP TABLE IF EXISTS t1;
DROP TABLE
gaussdb=# CREATE TABLE t1 (c1 int) DISTRIBUTE BY HASH (c1);
CREATE TABLE
gaussdb=# INSERT INTO t1 VALUES (1);
INSERT 0 1
gaussdb=# ANALYZE t1;
ANALYZE
gaussdb=# CREATE OR REPLACE PROCEDURE select_t1()
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb-# AS
gaussdb$# BEGIN
gaussdb$#     PERFORM * FROM t1;
gaussdb$# END;
gaussdb$# /
CREATE PROCEDURE
gaussdb=# DROP TABLE IF EXISTS t2;
DROP TABLE
gaussdb=# CREATE TABLE t2 (c1 int) DISTRIBUTE BY HASH (c1) PARTITION BY LIST (c1) (
gaussdb(#     PARTITION t2_p1 VALUES (1),
gaussdb(#     PARTITION t2_p2 VALUES (2)
gaussdb(# );
CREATE TABLE
gaussdb=# INSERT INTO t2 VALUES (1), (2);
INSERT 0 2
gaussdb=# ANALYZE t2;
ANALYZE

-- 1. Use the unique SQL ID queried from pgxc_stat_activity.
-- Session 1: Simulate a statement that cannot be executed.
gaussdb=# SET cursor_sharing = off; -- This avoids automatic parameterization without parsing.
SET
gaussdb=# SELECT pg_sleep(60) FROM t1;
 pg_sleep
----------

(1 row)

-- Session 2: Query the unique SQL ID and the corresponding table name during the execution of
session 1. The statement needs to be adjusted based on the actual SQL ID.
gaussdb=#  SELECT query, sessionid, unique_sql_id FROM pgxc_stat_activity WHERE query LIKE
'%pg_sleep%t1%' AND query NOT LIKE '%pgxc_stat_activity%';
            query              | sessionid | unique_sql_id
-------------------------------+-----------+---------------
 SELECT pg_sleep(60) FROM t1; |         1 |    3101300097
(1 row)

gaussdb=# SELECT analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public' AND
relname = 't1'; -- Number of previous ANALYZE operations.
 analyze_count
---------------
             1
(1 row)

gaussdb=# SELECT * FROM dbe_stats.analyze_sqlid(3101300097);
 nspname | tabname
---------+---------
 public  | t1
(1 row)

gaussdb=# SELECT analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public' AND
relname = 't1'; -- Ensure that the preceding operation has been analyzed by ANALYZE.
 analyze_count
---------------
             2
(1 row)

-- 2. Use the unique SQL ID queried from dbe_perf.statement.
gaussdb=# SHOW instr_unique_sql_count; -- View the maximum number of records in the statement
view.
 instr_unique_sql_count
------------------------
 200000
(1 row)

gaussdb=# SELECT count(*) FROM dbe_perf.summary_statement; -- Check whether the number of
records in the statement view exceeds the upper limit.
 count
-------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
   46
(1 row)

gaussdb=# SELECT reset_unique_sql('GLOBAL','ALL',0); -- If the number of records reaches the upper
limit, subsequent statements will not be recorded. You can clear the records. (Note: This will cause
SQL records to be lost.)
 reset_unique_sql
------------------
 t
(1 row)

gaussdb=# SELECT * FROM t1;
 c1
----
  1
(1 row)

gaussdb=# SELECT unique_sql_id FROM dbe_perf.summary_statement WHERE query LIKE '%SELECT *
FROM t1%';
 unique_sql_id
---------------
   4115569441
(1 row)

gaussdb=# SELECT analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public' AND
relname = 't1'; -- Number of previous ANALYZE operations.
 analyze_count
---------------
             2
(1 row)

gaussdb=# SELECT * FROM dbe_stats.analyze_sqlid(4115569441);
 nspname | tabname
---------+---------
 public  | t1
(1 row)

gaussdb=# SELECT analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public' AND
relname = 't1'; -- Ensure that the preceding operation has been analyzed by ANALYZE.
 analyze_count
---------------
             3
(1 row)

-- 3. Use the unique SQL ID queried from dbe_stats.statement_history.
gaussdb=# SET log_min_duration_statement = '1s'; -- Slow SQL statements whose execution time is
greater than or equal to 1s will be recorded.
SET
gaussdb=# DELETE FROM pg_catalog.statement_history; -- Avoid being affected by residual slow SQL
records.
DELETE 2
gaussdb=# SELECT pg_sleep(1) FROM t1;
 pg_sleep
----------


(1 row)

gaussdb=# SELECT unique_query_id FROM dbe_perf.statement_history WHERE query LIKE '%pg_sleep
%t1%' LIMIT 1;
 unique_query_id
-----------------
     3101300097
(1 row)

gaussdb=# SELECT analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public' AND
relname = 't1'; -- Number of previous ANALYZE operations.
 analyze_count
---------------
             3
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)

gaussdb=# SELECT * FROM dbe_stats.analyze_sqlid(3101300097);
 nspname | tabname
---------+---------
 public  | t1
(1 row)

gaussdb=# SELECT analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public' AND
relname = 't1'; -- Ensure that the preceding operation has been analyzed by ANALYZE.
 analyze_count
---------------
             4
(1 row)

-- 4. Use the unique SQL ID of the stored procedure queried from the dbe_stats.statement view.
gaussdb=# SHOW instr_unique_sql_track_type; -- Check whether the statement view records include
the sub-statements of the stored procedure.
 instr_unique_sql_track_type
-----------------------------
 all
(1 row)

gaussdb=# SELECT * FROM select_t1();
 select_t1
-----------

(1 row)

gaussdb=# SELECT s1.query, s1.unique_sql_id FROM dbe_perf.summary_statement s1 WHERE s1.query
= 'SELECT * FROM select_t1();'
gaussdb-# UNION ALL
gaussdb-# SELECT s2.query, s2.unique_sql_id FROM dbe_perf.summary_statement s1,
dbe_perf.summary_statement s2 WHERE s1.query = 'SELECT * FROM select_t1();' AND
s1.unique_sql_id = s2.parent_unique_sql_id;
            query             | unique_sql_id
------------------------------+---------------
 SELECT * FROM select_t1(); |    1861705880
 SELECT * FROM t1           |    4115569441
(2 rows)

gaussdb=# SELECT analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public' AND
relname = 't1'; -- Number of previous ANALYZE operations.
 analyze_count
---------------
             4
(1 row)

gaussdb=# SELECT * FROM dbe_stats.analyze_sqlid(1861705880); -- The top-level statement does not
contain the base table.
 nspname | tabname
---------+---------
(0 rows)

gaussdb=# SELECT analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public' AND
relname = 't1'; -- The preceding operation does not trigger the ANALYZE operation on the t1 table.
 analyze_count
---------------
             4
(1 row)

gaussdb=# SELECT * FROM dbe_stats.analyze_sqlid(4115569441); -- The sub-statement contains the
base table.
 nspname | tabname
---------+---------
 public  | t1
(1 row)

gaussdb=# SELECT analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public' AND
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
relname = 't1'; -- Ensure that the preceding operation has been analyzed by ANALYZE.
 analyze_count
---------------
             5
(1 row)

-- 5. Query dbe_perf.statement and collect statistics on the partitioned table in all_complete mode.
gaussdb=# SELECT * FROM t2;
 c1
----
  1
  2
(2 rows)

gaussdb=# SELECT query, unique_sql_id FROM dbe_perf.summary_statement WHERE query LIKE
'%SELECT * FROM t2%';
       query       | unique_sql_id
-------------------+---------------
 SELECT * FROM t2; |     909667705
(1 row)

gaussdb=# SELECT analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public' AND
relname = 't2'; -- Number of previous ANALYZE operations.
 analyze_count
---------------
             1
(1 row)

gaussdb=# SELECT * FROM dbe_stats.analyze_sqlid(909667705, true); -- In all_complete mode, a
larger sample is collected for each partition.
 nspname | tabname
---------+---------
 public  | t2
(1 row)

gaussdb=# SELECT analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public' AND
relname = 't2'; -- Ensure that the preceding operation has been analyzed by ANALYZE.
 analyze_count
---------------
             2
(1 row)

-- 6. Use node_name to filter out SQL statements and collect statistics only on the current CN.
Assume that the current node is data_coordinator1.
gaussdb=# SELECT node_name, node_type, node_port FROM pgxc_node;
     node_name     | node_type | node_port
-------------------+-----------+-----------
 data_coordinator1 | C         |     36415
 data_coordinator2 | C         |     36419
 data_coordinator3 | C         |     36423
 data_datanode1    | D         |     36815
 data_datanode2    | D         |     36819
 data_datanode3    | D         |     36823
(6 rows)

gaussdb=# select * from t2;
 c1
----
  1
  2
(2 rows)

gaussdb=# SELECT unique_sql_id, node_name FROM dbe_perf.summary_statement WHERE query LIKE
'%SELECT * FROM t2%'; -- The statement is recorded on CN 1.
 unique_sql_id |     node_name
---------------+-------------------
      57714958 | data_coordinator1
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public' AND
relname = 't2'; -- Number of previous ANALYZE operations.
 analyze_count
---------------
             2
(1 row)

gaussdb=# SELECT * FROM dbe_stats.analyze_sqlid(57714958, true, 'data_coordinator1');
 nspname | tabname
---------+---------
 public  | t2
(1 row)

gaussdb=# SELECT analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public' AND
relname = 't2'; -- Ensure that the preceding operation has been analyzed by ANALYZE.
 analyze_count
---------------
             3
(1 row)

-- 7. Query and collect statements involving multiple tables from dbe_perf.statement.
gaussdb=# SELECT * FROM t1, t2 LIMIT 1;
 c1 | c1
----+----
  1 |  1
(1 row)

gaussdb=# SELECT query, unique_sql_id FROM dbe_perf.summary_statement WHERE query LIKE
'%SELECT * FROM t1, t2 LIMIT%';
              query              | unique_sql_id
---------------------------------+---------------
 SELECT * FROM t1, t2 LIMIT ?; |    2763456878
(1 row)

gaussdb=# SELECT relname, analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public'
AND relname IN ('t1', 't2');
 relname | analyze_count
---------+---------------
 t1      |             5
 t2      |             3
(2 rows)

gaussdb=# SELECT * FROM dbe_stats.analyze_sqlid(2763456878);
 nspname | tabname
---------+---------
 public  | t1
 public  | t2
(2 rows)

gaussdb=# SELECT relname, analyze_count FROM pg_stat_all_tables WHERE schemaname = 'public'
AND relname IN ('t1', 't2');
 relname | analyze_count
---------+---------------
 t1      |             6
 t2      |             4
(2 rows)

-- Follow-up operations
gaussdb=# DROP PROCEDURE IF EXISTS select_t1;
DROP PROCEDURE
gaussdb=# DROP TABLE IF EXISTS t1;
DROP TABLE
gaussdb=# DROP TABLE IF EXISTS t2;
DROP TABLE
```

- dbe_stats_tools.py

  This Python script is used to simplify the process of exporting and importing database-level statistics.

- To export statistics, you must have the permission on the corresponding API or command (DBE_STATS.CREATE_STAT_TABLE, DBE_STATS.EXPORT_DATABASE_STATS, \COPY TO or DBE_STATS.DROP_STAT_TABLE) and the read and write permissions on the specified export directory.
- To import statistics, you must have the permission on the corresponding API or command (DBE_STATS.CREATE_STAT_TABLE, DBE_STATS.EXPORT_DATABASE_STATS, \COPY FROM or DBE_STATS.DROP_STAT_TABLE) and the read permission on the files in the specified export directory.
- When you export statement metadata, the following information must be collected:
  - Input SQL statement text.
  - The information obtained by performing the EXPLAIN operation on the SQL statement when the hint is effective and ineffective.
  - Definitions of all objects in the database.
  - All GUC parameter settings related to plan optimization, excluding the session-level configuration of applications.
  - Current and historical statistics of the table objects referenced by the SQL statement (exported only when the generated plan is of the stream type).

  To export statement metadata, you must have the system administrator permission on the database and the read and write permissions on the specified directory.

NOTE

This tool is not supported in M-compatible mode.

Table 2-400 dbe_stats_tools.py API parameters

Parameter	Can Be Null	Description
--mode/-m	No	Value range: **dump**, **restore**, **explain**, and **explain_analyze**. • **dump**: export. • **restore**: import. • **explain**: Statement metadata is exported without actually executing the statement. • **explain_analyze**: Statement metadata is exported after the statement is actually executed.
--database/-d	No	Name of the target database to be exported or imported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Can Be Null	Description
--port/-p	No	Port number of the database CN. The value ranges from 1 to 65535.
--user/-U	Yes	Username. If the **-U** parameter is not specified, the current username is used as the input parameter.
--passwd/-W	Yes	User password. If this option is specified, you need to enter the password separately.
--host/-H	Yes	IP address of the target database node to be exported or imported.
--set/-s	Yes	Sets the session-level GUC parameters in statement metadata collection mode. This parameter can be specified for multiple times, corresponding to multiple GUC parameter settings.
--command/-c	No	Specifies the SQL text in statement metadata collection mode. The value must be a single SQL statement that can be executed using **EXPLAIN** and cannot contain the **EXPLAIN** keyword. In **explain_analyze** mode, only the SELECT statement is supported.
--file/-f	No	Specifies the path of the file for storing SQL text in statement metadata collection mode. The value must be a single SQL statement that can be executed using **EXPLAIN** and cannot contain the **EXPLAIN** keyword. In **explain_analyze** mode, only the SELECT statement is supported.
--output/-o	No	Directory for storing exported statistics.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- After the statistics are exported, the **gs_stats.meta** (metadata file) and **gs_stats.csv** (statistics file of the specified database user table) files are generated in the directory specified by the **--output** parameter.
- After the statement metadata is exported, the following directories and files are generated in the directory specified by the **--output** parameter:
  - **query.sql**: stores the entered SQL statement text.
  - **metainfo_nohint**: appears by default and stores information when no hint applies to a statement (the statement lacks a hint, or a tool invalidates the hint).
    - **explain.log**: stores the EXPLAIN result of the SQL statement.
  - **metainfo_hinted**: appears only when a hint takes effect for the statement. The content structure is the same as that of **metainfo_nohint**.
  - **objdefs.sql**: stores the definitions of all objects in the database.
  - **querytuning.conf**: stores all GUC parameter settings related to plan optimization, excluding the session-level configuration of applications.
  - **currstats**: stores the current statistics of the table objects referenced by the SQL statement. The directory structure is the same as that of the output in **dump** mode.
  - **histstats**: stores the historical statistics of the table objects referenced by the SQL statement. Each **histstats**_*<timestamp>* subdirectory in this directory corresponds to a statistics snapshot of a timestamp. The file structure in the subdirectory is the same as that of the output in **dump** mode. **histstats** can store up to 50 historical statistics snapshots. If you query more than 50 snapshots, it returns 50 samples taken at equal intervals.

Example 1: Export and import statistics.

```
# The following is an example of exporting statistics.
[omm@localhost ~]$ python3 dbe_stats_tools.py -m dump -d db1 -p 8000 --output ./stats
export table name: "export_table_1749782634"
execute: "gsql -d db1 -p 8000 -U omm -c "select 1 from pg_catalog.pg_class where relname =
'export_table_1749782634' and relnamespace = 2200;""
execute: "gsql -d db1 -p 8000 -U omm -c "START TRANSACTION;call
dbe_stats.create_stat_table(ownname => 'public', stattab => 'export_table_1749782634');call
dbe_stats.export_database_stats(stattab=>'export_table_1749782634',statown=>'public');\COPY
public.export_table_1749782634TO '/home/omm/stats/gs_stats.csv' WITH (FORMAT CSV,
HEADER);call dbe_stats.drop_stat_table(ownname => 'public', stattab =>
'export_table_1749782634');COMMIT;""
execute: "gsql -d db1 -p 8000 -U omm -c "select pg_catalog.gs_get_current_version() from dual;""
deployment: "M"
execute: "gsql -d db1 -p 8000 -U omm -c "select count(1) from pg_catalog.pgxc_node where
node_type = 'D';""
datanode number: "6"
dump complete!

# After the statistics are exported, the metadata file and dump file of statistics are generated in the
stats directory.
[omm@localhost ~]$ ls stats/
gs_stats.csv  gs_stats.meta

# The following is an example of importing statistics.
[omm@localhost ~]$ python3 dbe_stats_tools.py -m restore -d db1 -p 8000 --output ./stats
export table name: "export_table_1749782643"
gs_stats meta file: "/home/omm/stats/gs_stats.meta"
deployment: "M"
execute: "gsql -d db1 -p 8000 -U omm -c "select pg_catalog.gs_get_current_version() from dual;""
datanode number: "6"
execute: "gsql -d db1 -p 8000 -U omm -c "select count(1) from pg_catalog.pgxc_node where
node_type = 'D';""
execute: "gsql -d db1 -p 8000 -U omm -c "select 1 from pg_catalog.pg_class where relname =
'export_table_1749782643' and relnamespace = 2200;""
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
execute: "gsql -d db1 -p 8000 -U omm -c "START TRANSACTION;call
dbe_stats.create_stat_table(ownname => 'public', stattab => 'export_table_1749782643');\COPY
public.export_table_1749782643 FROM '/home/omm/stats/gs_stats.csv' WITH (FORMAT CSV,
HEADER);call
dbe_stats.import_database_stats(stattab=>'export_table_1749782643',statown=>'public');call
dbe_stats.drop_stat_table(ownname => 'public', stattab => 'export_table_1749782643');COMMIT;""
restore complete!
```

Example 2: Reproduce the problem of poor SQL plan using statement metadata. This example contains a typical poor plan. After data changes, statistics are not collected in a timely manner. As a result, the plan with an incorrect index is selected.

Preparations: Create a table, insert data, and reproduce the slow SQL service.

```
# In the example, db1 is an ORA-compatible database. Replace it with the actual database name.
# In the example, the node where the command is executed is the cluster node.
[omm@localhost ~]$ gsql -d db1 -p 8000 -r

-- To prevent cardinality feedback, plan reuse, and statement pushdown from affecting the
reproduction of the slow SQL problem, set the following GUC parameters:
db1=# SET enable_feedback_cardest = OFF;
SET
db1=# SET cursor_sharing = OFF;
SET
db1=# SET enable_fast_query_shipping = OFF;
SET
db1=# SET explain_perf_mode = 'pretty';
SET

-- 2. Preset data. The s table is used only for data generation, and the t table is the service table to be
queried.
db1=# DROP TABLE IF EXISTS s;
NOTICE:  table "s" does not exist, skipping
DROP TABLE
db1=# CREATE TABLE s (
    i INT
) DISTRIBUTE BY HASH (i);
CREATE TABLE
db1=# INSERT INTO s SELECT generate_series(1, 550000);
INSERT 0 550000
db1=# ANALYZE s;
ANALYZE
db1=# DROP TABLE IF EXISTS t;
NOTICE:  table "t" does not exist, skipping
DROP TABLE
db1=# CREATE TABLE t (
    idate INT,
    no    INT
) DISTRIBUTE BY HASH (idate) PARTITION BY RANGE (idate) (
    PARTITION p1 VALUES LESS THAN (200),
    PARTITION p2 VALUES LESS THAN (400),
    PARTITION p3 VALUES LESS THAN (MAXVALUE)
);
CREATE TABLE
db1=# CREATE INDEX t_i  ON t(idate) LOCAL;
CREATE INDEX
db1=# CREATE INDEX t_no ON t(no)    LOCAL;
CREATE INDEX
db1=# INSERT INTO t
SELECT floor(random() * 540) AS idate,
       floor(random() * 30)  AS no
  FROM s
 WHERE i <= 540000;
INSERT 0 540000
db1=# ANALYZE t;
ANALYZE

-- 3. Insert new data but do not execute ANALYZE again.
db1=# INSERT INTO t
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
SELECT floor(540 + random() * 10) AS idate,
       floor(random() * 30)       AS no
  FROM s
 WHERE i <= 1000000;
INSERT 0 550000

-- 4. Query the new data (default plan vs. hint plan).
db1=# EXPLAIN ANALYZE
SELECT *
  FROM t
 WHERE idate >= 540 AND idate < 550 AND no = 1;
 id |                  operation                   |       A-time       | A-rows | E-rows | Peak Memory | A-width | E-
width | E-costs
----+----------------------------------------------+--------------------+--------+--------+-------------+---------
+---------+---------
  1 | ->  Streaming (type: GATHER)                 | 144.711            |  18204 |      1 | 81KB        |         |
8 | 2.60
  2 |    ->  Partitioned Index Scan using t_i on t | [100.696,130.653] |  18204 |      1 | [78KB,78KB]
|         |       8 | 2.47
(2 rows)

     Predicate Information (identified by plan id)
-----------------------------------------------------------
   2 --Partitioned Index Scan using t_i on t
         Index Cond: ((idate >= 540) AND (idate < 550))
         Filter: (no = 1)
         Rows Removed by Filter: 531796
         Selected Partitions:  3
(5 rows)

 Memory Information (identified by plan id)
--------------------------------------------
 Coordinator Query Peak Memory:
         Query Peak Memory: 1MB
 Datanode:
         Max Query Peak Memory: 0MB
         Min Query Peak Memory: 0MB
(5 rows)

                        User Define Profiling
---------------------------------------------------------------------------
 Plan Node id: 1  Track name: coordinator get datanode connection
  (actual time=[0.062, 0.062], calls=[1, 1])
 Plan Node id: 1  Track name: Coordinator serialize plan
  (actual time=[1.085, 1.085], calls=[1, 1])
 Plan Node id: 1  Track name: Coordinator send begin command
  (actual time=[0.003, 0.003], calls=[1, 1])
 Plan Node id: 1  Track name: Coordinator start transaction and send query
  (actual time=[0.053, 0.053], calls=[1, 1])
(8 rows)

                             ====== Query Summary =====
-----------------------------------------------------------------------------------------------
 Datanode executor start time [dn_6001_6002_6003, dn_6004_6005_6006]: [0.229 ms,0.261 ms]
 Datanode executor run time [dn_6007_6008_6009, dn_6001_6002_6003]: [111.230 ms,144.077 ms]
 Datanode executor end time [dn_6004_6005_6006, dn_6001_6002_6003]: [0.012 ms,0.022 ms]
 Coordinator executor start time: 0.095 ms
 Coordinator executor run time: 146.583 ms
 Coordinator executor end time: 0.044 ms
 Planner runtime: 0.474 ms
 Plan size: 3857 byte
 Query Id: 72339069015809757
 Total runtime: 146.743 ms
(10 rows)

db1=# EXPLAIN ANALYZE
SELECT /*+ indexscan(t t_no) */ *
  FROM t
 WHERE idate >= 540 AND idate < 550 AND no = 1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 id |                  operation                  |     A-time     | A-rows | E-rows | Peak Memory | A-width | E-
width | E-costs
----+---------------------------------------------+----------------+--------+--------+-------------+---------
+---------+---------
  1 | ->  Streaming (type: GATHER)                | 25.360         |  18204 |  17652 | 81KB        |         |
8 | 4265.30
  2 |    ->  Partitioned Index Scan using t_no on t | [5.622,8.476] |  18204 |  17652 | [78KB,78KB] |
|       8 | 3529.80
(2 rows)

   Predicate Information (identified by plan id)
-------------------------------------------------------
   2 --Partitioned Index Scan using t_no on t
         Index Cond: (no = 1)
         Filter: ((idate >= 540) AND (idate < 550))
         Rows Removed by Filter: 4554
         Selected Partitions:  3
(5 rows)

 Memory Information (identified by plan id)
---------------------------------------------
 Coordinator Query Peak Memory:
         Query Peak Memory: 1MB
 Datanode:
         Max Query Peak Memory: 0MB
         Min Query Peak Memory: 0MB
(5 rows)

                        User Define Profiling
---------------------------------------------------------------------------
 Plan Node id: 1  Track name: coordinator get datanode connection
  (actual time=[0.812, 0.812], calls=[1, 1])
 Plan Node id: 1  Track name: Coordinator serialize plan
  (actual time=[1.924, 1.924], calls=[1, 1])
 Plan Node id: 1  Track name: Coordinator send begin command
  (actual time=[0.003, 0.003], calls=[1, 1])
 Plan Node id: 1  Track name: Coordinator start transaction and send query
  (actual time=[0.058, 0.058], calls=[1, 1])
(8 rows)

                             ====== Query Summary =====
-----------------------------------------------------------------------------------------
 Datanode executor start time [dn_6004_6005_6006, dn_6007_6008_6009]: [0.397 ms,0.584 ms]
 Datanode executor run time [dn_6004_6005_6006, dn_6007_6008_6009]: [12.902 ms,21.639 ms]
 Datanode executor end time [dn_6004_6005_6006, dn_6007_6008_6009]: [0.011 ms,0.019 ms]
 Coordinator executor start time: 0.165 ms
 Coordinator executor run time: 26.601 ms
 Coordinator executor end time: 0.038 ms
 Planner runtime: 1.037 ms
 Plan size: 3886 byte
 Query Id: 72339069015811437
 Total runtime: 26.853 ms
(10 rows)

db1=# \q
```

Scenario 1: Use the **-c** option to collect metadata of SQL statements with hints and compare the plan information when the hints take effect and when they do not take effect.

```
# Specify the SQL text with hints and collect its metadata.
[omm@localhost ~]$ rm -rf ./hint_metainfo
[omm@localhost ~]$ python3 $GAUSSHOME/bin/dbe_stats_tools.py -m explain_analyze -d db1 -p
8000 -s enable_fast_query_shipping=off -c 'SELECT /*+ indexscan(t t_no) */ * FROM t WHERE idate >=
540 AND idate < 550 AND no = 1;' -o ./hint_metainfo
execute: "gsql -d db1 -p 8000 -U omm -q -f -
START TRANSACTION READ ONLY;
set enable_fast_query_shipping=E'off';
set explain_perf_mode = pretty; set cursor_sharing = off;
explain (metadata on, tables on, analyze on) SELECT /*+ indexscan(t t_no) */ * FROM t WHERE idate
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
>= 540 AND idate < 550 AND no = 1;
ROLLBACK;"
execute: "gsql -d db1 -p 8000 -U omm -q -f -
START TRANSACTION READ ONLY;
set enable_fast_query_shipping=E'off';
set explain_perf_mode = pretty; set cursor_sharing = off;
set parser_ignore_hints = on;
explain (metadata on, tables on, analyze on) SELECT /*+ indexscan(t t_no) */ * FROM t WHERE idate
>= 540 AND idate < 550 AND no = 1;
ROLLBACK;"
explaining query completed!
dumping query tuning settings completed!
dumping object definition completed!
dumping current statistics completed!
dumping historical statistics completed!

# Obtain the plan information of the statement when the hint takes effect or does not take effect,
and other information that can be used for reproduction.
[omm@localhost ~]$ ls -R ./hint_metainfo/
./hint_metainfo/:
currstats  histstats  metainfo_hinted  metainfo_nohint  objdefs.sql  query.sql  querytuning.conf

./hint_metainfo/currstats:
gs_stats.csv  gs_stats.meta

./hint_metainfo/histstats:
histstats_2026-07-21_15:34:04.161597+08  histstats_2026-07-21_15:35:43.781149+08
histstats_2026-07-21_15:35:43.970977+08  histstats_2026-07-21_15:35:44.108525+08
histstats_2026-07-21_15:35:44.255638+08

'./hint_metainfo/histstats/histstats_2026-07-21_15:34:04.161597+08':
gs_stats.csv  gs_stats.meta

'./hint_metainfo/histstats/histstats_2026-07-21_15:35:43.781149+08':
gs_stats.csv  gs_stats.meta

'./hint_metainfo/histstats/histstats_2026-07-21_15:35:43.970977+08':
gs_stats.csv  gs_stats.meta

'./hint_metainfo/histstats/histstats_2026-07-21_15:35:44.108525+08':
gs_stats.csv  gs_stats.meta

'./hint_metainfo/histstats/histstats_2026-07-21_15:35:44.255638+08':
gs_stats.csv  gs_stats.meta

./hint_metainfo/metainfo_hinted:
explain.log

./hint_metainfo/metainfo_nohint:
explain.log
```

Scenario 2: Use the **-f** option to collect metadata of the corresponding statement and try to reproduce the current and historical execution plans in the test database.

```
# Use gsql to connect to the t table and perform ANALYZE on the table to update statistics.
[omm@localhost ~]$ gsql -d db1 -p 8000 -r

db1=# SET enable_feedback_cardest = OFF;
SET
db1=# SET cursor_sharing = OFF;
SET
db1=# SET enable_fast_query_shipping = OFF;
SET
db1=# SET explain_perf_mode = 'pretty';
SET

-- Run the ANALYZE statement to solve the slow SQL problem.
db1=# ANALYZE t;
ANALYZE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- After the statistics are collected again, the plan is optimal by default.
db1=# EXPLAIN ANALYZE
SELECT *
  FROM t
 WHERE idate >= 540 AND idate < 550 AND no = 1;
 id |                    operation                    |    A-time     | A-rows | E-rows |  Peak Memory  | A-width |
E-width | E-costs
----+-------------------------------------------------+---------------+--------+--------+---------------
+---------+---------+---------
  1 | ->  Streaming (type: GATHER)                    | 25.861        |  18204 |  18636 | 82KB          |
|       8 | 4204.50
  2 |    ->  Partitioned Bitmap Heap Scan on t        | [8.731,9.053] |  18204 |  18636 | [23KB,23KB]
|         |       8 | 3428.00
  3 |       ->  Partitioned Bitmap Index Scan using t_no | [1.066,1.100] |  22758 |  23400 |
[178KB,200KB] |         |       0 | 60.95
(3 rows)

   Predicate Information (identified by plan id)
-----------------------------------------------------
   2 --Partitioned Bitmap Heap Scan on t
         Recheck Cond: (no = 1)
         Filter: ((idate >= 540) AND (idate < 550))
         Rows Removed by Filter: 4554
         Selected Partitions:  3
   3 --Partitioned Bitmap Index Scan using t_no
         Index Cond: (no = 1)
(7 rows)

 Memory Information (identified by plan id)
-----------------------------------------------
 Coordinator Query Peak Memory:
        Query Peak Memory: 1MB
 Datanode:
        Max Query Peak Memory: 0MB
        Min Query Peak Memory: 0MB
(5 rows)

                        User Define Profiling
---------------------------------------------------------------------------------
 Plan Node id: 1  Track name: coordinator get datanode connection
  (actual time=[1.722, 1.722], calls=[1, 1])
 Plan Node id: 1  Track name: Coordinator serialize plan
  (actual time=[1.236, 1.236], calls=[1, 1])
 Plan Node id: 1  Track name: Coordinator send begin command
  (actual time=[0.003, 0.003], calls=[1, 1])
 Plan Node id: 1  Track name: Coordinator start transaction and send query
  (actual time=[0.048, 0.048], calls=[1, 1])
(8 rows)

                             ====== Query Summary =====
-------------------------------------------------------------------------------------------
 Datanode executor start time [dn_6001_6002_6003, dn_6007_6008_6009]: [0.388 ms,0.550 ms]
 Datanode executor run time [dn_6001_6002_6003, dn_6007_6008_6009]: [19.315 ms,22.296 ms]
 Datanode executor end time [dn_6004_6005_6006, dn_6007_6008_6009]: [0.022 ms,0.033 ms]
 Coordinator executor start time: 0.113 ms
 Coordinator executor run time: 27.502 ms
 Coordinator executor end time: 0.032 ms
 Planner runtime: 0.594 ms
 Plan size: 3924 byte
 Query Id: 72339069015818205
 Total runtime: 27.669 ms
(10 rows)

db1=# \q

# Specify an SQL text file to collect statement metadata.
[omm@localhost ~]$ echo 'SELECT * FROM t WHERE idate >= 540 AND idate < 550 AND no = 1;' > ./
tmp.sql
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
[omm@localhost ~]$ python3 $GAUSSHOME/bin/dbe_stats_tools.py -m explain_analyze -d db1 -p
8000 -s enable_fast_query_shipping=off -f ./tmp.sql -o ./hist_metainfo
execute: "gsql -d db1 -p 8000 -U omm -q -f -
START TRANSACTION READ ONLY;
set enable_fast_query_shipping=E'off';
set explain_perf_mode = pretty; set cursor_sharing = off;
explain (metadata on, tables on, analyze on) SELECT * FROM t WHERE idate >= 540 AND idate < 550
AND no = 1;
ROLLBACK;"
explaining query completed!
dumping query tuning settings completed!
dumping object definition completed!
dumping current statistics completed!
dumping historical statistics completed!

# The exported statement metadata contains multiple historical statistics snapshots.
[omm@localhost ~]$ ls -R ./hist_metainfo
./hist_metainfo:
currstats  histstats  metainfo_nohint  objdefs.sql  query.sql  querytuning.conf

./hist_metainfo/currstats:
gs_stats.csv  gs_stats.meta

./hist_metainfo/histstats:
histstats_2026-07-21_15:34:04.161597+08  histstats_2026-07-21_15:35:43.970977+08
histstats_2026-07-21_15:35:44.255638+08
histstats_2026-07-21_15:35:43.781149+08  histstats_2026-07-21_15:35:44.108525+08
histstats_2026-07-21_15:38:27.837751+08

'./hist_metainfo/histstats/histstats_2026-07-21_15:34:04.161597+08':
gs_stats.csv  gs_stats.meta

'./hist_metainfo/histstats/histstats_2026-07-21_15:35:43.781149+08':
gs_stats.csv  gs_stats.meta

'./hist_metainfo/histstats/histstats_2026-07-21_15:35:43.970977+08':
gs_stats.csv  gs_stats.meta

'./hist_metainfo/histstats/histstats_2026-07-21_15:35:44.108525+08':
gs_stats.csv  gs_stats.meta

'./hist_metainfo/histstats/histstats_2026-07-21_15:35:44.255638+08':
gs_stats.csv  gs_stats.meta

'./hist_metainfo/histstats/histstats_2026-07-21_15:38:27.837751+08':
gs_stats.csv  gs_stats.meta

./hist_metainfo/metainfo_nohint:
explain.log

# Create a test database and import the object definitions and statistics.
[omm@localhost ~]$ gsql -d postgres -p 8000 -c 'DROP DATABASE IF EXISTS db_test;'
NOTICE:  database "db_test" does not exist, skipping
DROP DATABASE
[omm@localhost ~]$ gsql -d postgres -p 8000 -c "CREATE DATABASE db_test WITH dbcompatibility =
'ORA';"
CREATE DATABASE
[omm@localhost ~]$ gsql -d db_test -p 8000 -q -f ./hist_metainfo/objdefs.sql
total time: 386  ms

# Try to import the current statistics to reproduce the execution plan after the repair.
[omm@localhost ~]$ python3 $GAUSSHOME/bin/dbe_stats_tools.py -m restore -d db_test -p 8000 --
output ./hist_metainfo/currstats
export table name: "export_table_1784619829"
execute: "gsql -d db_test -p 8000 -U omm -q -f -
select pg_catalog.gs_get_current_version() from dual;"
gs_stats meta file: "/home/omm/hist_metainfo/currstats/gs_stats.meta"
deployment: "M"
datanode number: "3"
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
execute: "gsql -d db_test -p 8000 -U omm -q -f -
select pg_catalog.count(1) from pg_catalog.pgxc_node where node_type = 'D';"
execute: "gsql -d db_test -p 8000 -U omm -q -f -
select 1 from pg_catalog.pg_class where relname = 'export_table_1784619829' and relnamespace =
2200;"
execute: "gsql -d db_test -p 8000 -U omm -q -f -
START TRANSACTION;call dbe_stats.create_stat_table(ownname => 'public', stattab =>
'export_table_1784619829');\COPY public.export_table_1784619829 FROM '/home/omm/
hist_metainfo/currstats/gs_stats.csv' WITH (FORMAT CSV, HEADER);call
dbe_stats.import_database_stats(stattab=>'export_table_1784619829',statown=>'public');call
dbe_stats.drop_stat_table(ownname => 'public', stattab => 'export_table_1784619829');COMMIT;"
restore complete!
[omm@localhost ~]$ gsql -d db_test -p 8000 -c "SET cursor_sharing = OFF; SET
enable_fast_query_shipping = OFF; SET enable_feedback_cardest = OFF; SET explain_perf_mode =
'pretty'; EXPLAIN SELECT * FROM t WHERE idate >= 540 AND idate < 550 AND no = 1;"
SET
SET
SET
SET
 id |                    operation                     | E-rows | E-width | E-costs
----+--------------------------------------------------+--------+---------+---------
  1 | ->  Streaming (type: GATHER)                     |  18636 |       8 | 4204.50
  2 |    ->  Partitioned Bitmap Heap Scan on t         |  18636 |       8 | 3428.00
  3 |       ->  Partitioned Bitmap Index Scan using t_no |  23400 |       0 | 60.95
(3 rows)

   Predicate Information (identified by plan id)
-------------------------------------------------
  2 --Partitioned Bitmap Heap Scan on t
        Recheck Cond: (no = 1)
        Filter: ((idate >= 540) AND (idate < 550))
        Selected Partitions:  3
  3 --Partitioned Bitmap Index Scan using t_no
        Index Cond: (no = 1)
(6 rows)

# Try to import historical statistics to reproduce the execution plan before the repair (select the
timestamp closest to the time when the problem occurred).
[omm@localhost ~]$ python3 $GAUSSHOME/bin/dbe_stats_tools.py -m restore -d db_test -p 8000 --
output ./hist_metainfo/histstats/histstats_2026-07-21_15\:34\:04.161597+08
export table name: "export_table_1784620247"
execute: "gsql -d db_test -p 8000 -U omm -q -f -
select pg_catalog.gs_get_current_version() from dual;"
gs_stats meta file: "/home/omm/hist_metainfo/histstats/histstats_2026-07-21_15:34:04.161597+08/
gs_stats.meta"
deployment: "M"
datanode number: "3"
execute: "gsql -d db_test -p 8000 -U omm -q -f -
select pg_catalog.count(1) from pg_catalog.pgxc_node where node_type = 'D';"
execute: "gsql -d db_test -p 8000 -U omm -q -f -
select 1 from pg_catalog.pg_class where relname = 'export_table_1784620247' and relnamespace =
2200;"
execute: "gsql -d db_test -p 8000 -U omm -q -f -
START TRANSACTION;call dbe_stats.create_stat_table(ownname => 'public', stattab =>
'export_table_1784620247');\COPY public.export_table_1784620247 FROM '/home/omm/
hist_metainfo/histstats/histstats_2026-07-21_15:34:04.161597+08/gs_stats.csv' WITH (FORMAT CSV,
HEADER);call
dbe_stats.import_database_stats(stattab=>'export_table_1784620247',statown=>'public');call
dbe_stats.drop_stat_table(ownname => 'public', stattab => 'export_table_1784620247');COMMIT;"
restore complete!
[omm@localhost ~]$ gsql -d db_test -p 8000 -c 'SET cursor_sharing = OFF; SET
enable_fast_query_shipping = OFF; SET enable_feedback_cardest = OFF; SET explain_perf_mode =
'pretty'; EXPLAIN SELECT * FROM t WHERE idate >= 540 AND idate < 550 AND no = 1;'
SET
SET
SET
SET
 id |                  operation                   | E-rows | E-width | E-costs
----+----------------------------------------------+--------+---------+---------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 1 | ->  Streaming (type: GATHER)                |      1 |       8 | 2.60
 2 |    ->  Partitioned Index Scan using t_i on t |      1 |       8 | 2.47
(2 rows)

    Predicate Information (identified by plan id)
-------------------------------------------------------------
  2 --Partitioned Index Scan using t_i on t
        Index Cond: ((idate >= 540) AND (idate < 550))
        Filter: (no = 1)
        Selected Partitions:  3
(4 rows)
```

Follow-up operations: Clear database objects and export directory files.

```
# Clear the database objects.
[omm@localhost ~]$ gsql -d db1 -p 8000 -r
db1=# DROP TABLE s CASCADE;
DROP TABLE
db1=# DROP TABLE t CASCADE;
DROP TABLE
db1=# DROP DATABASE db_test;
DROP DATABASE
db1=# \q

# Clear the exported files.
[omm@localhost ~]$ rm -rf ./hint_metainfo
[omm@localhost ~]$ rm -rf ./hist_metainfo
```

2.12.2.19 DBE_TASK

APIs

Table 2-401 lists APIs supported by the DBE_TASK package.

Table 2-401 DBE_TASK

API	Description
DBE_TASK.SUBMIT	Submits a scheduled job. The job ID is automatically generated by the system.
DBE_TASK.JOB_SUBMIT	Provides the same function as DBE_TASK.SUBMIT, with syntax compatibility parameters.
DBE_TASK.ID_SUBMIT	Submits a scheduled job. The job ID is specified by the user.
DBE_TASK.CANCEL	Deletes a scheduled job by job ID.
DBE_TASK.RUN	Runs a scheduled job.
DBE_TASK.FINISH	Disables or enables scheduled job execution.
DBE_TASK.UPDATE	Modifies attributes of a scheduled job, including the job content, next-execution time, and execution interval.
DBE_TASK.CHANGE	Provides the same function as DBE_TASK.UPDATE, with syntax compatibility parameters.
DBE_TASK.CONTENT	Modifies the content attribute of a scheduled job.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_TASK.NEXT_TIME	Modifies the next-execution time attribute of a scheduled job.
DBE_TASK.INTERVAL	Modifies the execution interval attribute of a scheduled job.

NOTE

- DBE_TASK scheduling in a distributed environment does not ensure high reliability of job execution status information. The execution status of job scheduling can be queried only on the node where the job is submitted. The information queried on other nodes may not be synchronized.
- GaussDB provides a highly available, reliable, and flexible scheduling system using the DBE_SCHEDULER API (recommended in a distributed environment).

- DBE_TASK.SUBMIT

  DBE_TASK.SUBMIT submits a system scheduled job.

  The prototype of DBE_TASK.SUBMIT is as follows:

```
DBE_TASK.SUBMIT(
what          IN   TEXT,
next_time     IN   TIMESTAMP DEFAULT sysdate,
interval_time IN   TEXT  DEFAULT 'null',
id            OUT  INTEGER
)RETURN INTEGER;
```

  NOTE

  - When a scheduled job is created (using DBE_TASK), the system binds the current database and the username to the job by default.
  - The function can be called using the CALL or SELECT statement. If the function is called using the CALL statement, you do not need to set the output parameter. If the function is called using the SELECT statement, you need to set the output parameter. To call this function within a stored procedure, use the PERFORM statement.
  - If the submitted SQL statement job uses a non-**PUBLIC** schema, specify the schema where a table or a function resides, or run **SET current_schema =** *xxx*; before executing the SQL statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-402 DBE_TASK.SUBMIT API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
what	Text	IN	No	SQL statement to be executed. One or multiple DDL operations (excluding database-related operations), DML operations, anonymous blocks, and statements for calling stored procedures, or all four combined are supported.
next_time	Timestamp	IN	No	Next time the job will be executed. The default value is the current system time (**sysdate**). If the specified time has past, the job is executed immediately after being submitted.
interval_time	Text	IN	Yes	Expression for calculating the next time the job will be executed. It can be an interval expression, or **sysdate** followed by a numeric value, for example, **sysdate +1.0/24**. If the value is empty or **"null"**, it is executed only once. After execution, the job status is changed to **'d'**, indicating that the job is no longer executed.
id	Integer	OUT	No	Job ID. The value ranges from 1 to 3000000. When the SELECT statement is used for calling, this parameter cannot be added. When the CALL statement is used for calling, this parameter must be added.

NOTE

When you create a user using the **what** parameter, the plaintext password of the user is recorded in the log. Therefore, you are advised not to do so. Jobs created using this API may not be highly available. You are advised to use **PKG_SERVICE.SUBMIT_ON_NODES** to create a job and specify the CCN as the job execution node.

Example:

```
gaussdb=# SELECT DBE_TASK.SUBMIT('call pro_xxx();', to_date('20180101','yyyymmdd'),'sysdate+1');
 submit
--------
  31031
(1 row)

gaussdb=# SELECT DBE_TASK.SUBMIT('call pro_xxx();', to_date('20180101','yyyymmdd'),'sysdate
+1.0/24');
 submit
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
--------
   512
(1 row)

gaussdb=# DECLARE
gaussdb-#     jobid int;
gaussdb-# BEGIN
gaussdb$#     PERFORM DBE_TASK.SUBMIT('call pro_xxx();', sysdate, 'interval ''5 minute''', jobid);
gaussdb$# END;
gaussdb$# /
ANONYMOUS BLOCK EXECUTE
```

- DBE_TASK.JOB_SUBMIT

  Submits a system scheduled job. In addition, it provides additional compatibility parameters.

  The prototype of DBE_TASK.JOB_SUBMIT is as follows:

```
DBE_TASK.JOB_SUBMIT(
job           OUT  INTEGER,
what          IN   TEXT,
next_date     IN   TIMESTAMP  DEFAULT sysdate,
job_interval  IN   TEXT       DEFAULT 'null',
no_parse      IN   BOOLEAN    DEFAULT false,
instance      IN   INTEGER    DEFAULT 0,
force         IN   BOOLEAN    DEFAULT false
)RETURN INTEGER;
```

Table 2-403 DBE_TASK.JOB_SUBMIT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
job	Integer	OUT	No	Job ID. The value ranges from 1 to 3000000. When DBE_TASK.JOB_SUBMIT is called using the SELECT statement, this parameter can be omitted.
what	Text	IN	No	SQL statement to be executed. One or multiple DDL operations (excluding database-related operations), DML operations, anonymous blocks, and statements for calling stored procedures, or all four combined are supported.
next_date	Timestamp	IN	Yes	Next time the job will be executed. The default value is the current system time (**sysdate**). If the specified time has past, the job is executed immediately after being submitted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
job_interval	Text	IN	Yes	Expression for calculating the next time the job will be executed. It can be an interval expression, or **sysdate** followed by a numeric value, for example, **sysdate +1.0/24**. If the value is empty or **"null"**, it is executed only once. After execution, the job status is changed to **'d'**, indicating that the job is no longer executed.
no_parse	Boolean	IN	Yes	The default value is **false**, which is used only for syntax compatibility.
instance	Integer	IN	Yes	The default value is **0**, which is used only for syntax compatibility.
force	Boolean	IN	Yes	The default value is **false**, which is used only for syntax compatibility.

Example:

```
gaussdb=# DECLARE
gaussdb-#     id integer;
gaussdb-# BEGIN
gaussdb$#     id = DBE_TASK.JOB_SUBMIT(
gaussdb$#         what => 'insert into t1 values (1, 2)',
gaussdb$#         job_interval => 'sysdate + 1' --daily
gaussdb$#     );
gaussdb$# END;
gaussdb$# /
ANONYMOUS BLOCK EXECUTE
```

- DBE_TASK.ID_SUBMIT

  DBE_TASK.ID_SUBMIT has the same syntax function as DBE_TASK.SUBMIT. All parameters of DBE_TASK.ID_SUBMIT are input parameters, with the first parameter specifying a job ID. The last parameter of DBE_TASK.SUBMIT is an output parameter, specifying the job ID automatically generated by the system.

  The prototype of DBE_TASK.ID_SUBMIT is as follows:

  ```
  DBE_TASK.ID_SUBMIT(
  id            IN   BIGINT,
  what          IN   TEXT,
  next_time     IN   TIMESTAMP DEFAULT sysdate,
  interval_time IN   TEXT  DEFAULT 'null');
  ```

  Example:

  ```
  gaussdb=# CALL dbe_task.id_submit(101, 'insert_msg_statistic1;', sysdate, 'sysdate+3.0/24');
   id_submit
  -----------


  (1 row)
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_TASK.CANCEL

  Deletes a specified scheduled job.

  The prototype of DBE_TASK.CANCEL is as follows:

  ```
  CANCEL(id  IN  INTEGER);
  ```

  Table 2-404 DBE_TASK.CANCEL API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
id	Integer	IN	No	Specified job ID.

  Example:

  ```
  gaussdb=# CALL dbe_task.cancel(101);
   cancel
  --------

  (1 row)
  ```

- DBE_TASK.RUN

  Executes a scheduled job.

  The prototype of DBE_TASK.RUN is as follows:

  ```
  DBE_TASK.RUN(
  job          IN   BIGINT,
  force        IN   BOOLEAN DEFAULT FALSE);
  ```

  Table 2-405 DBE_TASK.RUN API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
job	bigint	IN	No	Specified job ID.
force	Boolean	IN	Yes	Used only for syntax compatibility.

  Example:

  ```
  gaussdb=# BEGIN
  gaussdb$#     DBE_TASK.ID_SUBMIT(12345, 'insert_msg_statistic1;', sysdate, 'sysdate+3.0/24');
  gaussdb$#     DBE_TASK.RUN(12345);
  gaussdb$# END;
  gaussdb$# /
  ANONYMOUS BLOCK EXECUTE
  ```

- DBE_TASK.FINISH

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Disables or enables a scheduled job.

The prototype of DBE_TASK.FINISH is as follows:

```
DBE_TASK.FINISH(
id          IN   INTEGER,
broken      IN   BOOLEAN,
next_time   IN   TIMESTAMP  DEFAULT  sysdate);
```

Table 2-406 DBE_TASK.FINISH API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
id	Integer	IN	No	Specified job ID.
broken	Boolean	IN	No	Status flag. **true** means disabled; **false** means enabled. The current job is updated based on the parameter value **true** or **false**. If it is null, the job status remains unchanged.
next_time	Timestamp	IN	Yes	Next execution time. The default value is the current system time. This parameter can be omitted, and its default value will be used in this case. • If the value of **broken** is **true**, the value of this parameter is updated to **'4000-1-1'**. • If the parameter **broken** is set to **false** and the parameter **next_time** is not null, the **next_time** value of a specified job is updated. If the parameter **next_time** is null, the **next_time** value is not updated.

Example:

```
gaussdb=# CALL dbe_task.id_submit(101, 'insert_msg_statistic1;', sysdate, 'sysdate+3.0/24');
 id_submit
-----------

(1 row)

gaussdb=# CALL dbe_task.finish(101, true);
 finish
--------

(1 row)

gaussdb=# CALL dbe_task.finish(101, false, sysdate);
 finish
--------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
(1 row)
```

- DBE_TASK.UPDATE

  Updates user-defined attributes of a scheduled job, including the job content, next-execution time, and execution interval.

  The prototype of DBE_TASK.UPDATE is as follows:

```
dbe_task.UPDATE(
id             IN     INTEGER,
content        IN     TEXT,
next_time      IN     TIMESTAMP,
interval_time  IN  TEXT);
```

Table 2-407 DBE_TASK.UPDATE API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
id	Integer	IN	No	Specified job ID.
content	Text	IN	Yes	Name of the stored procedure or SQL statement block that is executed. If this parameter is null, the system does not update the **content** parameter for the specified job. Otherwise, the system updates the **content** parameter for the specified job.
next_time	Timestamp	IN	Yes	Next execution time. If this parameter is null, the system does not update the **next_time** parameter for the specified job. Otherwise, the system updates the **next_time** parameter for the specified job.
interval_time	Text	IN	Yes	Expression for calculating the next time the job will be executed. ● If this parameter is null, the **interval_time** value of the specified job is not updated. ● If this parameter is not null, the system checks whether **interval_time** is a valid time type or interval type. If it is valid, the **interval_time** value of the specified job is updated. ● If the value is **"null"**, it is executed only once. After execution, the job status is changed to **'d'**, indicating that the job is no longer executed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
gaussdb=# CALL dbe_task.update(101, 'call userproc();', sysdate, 'sysdate + 1.0/1440');
 update
--------

(1 row)

gaussdb=# CALL dbe_task.update(101, 'insert into tbl_a values(sysdate);', sysdate, 'sysdate +
1.0/1440');
 update
--------

(1 row)
```

- DBE_TASK.CHANGE

  Changes user-defined attributes of a scheduled job, including the job content, next-execution time, and execution interval.

  The prototype of DBE_TASK.CHANGE is as follows:

```
DBE_TASK.CHANGE(
job             IN    INTEGER,
what            IN    TEXT       DEFAULT NULL,
next_date       IN    TIMESTAMP  DEFAULT NULL,
job_interval    IN    TEXT       DEFAULT NULL,
instance        IN    INTEGER    DEFAULT NULL,
force           IN    BOOLEAN    DEFAULT false);
```

Table 2-408 DBE_TASK.CHANGE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
job	Integer	IN	No	Specified job ID.
what	Text	IN	Yes	Name of the stored procedure or SQL statement block that is executed. If this parameter is null, the system does not update the **what** parameter for the specified job. Otherwise, the system updates the **what** parameter for the specified job.
next_date	Timestamp	IN	Yes	Next execution time. If this parameter is null, the system does not update the **next_time** parameter for the specified job. Otherwise, the system updates the **next_date** parameter for the specified job.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
job_interval	Text	IN	Yes	Expression for calculating the next time the job will be executed. ● If this parameter is empty, the **job_interval** value of a specified job is not updated. ● If this parameter is not empty, the system checks whether **job_interval** is a valid time type or interval type. If it is valid, the **job_interval** value of a specified job is updated. ● If the value is **"null"**, it is executed only once. After execution, the job status is changed to **'d'**, indicating that the job is no longer executed.
instance	Integer	IN	Yes	Used only for syntax compatibility.
force	Boolean	IN	No	Used only for syntax compatibility.

Example:

```
gaussdb=# BEGIN
gaussdb$#     DBE_TASK.CHANGE(
gaussdb$#         job => 101,
gaussdb$#         what => 'insert into t2 values (2);'
gaussdb$#     );
gaussdb$# END;
gaussdb$# /
ANONYMOUS BLOCK EXECUTE
```

- DBE_TASK.CONTENT

  Modifies the job description for a specified job.

  The prototype of DBE_TASK.CONTENT is as follows:

```
DBE_TASK.CONTENT(
id          IN     INTEGER,
content      IN TEXT);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-409 DBE_TASK.CONTENT API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	Integer	IN	No	Specified job ID.
content	Text	IN	No	Stored procedure call, SQL statement block, or program block that is executed.

NOTE

- If the value specified by the **content** parameter is one or multiple executable SQL statements, program blocks, or stored procedure calls, this function can be executed successfully; otherwise, it will fail to be executed.
- If the value specified by the **content** parameter is a simple statement such as INSERT and UPDATE, a schema name must be added in front of the table name.

Example:

```
gaussdb=# CALL dbe_task.content(101, 'call userproc();');
 content
---------

(1 row)

gaussdb=# CALL dbe_task.content(101, 'insert into tbl_a values(sysdate);');
 content
---------

(1 row)
```

- DBE_TASK.NEXT_TIME

  Modifies the next-execution time attribute of a scheduled job.

  The prototype of DBE_TASK.NEXT_TIME is as follows:

```
DBE_TASK.NEXT_TIME(
id         IN    BIGINT,
next_time  IN    TEXT);
```

Table 2-410 DBE_TASK.NEXT_TIME API parameters

Parameter	Type	Input/ Output Parameter	Can Be Null	Description
id	bigint	IN	No	Specified job ID.
next_time	Text	IN	No	Next execution time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

If the specified **next_time** value is earlier than the current date, the job is executed once immediately.

Example:

```
gaussdb=# CALL dbe_task.next_time(101, sysdate);
 next_time
-----------

(1 row)
```

- DBE_TASK.INTERVAL

  Modifies the execution interval attribute of a scheduled job.

  The prototype of DBE_TASK.INTERVAL is as follows:

```
DBE_TASK.INTERVAL(
id             IN      INTEGER,
interval_time      IN   TEXT);
```

Table 2-411 DBE_TASK.INTERVAL API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
id	Integer	IN	No	Specified job ID.
interval_time	Text	IN	Yes	Expression for calculating the next time the job will be executed. The job interval must be of a valid time type or interval type. If the value is empty or **"null"**, it is executed only once. After execution, the job status is changed to **'d'**, indicating that the job is no longer executed.

Example:

```
gaussdb=# CALL dbe_task.interval(101, 'sysdate + 1.0/1440');
 interval
----------

(1 row)

gaussdb=# CALL dbe_task.cancel(101);
 cancel
--------

(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> For a running job (that is, **job_status** is set to **'r'**), it is not allowed to use the CANCEL, UPDATE, NEXT_TIME, CONTENT, or INTERVAL API to delete or modify job parameters.

Limitations

- After a job is created by using SUMMIT/ID_SUBMIT, the job belongs to the current coordinator (that is, the job is scheduled and executed only on the current coordinator). Other coordinators do not schedule and execute the job. If the coordinator node is faulty, the job cannot be properly executed. You are advised to use the **PKG_SERVICE.SUBMIT_ON_NODES** API to specify the job execution node as CCN to ensure that the job is still available when the node is faulty. Not all coordinators can query, modify, and delete jobs created on other CNs.
- Jobs can be created, updated, and deleted only through the APIs provided by the DBE_TASK advanced package. This is because the APIs in the advanced package handle synchronization of job information across CNs, as well as the association between the primary keys of the PG_JOB and PG_JOB_PROC catalogs. If DML statements are used to add, delete, or modify the PG_JOB catalog, job information will be inconsistent across CNs, and system catalogs cannot associate changes. This will severely affect the internal management of jobs.
- Each job created by a user is bound to a CN. If the CN is faulty while a job is being executed, the job status cannot be updated in real time and will stay at **'r'**. The job status will be updated to **'s'** only after the CN is restored. When a CN is faulty before jobs are executed, all jobs on this CN cannot be scheduled or executed until the CN is restored manually, or deleted and then replaced.
- For each job, the hosting CN updates the real-time job information (including the job status, last execution start time, last execution end time, next execution start time, the number of execution failures [if any]) to the pg_job catalog, and synchronizes the information to other CNs, ensuring consistent job information between different CNs. If other CNs are faulty, job information synchronization is reattempted by the hosting CN, which increases job execution time. Although job information fails to be synchronized between CNs, job information can still be properly updated in the pg_job catalog on the hosting CN, and jobs can be executed successfully. After the faulty CN is restored, job information such as job execution time and status in its pg_job catalog may be incorrect and will be updated only after the jobs are executed again on the hosting CN.
- For each job, a thread is established to execute it. If multiple jobs are triggered concurrently as scheduled, the system will need some time to start the required threads, resulting in a latency of 0.1 ms in job execution.

2.12.2.20 DBE_UTILITY

APIs

Table 2-412 provides all APIs supported by the DBE_UTILITY package.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-412 DBE_UTILITY

API	Description
DBE_UTILITY.FORMAT_ERROR...	Outputs the call stack of an abnormal function.
DBE_UTILITY.FORMAT_ERROR...	Outputs detailed information about a function exception.
DBE_UTILITY.FORMAT_CALL_...DBE_UTILITY.FORMAT_CALL_...	Outputs the call stack of a function.
DBE_UTILITY.GET_TIME	Outputs the current time, which is used to obtain the execution duration.
DBE_UTILITY.CANONICALIZE	Canonicalizes the character string of a table name.
DBE_UTILITY.COMMA_TO_TABLE	Converts a comma-delimited string of names into a PL/SQL table of names.
DBE_UTILITY.DB_VERSION	Returns the version number and compatibility version number of the database.
DBE_UTILITY.EXEC_DDL_STATEMENT	Executes DDL statements entered by users.
DBE_UTILITY.EXPAND_SQL_TEXT_PROC	Expands the view of the SQL query.
DBE_UTILITY.GET_CPU_TIME	Returns the measured value of the current CPU processing time.
DBE_UTILITY.GET_ENDIANNESS	Obtains the endianness information of the platform where the database is located.
DBE_UTILITY.GET_HASH_VALUE	Returns the hash value of a given string.
DBE_UTILITY.GET_SQL_HASH	Outputs the hash value of a given string. This stored procedure is used when **proc_outparam_override** is not enabled.
DBE_UTILITY.IS_BIT_SET	Checks whether parameter **n** exists in **r**.
DBE_UTILITY.IS_CLUSTER_DATABASE	Determines whether the current database is running in database cluster mode.
DBE_UTILITY.NAME_RESOLVE	Parses the given object name, including synonym translation and necessary authorization checks.
DBE_UTILITY.NAME_TOKENIZE	Parses names in the format of a [. b [. c]][@ dblink].

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_UTILITY.OLD_CURRENT_SCHEMA	Returns the name of the database schema in the current user environment.
DBE_UTILITY.OLD_CURRENT_USER	Returns the name of the current user.
DBE_UTILITY.TABLE_TO_COMMA	Converts a PL/SQL table of names into a comma-delimited string of names.
DBE_UTILITY.GET_SQL_HASH_FUNC	Equivalent to DBE_UTILITY.GET_SQL_HASH. This function is used when **proc_outparam_override** is enabled.

- DBE_UTILITY.FORMAT_ERROR_BACKTRACE

  Returns the call stack where an error occurs during execution. The prototype of the DBE_UTILITY.FORMAT_ERROR_BACKTRACE function is as follows:

  ```
  DBE_UTILITY.FORMAT_ERROR_BACKTRACE()
  RETURN TEXT;
  ```

- DBE_UTILITY.FORMAT_ERROR_STACK

  Returns the detailed information about the error location when an error occurs during the execution. The prototype of the DBE_UTILITY.FORMAT_ERROR_STACK function is as follows:

  ```
  DBE_UTILITY.FORMAT_ERROR_STACK()
  RETURN TEXT;
  ```

- DBE_UTILITY.FORMAT_CALL_STACK

  Sets the call stack of the output function. The prototype of the DBE_UTILITY.FORMAT_CALL_STACK function is as follows:

  ```
  DBE_UTILITY.FORMAT_CALL_STACK()
  RETURN TEXT;
  ```

- DBE_UTILITY.GET_TIME

  Sets the output time, which is usually used for calculating a time difference. A separate return value is meaningless. The prototype of the DBE_UTILITY.GET_TIME function is as follows:

  ```
  DBE_UTILITY.GET_TIME()
  RETURN BIGINT;
  ```

- DBE_UTILITY.CANONICALIZE

  Canonicalizes the character string of a table name. The procedure handles a single reserved word or keyword, and removes white spaces for a single identifier so that "table" becomes TABLE. The prototype of the DBE_UTILITY.CANONICALIZE function is as follows:

  ```
  DBE_UTILITY.CANONICALIZE(
     name       IN   VARCHAR2,
     canon_name OUT  VARCHAR2,
     canon_len  IN   BINARY_INTEGER DEFAULT 1024
  );
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-413 DBE_UTILITY.CANONICALIZE API parameters

Parameter	Type	Input / Output Parameter	Can Be Null	Description
name	VARCHAR2	IN	No	Character string to be canonicalized.
canon_name	VARCHAR2	OUT	Yes	Canonicalized character string.
canon_len	BINARY_INTEGER	IN	Yes	Length of the string to be canonicalized. The default value is **1024** (in bytes). If the value of this parameter is less than the actual length (in bytes) of the character string to be canonicalized, the character string is truncated by byte.

- DBE_UTILITY.COMMA_TO_TABLE

  Converts a comma-delimited string of names into a PL/SQL table of names. The prototype of the DBE_UTILITY.COMMA_TO_TABLE function is as follows:

```
DBE_UTILITY.COMMA_TO_TABLE (
   list   IN  VARCHAR2,
   tablen OUT BINARY_INTEGER,
   tab    OUT VARCHAR2[]
);
```

Table 2-414 DBE_UTILITY.COMMA_TO_TABLE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
list	VARCHAR2	IN	No	A comma-delimited string of names.
tablen	BINARY_INTEGER	OUT	Yes	Number of names in the table.
tab	VARCHAR2	OUT	Yes	Table which contains the string of names.

- DBE_UTILITY.DB_VERSION

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Returns the version number and compatibility version number of the database. The prototype of the DBE_UTILITY.DB_VERSION function is as follows:

```
DBE_UTILITY.DB_VERSION (
   version OUT VARCHAR2
);
```

Table 2-415 DBE_UTILITY.DB_VERSION API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
version	VARCHAR2	OUT	No	Output parameter, which indicates the internal database software version. The value is a character string.

- DBE_UTILITY.EXEC_DDL_STATEMENT

  NOTE

  When the EXECUTE DIRECT ON syntax is used to execute the DBE_UTILITY.EXEC_DDL_STATEMENT function, if the GUC parameter **enable_udf_execute_direct** is set to **on**, the functions can be executed; otherwise, the functions cannot be executed.

  Executes DDL statements entered by users. The prototype of the DBE_UTILITY.EXEC_DDL_STATEMENT function is as follows:

  ```
  DBE_UTILITY.EXEC_DDL_STATEMENT (
     parse_string IN TEXT
  );
  ```

  Table 2-416 DBE_UTILITY.EXEC_DDL_STATEMENT API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
parse_string	TEXT	IN	Yes	DDL statements to be executed.

- DBE_UTILITY.EXPAND_SQL_TEXT_PROC

  Expands the queried view recursively till the table object layer. The prototype of the DBE_UTILITY.EXPAND_SQL_TEXT_PROC function is as follows:

  ```
  DBE_UTILITY.EXPAND_SQL_TEXT_PROC (
     input_sql_text  IN  CLOB,
     output_sql_text OUT CLOB
  );
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-417 DBE_UTILITY.EXPAND_SQL_TEXT_PROC API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
input_sql_text	CLOB	IN	No	Input SQL text.
output_sql_text	CLOB	OUT	No	Output SQL text of the expanded view.

NOTE

In the **input_sql_text** parameter entered by a user, a schema prefix must be added to the object in the SQL statement. Otherwise, the function reports an error indicating that no object is found. If **behavior_compat_options** is set to **bind_procedure_searchpath**, you do not need to forcibly specify the schema prefix.

- DBE_UTILITY.GET_CPU_TIME

  Returns the measured value of the current CPU processing time, in hundredths of a second. The prototype of the DBE_UTILITY.GET_CPU_TIME function is as follows:

```
DBE_UTILITY.GET_CPU_TIME()
RETURN BIGINT;
```

- DBE_UTILITY.GET_ENDIANNESS

  Obtains the endianness information of the platform where the database is located. DBE_UTILITY. The prototype of the GET_ENDIANNESS function is as follows:

```
DBE_UTILITY.GET_ENDIANNESS
RETURN INTEGER;
```

- DBE_UTILITY.GET_HASH_VALUE

  Returns the hash value of a given string. The prototype of the DBE_UTILITY.GET_HASH_VALUE function is as follows:

```
DBE_UTILITY.GET_HASH_VALUE(
   name      IN VARCHAR2(n),
   base      IN INTEGER,
   hash_size IN INTEGER)
RETURN INTEGER;
```

Table 2-418 DBE_UTILITY.GET_HASH_VALUE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
name	VARCHAR2	IN	No	Character string to be hashed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
base	INTEGER	IN	No	Start value of the returned hash value.
hash_size	INTEGER	IN	No	Size of the hash table to which the hash is mapped.

- DBE_UTILITY.GET_SQL_HASH

  Outputs the hash value of a given character string using the MD5 algorithm. The prototype of the DBE_UTILITY.GET_SQL_HASH function is as follows:

```
DBE_UTILITY.GET_SQL_HASH(
   name       IN  VARCHAR2,
   hash       OUT RAW,
   last4bytes OUT BIGINT
)RETURN BIGINT;
```

Table 2-419 DBE_UTILITY.GET_SQL_HASH API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
name	VARCHAR2	IN	No	Character string to be hashed.
hash	RAW	OUT	No	Complete hexadecimal MD5 hash value.
last4bytes	BIGINT	OUT	No	Last four bytes of the MD5 hash value, which is displayed as an unsigned integer.

NOTE

After setting **behavior_compat_options** to a value other than **proc_outparam_override**, call the DBE_UTILITY.GET_SQL_HASH function, instead of DBE_UTILITY.GET_SQL_HASH_FUNC; otherwise, the value assignment will fail.

- DBE_UTILITY.IS_BIT_SET

  Checks whether parameter **n** exists in **r**. DBE_UTILITY. The prototype of the IS_BIT_SET function is as follows:

```
DBE_UTILITY.IS_BIT_SET (
   r IN RAW,
   n IN INTEGER)
RETURN INTEGER;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-420 DBE_UTILITY.IS_BIT_SET API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
r	RAW	IN	No	4 bytes plus the actual hexadecimal string.
n	INTEGER	IN	No	Determines whether the value exists in the binary system.

- DBE_UTILITY.IS_CLUSTER_DATABASE

  Determines whether the current database is running in database cluster mode. The prototype of the DBE_UTILITY.IS_CLUSTER_DATABASE function is as follows:

  ```
  DBE_UTILITY.IS_CLUSTER_DATABASE
  RETURN BOOLEAN;
  ```

- DBE_UTILITY.NAME_RESOLVE

  Parses the given object name, including synonym translation and necessary authorization checks. The prototype of the DBE_UTILITY.NAME_RESOLVE function is as follows:

  ```
  DBE_UTILITY.NAME_RESOLVE (
     name          IN   VARCHAR2,
     context       IN   INTEGER,
     schema          OUT VARCHAR2,
     part1         OUT VARCHAR2,
     part2         OUT VARCHAR2,
     dblink        OUT VARCHAR2,
     part1_type    OUT INTEGER,
     object_number OUT OID
  );
  ```

Table 2-421 DBE_UTILITY.NAME_RESOLVE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
name	VARCHAR2	IN	No	Name of the object to be parsed. The structure is [[a.]b.]c[@d].

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Input/Output Parameter	Can Be Null	Description
context	INTEGER	IN	No	Type of the object to be parsed: ● **0**: table ● **1**: function, procedure, and package ● **2**: sequence ● **3**: trigger ● **4**: Java source ● **5**: Java resource ● **6**: Java class ● **7**: type ● **8**: Java shared data ● **9**: index
schema	VARCHAR2	OUT	No	Schema of an object.
part1	VARCHAR2	OUT	No	First part of the name. The type of this column is specified by **part1_type**.
part2	VARCHAR2	OUT	Yes	If this column is not null, the value is the subprogram name.
dblink	VARCHAR2	OUT	Yes	Database link.
part1_type	INTEGER	OUT	No	Part 1 types: ● **5**: synonym ● **7**: procedure (top level) ● **8**: function (top level) ● **9**: package
object_number	OID	OUT	No	Object ID. In database ORA, **object_number** is of the numeric type, indicating the object ID. In GaussDB, **object_number** is of the OID type and does not support implicit conversion from a number to an OID.

- DBE_UTILITY.NAME_TOKENIZE

  Parses names in the format of a [. b [. c]][@ dblink]. If a name contains double quotation marks, remove the double quotation marks; otherwise, the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

name will be in uppercase. The prototype of the DBE_UTILITY.NAME_TOKENIZE function is as follows:

```
DBE_UTILITY.NAME_TOKENIZE (
name    IN  VARCHAR2,
a       OUT VARCHAR2,
b       OUT VARCHAR2,
c       OUT VARCHAR2,
dblink  OUT VARCHAR2,
nextpos OUT INTEGER
);
```

Table 2-422 DBE_UTILITY.NAME_TOKENIZE API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
name	VARCHAR2	IN	No	Name, consisting of SQL identifiers (for example, **scott.foo@dblink**).
a	VARCHAR2	OUT	No	First token of the name.
b	VARCHAR2	OUT	Yes	Second token of the name.
c	VARCHAR2	OUT	Yes	Third token of the name.
dblink	VARCHAR2	OUT	Yes	Database link.
nextpos	INTEGER	OUT	No	Next position of a parsed character string.

- DBE_UTILITY.OLD_CURRENT_SCHEMA

  Returns the name of the database schema in the current user environment. The prototype of the DBE_UTILITY.OLD_CURRENT_SCHEMA function is as follows:

  ```
  DBE_UTILITY.OLD_CURRENT_SCHEMA()
  RETURN VARCHAR;
  ```

- DBE_UTILITY.OLD_CURRENT_USER

  Returns the name of the current user. The prototype of the DBE_UTILITY.OLD_CURRENT_USER function is as follows:

  ```
  DBE_UTILITY.OLD_CURRENT_USER()
  RETURN TEXT;
  ```

- DBE_UTILITY.TABLE_TO_COMMA

  Converts a PL/SQL table of names into a comma-delimited string of names. The prototype of the DBE_UTILITY.TABLE_TO_COMMA function is as follows:

  ```
  DBE_UTILITY.TABLE_TO_COMMA (
     tab    IN  VARCHAR2[],
     tablen OUT BINARY_INTEGER,
     list   OUT VARCHAR2
  );
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-423 DBE_UTILITY.TABLE_TO_COMMA API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
tab	VARCHAR2[]	IN	No	Structure table array that contains table names.
tablen	BINARY_INTEGER	OUT	No	Number of tables in a structure table.
list	VARCHAR2	OUT	No	A comma-delimited string of names.

- DBE_UTILITY.GET_SQL_HASH_FUNC

  Uses the MD5 algorithm to output the hash value of a given character string. The prototype of the DBE_UTILITY.GET_SQL_HASH_FUNC function is:

```
DBE_UTILITY.GET_SQL_HASH_FUNC(
   name       IN  VARCHAR2,
   hash       OUT RAW,
   last4bytes OUT BIGINT
);
```

Table 2-424 DBE_UTILITY.GET_SQL_HASH_FUNC API parameters

Parameter	Type	Input/Output Parameter	Can Be Null	Description
name	VARCHAR2	IN	No	Character string to be hashed.
hash	RAW	OUT	No	Complete hexadecimal MD5 hash value.
last4bytes	BIGINT	OUT	No	Last four bytes of the MD5 hash value, which is displayed as an unsigned integer.

NOTE

After setting **behavior_compat_options** to **'proc_outparam_override'**, call the DBE_UTILITY.GET_SQL_HASH_FUNC function. If you call the DBE_UTILITY.GET_SQL_HASH function, a parameter mismatch error is reported.

Examples

```
-- Example 1
create or replace procedure print_err() as
DECLARE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    a bool;
BEGIN
    a := not_exist;
exception
    when others then
        dbe_output.print_line('err_stack: ' || DBE_UTILITY.FORMAT_ERROR_STACK());
END;
/
CREATE PROCEDURE

call print_err();
-- Expected result:
err_stack: 50360452: column "not_exist" does not exist
 print_err
-----------

(1 row)

-- Clean up the environment.
drop procedure print_err;
DROP PROCEDURE

-- Example 2
create or replace procedure print_err() as
DECLARE
    a bool;
BEGIN
    a := not_exist;
exception
    when others then
        dbe_output.print_line('backtrace: ' || DBE_UTILITY.FORMAT_ERROR_BACKTRACE());
END;
/
CREATE PROCEDURE

call print_err();
-- Expected result:
backtrace: 50360452: PL/pgSQL function print_err() line 5 at assignment

 print_err
-----------

(1 row)

-- Clean up the environment.
drop procedure print_err;
DROP PROCEDURE

-- Example 3
create or replace procedure print_err() as
DECLARE
    a bool;
BEGIN
    a := not_exist;
exception
    when others then
        dbe_output.print_line('call_stack: ');
        dbe_output.print_line(DBE_UTILITY.FORMAT_CALL_STACK());
END;
/
CREATE PROCEDURE

call print_err();
-- Expected result:
call_stack:
      3  dbe_utility.format_call_stack()
      9  print_err()

 print_err
```

```
-----------

(1 row)

-- Clean up the environment.
drop procedure print_err;
DROP PROCEDURE

-- Example 4
CREATE OR REPLACE PROCEDURE test_get_time1()
AS
declare
    start_time  bigint;
    end_time  bigint;
BEGIN
    start_time:= dbe_utility.get_time ();
    pg_sleep(1);
    end_time:=dbe_utility.get_time ();
    dbe_output.print_line(end_time - start_time);
END;
/
CREATE PROCEDURE

call test_get_time1();
-- Expected result:
101
 test_get_time1
----------------

(1 row)

-- Clean up the environment.
drop PROCEDURE test_get_time1;
DROP PROCEDURE

-- Example 5
-- Canonicalize the character string of a table name.
declare
    cname varchar2(50);
begin
    dbe_utility.canonicalize('seg1', cname, 50);
    dbe_output.put_line(cname);
end;
/
-- Expected result:
SEG1
ANONYMOUS BLOCK EXECUTE

-- Example 6
-- Convert the input character string into an array of table names.
DECLARE
 tab_list VARCHAR2(100) := 't1,t2';
 len BINARY_INTEGER;
 tab varchar2[];
BEGIN
 dbe_output.put_line('table list is: ' || tab_list);
 dbe_utility.comma_to_table(tab_list, len, tab);
END;
/
-- Expected result:
table list is: t1,t2
ANONYMOUS BLOCK EXECUTE

-- Example 7
-- Check the version number and compatibility version number of the database.
declare
    v_version varchar2;
begin
    dbe_utility.db_version(v_version);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    v_version:=left(v_version, 8);
    dbe_output.print_line('version:' || v_version);
end;
/
-- Expected result:
version:gaussdb
ANONYMOUS BLOCK EXECUTE

-- Example 8
-- Check the measured value of the current CPU processing time.
DECLARE
    cputime NUMBER;
BEGIN
    cputime := dbe_utility.get_cpu_time();
    dbe_output.put_line('cpu time:' || cputime);
END;
/
-- The expected result is as follows (the value is not fixed):
cpu time: 70179
ANONYMOUS BLOCK EXECUTE

-- Example 9
-- Obtain the endianness information of the platform where the database is located.
BEGIN
    dbe_output.PUT_LINE(dbe_utility.GET_ENDIANNESS());
END;
/
-- Expected result:
2
ANONYMOUS BLOCK EXECUTE

-- Example 10
SET system_function_version TO DEFAULT;
SET
-- Obtain the hash value of a given string.
DECLARE
    result NUMBER(28);
BEGIN
    result := dbe_utility.get_hash_value('hello',10,10);
    dbe_output.put_line(result);
END;
/
-- Expected result:
15
ANONYMOUS BLOCK EXECUTE

-- Example 11
-- Check whether the current database is in cluster mode.
DECLARE
    is_cluster BOOLEAN;
BEGIN
    is_cluster := dbe_utility.IS_CLUSTER_DATABASE();
    dbe_output.put_line('CLUSTER DATABASE: ' || CASE WHEN is_cluster THEN 'TRUE' ELSE 'FALSE' END);
END;
/
-- Expected result:
CLUSTER DATABASE: TRUE
ANONYMOUS BLOCK EXECUTE

-- Example 12
-- Obtain the name of the database schema in the current user environment.
DECLARE
    schm varchar2(100);
BEGIN
    schm := dbe_utility.old_current_schema();
    dbe_output.put_line('current schema: ' || schm);
END;
/
-- The expected result is as follows (the result is the schema name of the current database, which is not
```

```
fixed):
current schema: public
ANONYMOUS BLOCK EXECUTE

-- Example 13
-- Obtain the current username.
select dbe_utility.old_current_user() from sys_dummy;
-- The expected result is as follows (the result is the username of the current database, which is not fixed):
 old_current_user
------------------
 test
(1 row)

-- Example 14
DECLARE
    ddl_str VARCHAR2(255);
BEGIN
    dbe_output.print_line('start to test exec_ddl_statement create table.');
    ddl_str := 'CREATE TABLE test_ddl (COL1 INT)';
    dbe_utility.exec_ddl_statement(ddl_str);
END;
/
-- Expected result:
start to test exec_ddl_statement create table.
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'col1' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CONTEXT:  SQL statement "CREATE TABLE test_ddl (COL1 INT)"
SQL statement "CALL dbe_utility.exec_ddl_statement(ddl_str)"
PL/pgSQL function inline_code_block line 5 at PERFORM
ANONYMOUS BLOCK EXECUTE

select * from test_ddl;
-- Expected result:
 col1
------
(0 rows)

-- Clean up the environment.
drop table test_ddl;
DROP TABLE

-- Example 15
create table t1 (c1 int primary key, c2 int);
insert into t1 values(1,1),(2,1),(3,2),(4,2),(5,3),(6,3);
create view v1 as select * from t1 where c1 > 1;
create view v2 as select c1 from v1 where c2 > 1;
create view v3 as select * from v2 where c1 > 2;

declare
    in_sql clob := 'select * from public.v3';
    out_sql clob;
begin
    dbe_output.print_line('start to test expand_sql_text_proc v3 expend sql text.');
    dbe_utility.expand_sql_text_proc(in_sql, out_sql);
    dbe_output.print_line(out_sql);
end;
/
-- Expected result:
start to test expand_sql_text_proc v3 expend sql text.
SELECT c1 FROM (SELECT v2.c1 FROM (SELECT v1.c1 FROM (SELECT t1.c1, t1.c2 FROM public.t1 WHERE
t1.c1 > 1) v1 WHERE v1.c2 > 1) v2 WHERE v2.c1 > 2) v3
ANONYMOUS BLOCK EXECUTE

-- Clean up the environment.
drop table t1 cascade;
-- Expected result:
NOTICE:  drop cascades to 3 other objects
DETAIL:  drop cascades to view v1
drop cascades to view v2
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
drop cascades to view v3


-- Example 16
declare
    name varchar2;
    hash raw;
    last4bytes bigint;
BEGIN
    name := '';
    -- return correctly (D41D8CD98F00B204E9800998ECF8427E, 2118318316)
    dbe_utility.get_sql_hash(name,hash,last4bytes);
    raise notice '%',hash;
    raise notice '%',last4bytes;
END;
/
-- Expected result:
NOTICE:  D41D8CD98F00B204E9800998ECF8427E
NOTICE:  2118318316
ANONYMOUS BLOCK EXECUTE

-- Example 17
declare
    bitchar raw(8);
begin
    bitchar := '2111111f';
    dbe_output.print('test '|| bitchar ||' bit is_bit_set value from 1 to 32 bit: ');
    for i in reverse 32 .. 1 loop
        dbe_output.print(dbe_utility.is_bit_set(bitchar, i));
    end loop;
    dbe_output.print_line('.');
end;
/
-- Expected result:
test 2111111F bit is_bit_set value from 1 to 32 bit: 00100001000100010001000100011111.
ANONYMOUS BLOCK EXECUTE

-- Example 18
create or REPLACE PROCEDURE p_test_pk ( -- for print result
    name in varchar2,
    type in integer
)
as
    schema varchar2;
    part1 varchar2;
    part2 varchar2;
    dblink varchar2;
    part1_type integer;
    object_number integer;
begin
    dbe_utility.name_resolve(name,type,schema,part1,part2,dblink,part1_type,object_number);
    raise notice 'schema: % -- part1: % -- part2: % -- dblink: % -- part1_type: %',
schema,part1,part2,dblink,part1_type;
end;
/
CREATE PROCEDURE

declare begin p_test_pk('a.b.c@aa',3); end;
/
-- Expected result:
NOTICE:  schema: a -- part1: b -- part2: c -- dblink: aa -- part1_type: 0
CONTEXT:  SQL statement "CALL p_test_pk('a.b.c@aa',3)"
PL/pgSQL function inline_code_block line 1 at PERFORM
ANONYMOUS BLOCK EXECUTE

-- Example 19
DECLARE
    name varchar;
    a    varchar;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    b    varchar;
    c    varchar;
    dblink varchar;
    nextpos INTEGER;
BEGIN
    name := 'Me.w#sdfsdf.CD';
    DBE_UTILITY.NAME_TOKENIZE(name, a, b, c, dblink, nextpos);
    RAISE INFO E'dbe_utility.name_tokenize parse error: name:%\na:%\nb:%\nc:%\ndblink:%',
    name, a, b, c, dblink;
    IF nextpos <> OCTET_LENGTH(name) THEN
        RAISE INFO E'dbe_utility.name_tokenize length error: name:%\nlength of name:%\nnextpos:%',
        name, OCTET_LENGTH(name), nextpos;
    END IF;
END;
/
-- Expected result:
INFO:  dbe_utility.name_tokenize parse error: name:Me.w#sdfsdf.CD
a: Me
b:W#SDFSDF
c:CD
dblink:<NULL>
ANONYMOUS BLOCK EXECUTE

-- Example 20
DECLARE
list varchar2(50) := 'aabb,ccdd,eeff,gghh';
len_list integer;
tab varchar2[];
get_list varchar2(50);
len_tab integer;
BEGIN
dbe_output.print_line(''Parameter list:' || list);
dbe_utility.comma_to_table(list,len_list,tab);
dbe_output.print_line('Parameter length:' || len_list);
FOR i IN 1 .. len_list LOOP
    dbe_output.print_line('List name' || i || ' : ' || tab(i));
END LOOP;
dbe_output.print_line('Call table_to_comma:');
dbe_utility.table_to_comma(tab,len_tab,get_list );
dbe_output.print_line('Output:' || get_list );
dbe_output.print_line('Array length:' || len_tab);
END;
/
-- Expected result:
Parameter list: aabb,ccdd,eeff,gghh
Parameter length: 4
List name 1: aabb
List name 2: ccdd
List name 3: eeff
List name 4: gghh
Call table_to_comma:
Output: aabb,ccdd,eeff,gghh
Array length: 4
ANONYMOUS BLOCK EXECUTE

-- Example 21
declare
    name varchar2;
    hash raw;
    last4bytes bigint;
BEGIN
    name := 'hello world';
    -- return correctly(5EB63BBBE01EEED093CB22BB8F5ACDC3, 3285015183)
    dbe_utility.get_sql_hash_func(name,hash,last4bytes);
    raise notice '%',hash;
    raise notice '%',last4bytes;
END;
/
-- Expected result:
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
NOTICE:  3285015183
NOTICE:  <NULL>
ANONYMOUS BLOCK EXECUTE
```

2.12.2.21 DBE_XMLDOM

APIs

The advanced package DBE_XMLDOM is used to access XMLType objects and implement Document Object Model (DOM), which is an API used to access HTML and XML documents. For details about all types supported by the advanced function package DBE_XMLDOM, see **Table 2-425**. For details about all APIs supported by DBE_XMLDOM, see **Table 2-426**.

NOTE

When the DBE_XMLDOM advanced package is used in the database whose character set is set to **SQL_ASCII**, an error is reported if characters beyond the ASCII range are input.

Table 2-425 DBE_XMLDOM data types

Type	Description
DOMATTR	Implements the DOMAttributes API.
DOMDOCUMENT	Implements the DOMDocument API.
DOMELEMENT	Implements the DOMElement API.
DOMNAMEDNODEMAP	Implements the DOMNamedNodeMap API.
DOMNODELIST	Implements the DOMNodeList API.
DOMNODE	Implements the DOMNode API.
DOMTEXT	Implements the DOMText API.

Table 2-426 DBE_XMLDOM API parameters

API	Description
DBE_XMLDOM.APPENDCHILD	Adds the newchild node to the end of the parent(n) node and returns the newly added node.
DBE_XMLDOM.CREATEELEMENT	Creates a DOMElement object with the specified name.
DBE_XMLDOM.CREATETEXTNODE	Creates a DOMText node.
DBE_XMLDOM.FREEDOCUMENT	Frees resources related to DOMDocument nodes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_XMLDOM.FREEELEMENT	Frees resources related to DOMElement nodes.
DBE_XMLDOM.FREENODE	Frees resources related to DOMNode nodes.
DBE_XMLDOM.FREENODELIST	Frees resources related to DOMNodeList nodes.
DBE_XMLDOM.GETATTRIBUTE	Returns the attribute values of a DOMElement object by name.
DBE_XMLDOM.GETATTRIBUTES	Returns the attribute values of a DOMNode node as a map.
DBE_XMLDOM.GETCHILDNODES	Converts several subnodes under a node into a node list.
DBE_XMLDOM.GETCHILDRENBYTAGNAME	Returns the subnodes of a DOMElement node by name.
DBE_XMLDOM.GETDOCUMENTELEMENT	Returns the first subnode of the specified document.
DBE_XMLDOM.GETFIRSTCHILD	Returns the first subnode.
DBE_XMLDOM.GETLASTCHILD	Returns the last subnode.
DBE_XMLDOM.GETLENGTH	Obtains the number of subnodes under a specified node.
DBE_XMLDOM.GETLOCALNAME	Returns the local name of a node.
DBE_XMLDOM.GETNAMEDITEM	Returns the node specified by name.
DBE_XMLDOM.GETNEXTSIBLING	Returns the next node of the specified node.
DBE_XMLDOM.GETNODENAME	Returns the name of a node.
DBE_XMLDOM.GETNODETYPE	Returns the type of a node.
DBE_XMLDOM.GETNODEVALUE	Obtains the value of a node, depending on its type.
DBE_XMLDOM.GETPARENTNODE	Returns the parent node of a node.
DBE_XMLDOM.GETTAGNAME	Returns the tag name of the specified DOMElement node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_XMLDOM.HASCHILDNODES	Checks whether the DOMNode object has any subnode.
DBE_XMLDOM.IMPORTNODE	Copies a node and specifies the document to which the node belongs.
DBE_XMLDOM.ISNULL	Checks whether a node is null.
DBE_XMLDOM.ITEM	Returns the item corresponding to the index parameter in the mapping.
DBE_XMLDOM.MAKEELEMENT	Converts a DOMNode object to the DOMElement type.
DBE_XMLDOM.MAKENODE	Forcibly converts a node to the DOMNode type.
DBE_XMLDOM.NEWDOMDOCUMENT	Returns a new DOMDocument object.
DBE_XMLDOM.SETATTRIBUTE	Sets the value of the DOMElement attribute by name.
DBE_XMLDOM.SETCHARSET	Sets the character set for a DOMDocument object.
DBE_XMLDOM.SETDOCTYPE	Sets the external DTD of a DOMDocument object.
DBE_XMLDOM.SETNODEVALUE	Sets the value of a node in the DOMNode object.
DBE_XMLDOM.WRITETOBUFFER	Writes an XML node to a specified buffer.
DBE_XMLDOM.WRITETOCLOB	Writes an XML node to a specified CLOB.
DBE_XMLDOM.WRITETOFILE	Writes an XML node to a specified file.
DBE_XMLDOM.GETSESSIONTREENUM	Displays the number of DOM trees of all types in the current session.
DBE_XMLDOM.GETDOCTREESINFO	Displays statistics such as the memory usage and number of nodes of the DOM tree of the document type.
DBE_XMLDOM.GETDETAILDOCTREEINFO	Displays the number of nodes of each type for a specific document variable.
DBE_XMLDOM.GETELEMENTSBYTAGNAM	Returns the list of DOMNodeList nodes that matches TAGNAME.

- DBE_XMLDOM.APPENDCHILD

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Adds the newchild node to the end of the parent(n) node and returns the newly added node. The prototype of the DBE_XMLDOM.APPENDCHILD function is as follows:

```
DBE_XMLDOM.APPENDCHILD(
  n IN DOMNode,
  newchild IN DOMNode)
RETURN DOMNODE;
```

Table 2-427 DBE_XMLDOM.APPENDCHILD API parameters

Parameter	Description
n	Node to be added.
newchild	New node added.

NOTE

- The error message "operation not support" is displayed for the APPEND ATTR node under the DOCUMENT node. Database ORA does not report an error in this scenario, but the mounting fails.
- The error message "operation not support" is displayed for the APPEND ATTR node under the ATTR node. Database ORA does not report an error in this scenario, but the mounting fails.
- When multiple child nodes of the ATTR type are added to a parent node, the child nodes with the same key value cannot exist under the same parent node.

Example:

```
-- Add a DOMNode node to a specified DOC tree and use DBE_XMLDOM.HASCHILDNODES() to check whether the subnode is successfully added.
DECLARE
   doc DBE_XMLDOM.DOMDocument;
   doc1 DBE_XMLDOM.DOMDocument;
   root DBE_XMLDOM.DOMElement;
   rootnode DBE_XMLDOM.DOMNode;
   child1 DBE_XMLDOM.DOMElement;
   child2 DBE_XMLDOM.DOMElement;
   attr DBE_XMLDOM.DOMAttr;
   text DBE_XMLDOM.DOMTEXT;
   node DBE_XMLDOM.DOMNode;
   child1_node DBE_XMLDOM.DOMNode;
   attr_node DBE_XMLDOM.DOMNode;
   parent DBE_XMLDOM.DOMNode;
   buf varchar2(1000);
BEGIN
   doc := DBE_XMLDOM.newDOMDocument();
   root := DBE_XMLDOM.createElement(doc, 'root');
   rootnode := DBE_xmldom.makeNode(root);
   node := DBE_XMLDOM.appendChild(DBE_xmldom.makeNode(doc), rootnode);
   child1 := DBE_XMLDOM.createElement(doc, 'child1');
   child1_node := DBE_XMLDOM.makeNode(child1);
   node := DBE_XMLDOM.appendChild(rootnode, child1_node);
   attr := DBE_XMLDOM.createAttribute(doc, 'abc');
   attr_node := DBE_XMLDOM.makeNode(attr);
   node := DBE_XMLDOM.appendChild(child1_node, attr_node);
   IF DBE_XMLDOM.HASCHILDNODES(child1_node) THEN
      DBE_OUTPUT.print_line('HAS CHILD NODES');
   ELSE
      DBE_OUTPUT.print_line('NOT HAS CHILD NODES ');
   END IF;
   parent := DBE_XMLDOM.GETPARENTNODE(attr_node);
   buf := DBE_XMLDOM.GETNODENAME(parent);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    DBE_OUTPUT.print_line(buf);
END;
/
-- Expected result:
NOT HAS CHILD NODES
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.CREATEELEMENT

  Returns the DOMElement object with the specified name. The prototype of the DBE_XMLDOM.CREATEELEMENT function is as follows:

```
DBE_XMLDOM.CREATEELEMENT(
  doc       IN      DOMDOCUMENT,
  tagName   IN      VARCHAR2)
 RETURN DOMELEMENT;
```

  Returns the DOMElement object with the specified name and namespace. The prototype of the DBE_XMLDOM.CREATEELEMENT function is as follows:

```
DBE_XMLDOM.CREATEELEMENT(
  doc       IN      DOMDOCUMENT,
  tagName   IN      VARCHAR2,
  ns        IN      VARCHAR2)
 RETURN DOMELEMENT;
```

Table 2-428 DBE_XMLDOM.CREATEELEMENT API parameters

Parameter	Description
doc	Specified DOMDocument object.
tagName	Name of the new DOMElement object.
ns	Namespace.

NOTE

- When the **tagName** parameter is set to null or an empty character string, the exception "NULL or invalid TagName argument specified" is thrown.
- The default maximum length of **tagName** and **ns** is 32767. If the length exceeds 32767, an exception is thrown.

Example:

```
-- 1. Create a DOMElement object with the specified name.
DECLARE
  doc dbe_xmldom.domdocument;
  attr DBE_XMLDOM.DOMATTR;
  elem DBE_XMLDOM.DOMELEMENT;
  ans DBE_XMLDOM.DOMATTR;
  buf varchar2(1010);
BEGIN
   doc := dbe_xmldom.newdomdocument('<?xml version="1.0" encoding="UTF-8"?>
      <computer size="ITX"><cpu>Ryzen 9 3950X</cpu>
      <ram>32GBx2 DDR4 3200MHz</ram>
      <motherboard>ROG X570i</motherboard>
      <gpu>RTX2070 Super</gpu>
      <ssd>1TB NVMe Toshiba + 2TB NVMe WD Black</ssd>
      <hdd>12TB WD Digital</hdd>
      <psu>CORSAIR SF750</psu>
      <case>LIANLI TU150</case>
      </computer>');
   elem := dbe_xmldom.createelement(doc,'elem');
   DBE_XMLDOM.WRITETOBUFFER(dbe_xmldom.makenode(elem), buf);
   DBE_OUTPUT.print_line(buf);
END;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
/
-- Expected result:
ANONYMOUS BLOCK EXECUTE

-- 2. Create a DOMElement object with the specified name and namespace.
DECLARE
  doc dbe_xmldom.domdocument;
  attr DBE_XMLDOM.DOMATTR;
  elem DBE_XMLDOM.DOMELEMENT;
  ans DBE_XMLDOM.DOMNODE;
  buf varchar2(1010);
  list DBE_XMLDOM.DOMNODELIST;
  node DBE_XMLDOM.DOMNODE;
BEGIN
   doc := dbe_xmldom.newdomdocument('<h:data xmlns:h="http://www.w3.org/TR/html4/">
       <h:da1 len="10">test namespace</h:da1><h:da1>bbbbbbbbbb</h:da1></h:data>');
   elem := dbe_xmldom.createelement(doc,'elem','http://www.w3.org/TR/html5/');
   ans := DBE_XMLDOM.APPENDCHILD(dbe_xmldom.makenode(doc), dbe_xmldom.makenode(elem));
   DBE_XMLDOM.WRITETOBUFFER(doc, buf);
   DBE_OUTPUT.print_line(buf);
END;
/
-- Expected result:
<?xml version="1.0" encoding="UTF-8"?>
<h:data xmlns:h="http://www.w3.org/TR/html4/">
 <h:da1 len="10">test namespace</h:da1>
 <h:da1>bbbbbbbbbb</h:da1>
</h:data>
<elem xmlns="http://www.w3.org/TR/html5/"/>

ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.CREATETEXTNODE

  Creates and returns a DOMText object. The prototype of the DBE_XMLDOM.CREATETEXTNOD function is as follows:

```
DBE_XMLDOM.CREATETEXTNODE(
  doc IN DOMDocument,
  data IN VARCHAR2)
RETURN DOMTEXT;
```

Table 2-429 DBE_XMLDOM.CREATETEXTNODE API parameters

Parameter	Description
doc	Specified DOMDocument object.
data	Content of the DOMText node.

NOTE

- You can enter an empty string or null value for **data**.
- The default maximum length of **data** is 32767. If the length exceeds 32767, an exception is thrown.

Example:

```
-- Add a DOMText node to the DOC tree and print the DOC tree to the buffer.
DECLARE
  doc DBE_XMLDOM.DOMDOCUMENT;
  doctext DBE_XMLDOM.DOMTEXT;
  node DBE_XMLDOM.DOMNODE;
  buffer varchar2(1010);
BEGIN
  doc := dbe_xmldom.newdomdocument('<?xml version="1.0"?>
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)>
    <!ELEMENT to (#PCDATA)>
    <!ELEMENT from (#PCDATA)>
    <!ELEMENT heading (#PCDATA)>
    <!ELEMENT body (#PCDATA)>]>
    <note>
      <to>Chinese</to>
      <from>Jani</from>
      <heading>Reminder</heading>
      <body>Don''t forget me this weekend!</body>
    </note>');
  doctext := DBE_XMLDOM.CREATETEXTNODE(doc, 'there is nothing');
  node := DBE_XMLDOM.MAKENODE(doctext);
  dbe_xmldom.writetobuffer(node, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
END;
/
-- Expected result:
buffer:
there is nothing
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.FREEDOCUMENT

  Frees a DOMDocument node. The prototype of the DBE_XMLDOM.FREEDOCUMENT function is as follows:

```
DBE_XMLDOM.FREEDOCUMENT(
  doc     IN     DOMDOCUMENT);
```

Table 2-430 DBE_XMLDOM.FREEDOCUMENT API parameters

Parameter	Description
doc	Specified DOMDocument node.

Example:

```
-- Free the entire DOC tree after the DOMNode node is added to the DOC tree.
DECLARE
  doc dbe_xmldom.domdocument;
  elem dbe_xmldom.domelement;
  doc_node dbe_xmldom.DOMNODE;
  root_elmt dbe_xmldom.DOMELEMENT;
  root_node dbe_xmldom.DOMNODE;
  value  varchar(1000);
BEGIN
  doc := dbe_xmldom.newdomdocument();
  doc_node := dbe_xmldom.MAKENODE(doc);
  root_elmt := dbe_xmldom.CREATEELEMENT(doc,'staff');
  root_node:=dbe_xmldom.APPENDCHILD(doc_node, dbe_xmldom.MAKENODE(root_elmt));
  dbe_xmldom.freedocument(doc);
END;
/
-- Expected result:
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.FREEELEMENT

  Frees a DOMElement node. The prototype of the DBE_XMLDOM.FREEELEMENT function is as follows:

```
DBE_XMLDOM.FREEELEMENT(
  elem     IN     DOMELEMENT);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-431 DBE_XMLDOM.FREEELEMENT API parameters

Parameter	Description
elem	Specified DOMElement node.

Example:

```
-- Obtain a DOMElement node from the DOC tree and free the node. Check whether the node is
empty before and after the node is freed.
DECLARE
  doc dbe_xmldom.domdocument;
  elem dbe_xmldom.domelement;
  node dbe_xmldom.domnode;
  node1 dbe_xmldom.domnode;
  nodelist DBE_XMLDOM.DOMNODELIST;
  len INTEGER;
  buffer varchar2(1010);
BEGIN
  doc := dbe_xmldom.newdomdocument('<?xml version="1.0"?>
     <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)>
     <!ELEMENT to (#PCDATA)>
     <!ELEMENT from (#PCDATA)>
     <!ELEMENT heading (#PCDATA)>
     <!ELEMENT body (#PCDATA)>]>
     <note>
       <to>Chinese</to>
       <from>Jani</from>
       <heading>Reminder</heading>
       <body>Don''t forget me this weekend!</body>
     </note>');
  elem := dbe_xmldom.GETDOCUMENTELEMENT(doc);
  IF DBE_XMLDOM.ISNULL(elem) THEN
      dbe_output.print_line('IS NULL');
  ELSE
      dbe_output.print_line('NOT NULL');
  END IF;
  dbe_xmldom.FREEELEMENT(elem);

  IF DBE_XMLDOM.ISNULL(elem) THEN
       dbe_output.print_line('IS NULL');
  ELSE
       dbe_output.print_line('NOT NULL');
  END IF;
END;
/
-- Expected result:
NOT NULL
IS NULL
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.FREENODE

  Frees a DOMNode node. The prototype of the DBE_XMLDOM.FREENODE function is as follows:

```
DBE_XMLDOM.FREENODE(
  n IN DOMNODE);
```

Table 2-432 DBE_XMLDOM.FREENODE API parameters

Parameter	Description
n	Specified DOMNode node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- After GaussDB performs the FREENODE operation, the freed node is not available again. After the database ORA performs the FREENODE operation, the freed node is available again and becomes another node.
- When other APIs call the freed DOMNode node, the calling is different from that in database ORA.

Example:

```
-- Obtain a DOMNode node from the DOC tree and free the node. Check whether the node is empty
before and after the node is freed.
DECLARE
  doc dbe_xmldom.domdocument;
  node dbe_xmldom.domnode;
  node1 dbe_xmldom.domnode;
  nodelist DBE_XMLDOM.DOMNODELIST;
  len INTEGER;
  buffer1 varchar2(1010);
BEGIN
  doc := dbe_xmldom.newdomdocument('<?xml version="1.0"?>
     <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)>
     <!ELEMENT to (#PCDATA)>
     <!ELEMENT from (#PCDATA)>
     <!ELEMENT heading (#PCDATA)>
     <!ELEMENT body (#PCDATA)>]>
     <note>
       <to>Chinese</to>
       <from>Jani</from>
       <heading>Reminder</heading>
       <body>Don''t forget me this weekend!</body>
     </note>');
  node := dbe_xmldom.makenode(doc);
  node := dbe_xmldom.GETFIRSTCHILD(node);
  IF DBE_XMLDOM.ISNULL(node) THEN
        dbe_output.print_line('IS NULL');
  ELSE
        dbe_output.print_line('NOT NULL');
  END IF;
  DBE_XMLDOM.FREENODE(node);
  IF DBE_XMLDOM.ISNULL(node) THEN
        dbe_output.print_line('IS NULL');
  ELSE
        dbe_output.print_line('NOT NULL');
  END IF;
END;
/
-- Expected result:
NOT NULL
IS NULL
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.FREENODELIST

  Frees a DOMNodeList node. The prototype of the DBE_XMLDOM.FREENODE function is as follows:

```
DBE_XMLDOM.GETLENGTH(
  nl     IN    DOMNODELIST);
```

Table 2-433 DBE_XMLDOM.FREENODELIST API parameters

Parameter	Description
nl	Specified DOMNodeList node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- A DOMNodeList node will be completed freed by FREENODELIST.
- When other APIs call the freed DOMNodelist node, the calling is different from that in database ORA.
- The input parameter **freenodelist** cannot be empty.

Example:

```
-- Obtain a DOMNodeList node from the DOC tree and free the node. Check the node length before and after the node is freed.
DECLARE
  doc dbe_xmldom.domdocument;
  node dbe_xmldom.domnode;
  node1 dbe_xmldom.domnode;
  nodelist DBE_XMLDOM.DOMNODELIST;
  len INTEGER;
  buffer1 varchar2(1010);
BEGIN
  doc := dbe_xmldom.newdomdocument('<?xml version="1.0"?>
     <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)>
     <!ELEMENT to (#PCDATA)>
     <!ELEMENT from (#PCDATA)>
     <!ELEMENT heading (#PCDATA)>
     <!ELEMENT body (#PCDATA)>]>
     <note>
       <to>Chinese</to>
       <from>Jani</from>
       <heading>Reminder</heading>
       <body>Don''t forget me this weekend!</body>
     </note>');
  node := dbe_xmldom.makenode(doc);
  node := dbe_xmldom.GETFIRSTCHILD(node);
  nodelist := DBE_XMLDOM.GETCHILDNODES(node);
  len := DBE_XMLDOM.GETLENGTH(nodelist);
  RAISE NOTICE 'len :  %', len;
  DBE_XMLDOM.FREENODELIST(nodelist);
  len := DBE_XMLDOM.GETLENGTH(nodelist);
  RAISE NOTICE 'len :  %', len;
END;
/
-- Expected result:
NOTICE:  len :  4
NOTICE:  len :  0
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETATTRIBUTE

  Returns the attribute values of a DOMElement object by name. The prototype of the DBE_XMLDOM.GETATTRIBUTE function is as follows:

  ```
  DBE_XMLDOM.GETATTRIBUTE(
    elem       IN      DOMELEMENT,
    name       IN      VARCHAR2)
  RETURN VARCHAR2;
  ```

  Returns the attribute values of a DOMElement object by name and namespace URI. The prototype of the DBE_XMLDOM.GETATTRIBUTE function is as follows:

  ```
  DBE_XMLDOM.GETATTRIBUTE(
    elem      IN     DOMELEMENT,
    name      IN     VARCHAR2,
    ns        IN     VARCHAR2)
  RETURN VARCHAR2;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-434 DBE_XMLDOM.GETATTRIBUTE API parameters

Parameter	Description
elem	Specified DOMElement node.
name	Attribute name.
ns	Namespace.

NOTE

- The **ns** parameter of the DBE_XMLDOM.GETATTRIBUTE API does not support the asterisk (*) parameter.
- GaussDB does not support the namespace prefix as an attribute, and the value of the prefix cannot be queried through the DBE_XMLDOM.GETATTRIBUTE API.

Example:

```
-- 1. Return the attribute values of a DOMElement object by name.
DECLARE
  doc dbe_xmldom.domdocument;
  elem dbe_xmldom.domelement;
  docnode DBE_XMLDOM.DOMNode;
  buffer varchar2(1010);
  value  varchar2(1000);
BEGIN
  doc := dbe_xmldom.newDOMDocument();
  elem := DBE_XMLDOM.CREATEELEMENT(doc, 'root');
  DBE_XMLDOM.setattribute(elem, 'len', '50cm');
  docnode := DBE_XMLDOM.appendChild(DBE_XMLDOM.makeNode(doc),
DBE_XMLDOM.makeNode(elem));
  value := DBE_XMLDOM.getattribute(elem, 'len');
  dbe_output.print_line('value: ');
  dbe_output.print_line(value);
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
END;
/
-- Expected result:
value:
50cm
buffer:
<?xml version="1.0" encoding="UTF-8"?>
<root len="50cm"/>

ANONYMOUS BLOCK EXECUTE

-- 2. Return the attribute values of a DOMElement object by name and namespace URI.
DECLARE
  doc dbe_xmldom.domdocument;
  elem dbe_xmldom.domelement;
  docnode DBE_XMLDOM.DOMNode;
  buffer varchar2(1010);
  value  varchar(1000);

BEGIN
  doc := dbe_xmldom.newDOMDocument();
  elem := DBE_XMLDOM.CREATEELEMENT(doc, 'root');
  DBE_XMLDOM.setattribute(elem, 'len', '50cm', 'www.xxxxx.com');
  docnode := DBE_XMLDOM.appendChild(DBE_XMLDOM.makeNode(doc),
DBE_XMLDOM.makeNode(elem));
  value := DBE_XMLDOM.getattribute(elem, 'len', 'www.xxxxx.com');
  dbe_output.print_line('value: ');
  dbe_output.print_line(value);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
END;
/
-- Expected result:
value:
50cm
buffer:
<?xml version="1.0" encoding="UTF-8"?>
<root len="50cm"/>

ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETATTRIBUTES

  Returns the attribute values of a DOMNode node as a map. The prototype of the DBE_XMLDOM.GETATTRIBUTES function is as follows:

```
DBE_XMLDOM.GETATTRIBUTES(
  n IN DOMNode)
RETURN DOMNAMEDNODEMAP;
```

Table 2-435 DBE_XMLDOM.GETATTRIBUTES API parameters

Parameter	Description
n	Specified DOMNode node.

Example:

```
-- Obtain the attribute values of a DOMNode node, return an object of the DOMNamedNodeMap type, and output the length of the DOMNamedNodeMap object and the value of the first node.
DECLARE
  doc dbe_xmldom.domdocument;
  node dbe_xmldom.domnode;
  node1 dbe_xmldom.domnode;
  len INTEGER;
  map DBE_XMLDOM.DOMNAMEDNODEMAP;
  buffer1 varchar2(1010);
BEGIN
  doc := dbe_xmldom.newdomdocument('<?xml version="1.0"?>
      <note a="16" b="176" c="asd">
        <to>Chinese</to>
        <from>Jani</from>
        <heading>Reminder</heading>
        <body>Don''t forget me this weekend!</body>
      </note>');
  node := dbe_xmldom.makenode(doc);
  node := dbe_xmldom.GETFIRSTCHILD(node);
  map := DBE_XMLDOM.GETATTRIBUTES(node);
  IF DBE_XMLDOM.ISNULL(map) THEN
        dbe_output.print_line('IS NULL');
  ELSE
        dbe_output.print_line('NOT NULL');
  END IF;
    len := DBE_XMLDOM.GETLENGTH(map);
  RAISE NOTICE 'len :  %', len;
   node1 := DBE_XMLDOM.ITEM(map, 0);
  dbe_xmldom.writetobuffer(node1, buffer1);
  dbe_output.print_line('buffer1: ');
  dbe_output.print_line(buffer1);
END;
/
-- Expected result:
NOT NULL
NOTICE:  len :  3
buffer1:
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
16
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETCHILDNODES

  Converts several subnodes under a node into a node list. The prototype of the DBE_XMLDOM.GETCHILDNODES function is as follows:

```
DBE_XMLDOM.GETCHILDNODES(
  n IN DOMNode)
RETURN DOMNodeList;
```

Table 2-436 DBE_XMLDOM.GETCHILDNODES API parameters

Parameter	Description
n	Specified DOMNode node.

Example:

```
-- After obtaining the first subnode of the DOC tree, convert several subnodes under the node into a node list and output the length information.
DECLARE
    doc dbe_xmldom.domdocument;
    doc_node dbe_xmldom.domnode;
    root_node dbe_xmldom.domnode;
    node_list dbe_xmldom.domnodelist;
    list_len integer;
    node_name varchar2(1000);
    node_type integer;
    buffer varchar2(1010);
BEGIN
   doc := dbe_xmldom.newdomdocument('<?xml version="1.0"?>
       <note>
         <to>Chinese</to>
         <from>Jani</from>
         <heading>Reminder</heading>
         <body>Don''t forget me this weekend!</body>
       </note>');
    doc_node := DBE_XMLDOM.MAKENODE(doc);
    root_node := DBE_XMLDOM.GETFIRSTCHILD(doc_node);
    node_name := DBE_XMLDOM.GETNODENAME(root_node);
    node_type := DBE_XMLDOM.GETNODETYPE(root_node);
    dbe_output.print_line(node_name);
    dbe_output.print_line(node_type);
    node_list := DBE_XMLDOM.GETCHILDNODES(root_node);
    list_len := DBE_XMLDOM.GETLENGTH(node_list);
    dbe_output.print_line(list_len);
END;
/
-- Expected result:
note
1
4
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETCHILDRENBYTAGNAME

  Returns the subnodes of a DOMElement node by name. The prototype of the DBE_XMLDOM.GETCHILDRENBYTAGNAME function is as follows:

```
DBE_XMLDOM.GETCHILDRENBYTAGNAME (
  elem       IN      DOMELEMENT,
  name       IN      VARCHAR2)
RETURN DOMNODELIST;
```

  Returns the subnodes of a DOMElement node by name and namespace. The prototype of the DBE_XMLDOM.GETCHILDRENBYTAGNAME function is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_XMLDOM.GETCHILDRENBYTAGNAME (
  elem      IN     DOMELEMENT,
  name      IN     VARCHAR2,
  ns        IN     VARCHAR2)
 RETURN DOMNODELIST;
```

Table 2-437 DBE_XMLDOM.GETCHILDRENBYTAGNAME API parameters

Parameter	Description
elem	Specified DOMElement node.
name	Attribute name.
ns	Namespace.

NOTE

The **ns** parameter of the DBE_XMLDOM.GETCHILDRENBYTAGNAME API does not support the asterisk (*) parameter. To obtain all attributes of a node, use the DBE_XMLDOM.GETCHILDNODES API.

Example:

```
-- 1. Return the subnodes of a DOMElement node by name.
DECLARE
  doc dbe_xmldom.domdocument;
  elem dbe_xmldom.domelement;
  docnodelist dbe_xmldom.domnodelist;
  node_elem dbe_xmldom.domelement;
  node dbe_xmldom.domnode;
  buffer varchar2(1010);
  value  varchar2(1000);
BEGIN
  doc := dbe_xmldom.newdomdocument('<?xml version="1.0" encoding="UTF-8"?>
     <students age="16" hight="176">
     <student>
       <name>Jerry</name><age>519</age><gender>man</gender><abc>12345</abc>
     </student>
     <student>
       <name>Bob</name><age>245</age><gender>woman</gender><abc>54321</abc>
     </student>
     </students>');
  elem := dbe_xmldom.GETDOCUMENTELEMENT(doc);
  docnodelist := dbe_xmldom.GETCHILDRENBYTAGNAME(elem, 'student');
  node := dbe_xmldom.ITEM(docnodelist, 0);
  node_elem := dbe_xmldom.makeelement(node);
  value := DBE_XMLDOM.gettagname(node_elem);
  dbe_output.print_line('value: ');
  dbe_output.print_line(value);
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
END;
/
-- Expected result:
value:
student
buffer:
<?xml version="1.0" encoding="UTF-8"?>
<students age="16" hight="176">
 <student>
  <name>Jerry</name>
  <age>519</age>
  <gender>man</gender>
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    <abc>12345</abc>
  </student>
  <student>
    <name>Bob</name>
    <age>245</age>
    <gender>woman</gender>
    <abc>54321</abc>
  </student>
</students>

ANONYMOUS BLOCK EXECUTE

-- 2. Return the child nodes of DOMElement by name and namespace.
DECLARE
  doc dbe_xmldom.domdocument;
  elem dbe_xmldom.domelement;
  node dbe_xmldom.domnode;
  node_elem dbe_xmldom.domelement;
  docnodelist dbe_xmldom.domnodelist;
  buffer varchar2(1010);
  value varchar2(1000);
BEGIN
  doc := dbe_xmldom.newdomdocument('
    <note xmlns:h="www.xxxxx.com">
    <h:to h:len="50cm">Chinese</h:to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don''t forget me this weekend!</body>
    </note>');
  elem := dbe_xmldom.GETDOCUMENTELEMENT(doc);
  docnodelist := dbe_xmldom.GETCHILDRENBYTAGNAME(elem, 'to', 'www.xxxxx.com');
  node := dbe_xmldom.ITEM(docnodelist, 0);
  node_elem := dbe_xmldom.makeelement(node);
  value := DBE_XMLDOM.getattribute(node_elem, 'len');
  dbe_output.print_line('value: ');
  dbe_output.print_line(value);
END;
/
-- Expected result:
value:
50cm
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETDOCUMENTELEMENT

  Returns the first subnode of the specified document. The prototype of the DBE_XMLDOM.GETDOCUMENTELEMENT function is as follows:

```
DBE_XMLDOM.GETDOCUMENTELEMENT(
  doc     IN      DOMDOCUMENT)
 RETURN DOMELEMENT;
```

Table 2-438 DBE_XMLDOM.GETDOCUMENTELEMENT API parameters

Parameter	Description
doc	Specified DOMDocument node.

Example:

```
-- Obtain the first subnode in the DOC tree and output the node name.
DECLARE
  doc dbe_xmldom.domdocument;
  elem dbe_xmldom.domelement;
  doc_node dbe_xmldom.DOMNODE;
  root_elmt dbe_xmldom.DOMELEMENT;
  root_node dbe_xmldom.DOMNODE;
  value  varchar(1000);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
BEGIN
  doc := dbe_xmldom.newdomdocument();
  doc_node := dbe_xmldom.MAKENODE(doc);
  root_elmt := dbe_xmldom.CREATEELEMENT(doc,'staff');
  root_node:=dbe_xmldom.APPENDCHILD(doc_node, dbe_xmldom.MAKENODE(root_elmt));
  elem := dbe_xmldom.GETDOCUMENTELEMENT(doc);
  value := DBE_XMLDOM.gettagname(elem);
  dbe_output.print_line(value);
END;
/
-- Expected result:
staff
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETFIRSTCHILD

  Returns the first subnode of a node. The prototype of the DBE_XMLDOM.GETFIRSTCHILD function is as follows:

```
DBE_XMLDOM.GETFIRSTCHILD(
  n IN DOMNODE)
RETURN DOMNODE;
```

Table 2-439 DBE_XMLDOM.GETFIRSTCHILD API parameters

Parameter	Description
n	Specified DOMNode node.

Example:

```
-- Obtain the name and type of the first subnode after the DOC tree is converted to the DOMNode type, and then obtain the name of the first subnode of the obtained first DOMNode node.
DECLARE
   doc dbe_xmldom.domdocument;
   doc_node dbe_xmldom.domnode;
    root_node dbe_xmldom.domnode;
   inside_node dbe_xmldom.domnode;
   node_name varchar2(1000);
   node_type integer;
BEGIN
   doc := dbe_xmldom.newdomdocument('<?xml version="1.0" encoding="UTF-8"?>
      <students age="16" hight="176">
      <student1>
        <name>Jerry</name><age>519</age><gender>man</gender><abc>12345</abc>
      </student1>
      <student2>
        <name>Bob</name><age>245</age><gender>woman</gender><abc>54321</abc>
      </student2>
      </students>');
   doc_node := DBE_XMLDOM.MAKENODE(doc);
   root_node := DBE_XMLDOM.GETFIRSTCHILD(doc_node);
   node_name := DBE_XMLDOM.GETNODENAME(root_node);
   node_type := DBE_XMLDOM.GETNODETYPE(root_node);
   dbe_output.print_line(node_name);
   dbe_output.print_line(node_type);
   inside_node := DBE_XMLDOM.GETFIRSTCHILD(root_node);
   node_name := DBE_XMLDOM.GETNODENAME(inside_node);
   dbe_output.print_line(node_name);
END;
/
-- Expected result:
students
1
student1
ANONYMOUS BLOCK EXECUTE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_XMLDOM.GETLASTCHILD

  Returns the last subnode of a node. The prototype of the DBE_XMLDOM.GETLASTCHILD function is as follows:

```
DBE_XMLDOM.GETLASTCHILD(
  n IN DOMNODE)
RETURN DOMNODE;
```

  Table 2-440 DBE_XMLDOM.GETLASTCHILD API parameters

Parameter	Description
n	Specified DOMNode node.

  Example:

```
-- Obtain the name and type of the last subnode after the DOC tree is converted to the DOMNode type, and then obtain the name of the last subnode of the obtained last DOMNode node.
DECLARE
    doc dbe_xmldom.domdocument;
    doc_node dbe_xmldom.domnode;
    root_node dbe_xmldom.domnode;
    inside_node dbe_xmldom.domnode;
    node_name varchar2(1000);
    node_type integer;
BEGIN
    doc := dbe_xmldom.newdomdocument('<?xml version="1.0" encoding="UTF-8"?>
      <students age="16" hight="176">
      <student1>
        <name>Jerry</name><age>519</age><gender>man</gender><abc>12345</abc>
      </student1>
      <student2>
        <name>Bob</name><age>245</age><gender>woman</gender><abc>54321</abc>
      </student2>
      </students>');
    doc_node := DBE_XMLDOM.MAKENODE(doc);
    root_node := DBE_XMLDOM.GETFIRSTCHILD(doc_node);
    node_name := DBE_XMLDOM.GETNODENAME(root_node);
    node_type := DBE_XMLDOM.GETNODETYPE(root_node);
    dbe_output.print_line(node_name);
    dbe_output.print_line(node_type);
    inside_node := DBE_XMLDOM.GETLASTCHILD(root_node);
    node_name := DBE_XMLDOM.GETNODENAME(inside_node);
    dbe_output.print_line(node_name);
END;
/
-- Expected result:
students
1
student2
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETLENGTH

  Returns the number of subnodes under a DOMNamedNodeMap node. The prototype of the DBE_XMLDOM.GETLENGTH function is as follows:

```
DBE_XMLDOM.GETLENGTH(
  nnm     IN    DOMNAMEDNODEMAP)
 RETURN NUMBER;
```

  Returns the number of subnodes under a DOMNodeList node. The prototype of the DBE_XMLDOM.GETLENGTH function is as follows:

```
DBE_XMLDOM.GETLENGTH(
  nl    IN    DOMNODELIST)
 RETURN NUMBER;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-441 DBE_XMLDOM.GETLENGTH API parameters

Parameter	Description
nnm	Specified DOMNamedNodeMap node.
nl	Specified DOMNodeList node.

Example:

```
-- 1. Declare a DOMNamedNodeMap parameter in a function.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  elem DBE_XMLDOM.DOMElement;
  map DBE_XMLDOM.DOMNAMEDNODEMAP;
  node DBE_XMLDOM.DOMNODE;
  buf varchar2(10000);
  len INTEGER;
BEGIN
  doc := dbe_xmldom.newdomdocument('<?xml version="1.0"?>
     <bookstore category="web" cover="paperback">
     <book category="cooking">
       <title lang="en">Everyday Italian</title>
       <author>Giada De Laurentiis</author>
       <year>2005</year>
       <price>30.00</price>
     </book>
     </bookstore>');
  elem := DBE_XMLDOM.GETDOCUMENTELEMENT(doc);
  node := DBE_XMLDOM.MAKENODE(elem);
  map := DBE_XMLDOM.GETATTRIBUTES(node);
  len := DBE_XMLDOM.GETLENGTH(map);
  DBE_OUTPUT.print_line(len);
END;
/
-- Expected result:
2
ANONYMOUS BLOCK EXECUTE

-- 2. Declare a NodeList parameter in a function.
DECLARE
  doc dbe_xmldom.domdocument;
  node dbe_xmldom.domnode;
  node1 dbe_xmldom.domnode;
  nodelist DBE_XMLDOM.DOMNODELIST;
  len INTEGER;
  buffer1 varchar2(1010);
BEGIN
  doc := dbe_xmldom.newdomdocument('<?xml version="1.0" encoding="UTF-8"?>
     <students age="16" hight="176">
     <student>
       <name>Jerry</name><age>519</age><gender>man</gender><abc>12345</abc>
     </student>
     <student>
       <name>Jerry</name><age>519</age><gender>man</gender><abc>12345</abc>
     </student>
     </students>');
  node := dbe_xmldom.makenode(doc);
  node := dbe_xmldom.GETFIRSTCHILD(node);
  nodelist := DBE_XMLDOM.GETCHILDNODES(node);
  len := DBE_XMLDOM.GETLENGTH(nodelist);
  RAISE NOTICE 'len :  %', len;
END;
/
-- Expected result:
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
NOTICE:  len :  2
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETLOCALNAME

  Returns the local name of the given DOMAttr node. The prototype of the DBE_XMLDOM.MAKENODE function is as follows:

```
DBE_XMLDOM.GETLOCALNAME(
  a       IN     DOMATTR)
RETURN VARCHAR2;
```

  Returns the local name of the given DOMElement node. The prototype of the DBE_XMLDOM.MAKENODE function is as follows:

```
DBE_XMLDOM.GETLOCALNAME(
  elem       IN     DOMELEMENT)
RETURN VARCHAR2;
```

  Returns the local name of the given DOMNode node. The prototype of the DBE_XMLDOM.MAKENODE function is as follows:

```
DBE_XMLDOM.GETLOCALNAME(
  n      IN     DOMNODE,
  data   OUT    VARCHAR2);
```

Table 2-442 DBE_XMLDOM.GETLOCALNAME API parameters

Parameter	Description
a	Specified DOMAttr node.
elem	Specified DOMElement node.
n	Specified DOMNode node.
data	Returned local name.

Example:

```
-- 1. Use the createAttribute function to generate an attr node to obtain the local name.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  root DBE_XMLDOM.DOMElement;
  attr1 DBE_XMLDOM.DOMATTR;
  value VARCHAR2(1000);
BEGIN
  doc := DBE_xmldom.newdomdocument('<?xml version="1.0"?>
    <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)>
    <!ELEMENT to (#PCDATA)>
    <!ELEMENT from (#PCDATA)>
    <!ELEMENT heading (#PCDATA)>
    <!ELEMENT body (#PCDATA)>]>
    <note><to>Chinese</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don''t forget me this weekend!</body>
    </note>');
  attr1 := DBE_XMLDOM.createAttribute(doc,'len');
  value := DBE_XMLDOM.getlocalname(attr1);
  DBE_output.print_line('value: ');
  DBE_output.print_line(value);
END;
/
-- Expected result:
value:
len
ANONYMOUS BLOCK EXECUTE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- 2. Use the createElement function to generate a DOMElement node and obtain its local name.
DECLARE
  doc DBE_xmldom.domdocument;
  elem DBE_xmldom.domelement;
  value  varchar2(10000);
BEGIN
  doc := DBE_xmldom.newdomdocument();
  elem := DBE_XMLDOM.createELEMENT(doc, 'root');
  value := DBE_XMLDOM.getlocalname(elem);
  DBE_output.print_line('value: ');
  DBE_output.print_line(value);
END;
/
-- Expected result:
value:
root
ANONYMOUS BLOCK EXECUTE

-- 3. Convert a DOMElement to a DOMNode node and obtain its local name.
DECLARE
  doc DBE_xmldom.domdocument;
  elem DBE_xmldom.domelement;
  node DBE_xmldom.domnode;
  value  varchar2(100);
  buf varchar2(100);
BEGIN
  doc := DBE_xmldom.newdomdocument();
  elem := DBE_XMLDOM.createELEMENT(doc, 'root');
  node := DBE_xmldom.makenode(elem);
  DBE_XMLDOM.getlocalname(node, buf);
  DBE_output.print_line('buf: ');
  DBE_output.print_line(buf);
END;
/
-- Expected result:
buf:
root
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETNAMEDITEM

  Returns the node specified by name. The prototype of the DBE_XMLDOM.GETNAMEDITEM function is as follows:

```
DBE_XMLDOM.GETNAMEDITEM(
  nnm IN DOMNAMEDNODEMAP,
  name IN VARCHAR2)
RETURN DOMNODE;
```

  Returns the node specified by name and namespace. The prototype of the DBE_XMLDOM.GETNAMEDITEM function is as follows:

```
DBE_XMLDOM.GETNAMEDITEM(
  nnm IN DOMNAMEDNODEMAP,
  name IN VARCHAR2,
  ns IN VARCHAR2)
RETURN DOMNODE;
```

Table 2-443 DBE_XMLDOM.GETNAMEDITEM API parameters

Parameter	Description
nnm	Specified DOMNamedNodeMap object.
name	Name of the element to be retrieved.
ns	Namespace.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The values of **name** and **nnm** can be null, but they are required arguments.
- The default maximum length of **name** and **ns** is 32767. If the length exceeds 32767, an error is reported.
- The values of **name** and **ns** can be of the int type and contain more than 127 bits.

Example:

```
-- 1. Return the node specified by name.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  elem DBE_XMLDOM.DOMElement;
  map DBE_XMLDOM.DOMNAMEDNODEMAP;
  node DBE_XMLDOM.DOMNODE;
  node2 DBE_XMLDOM.DOMNODE;
  buf varchar2(1000);
  buf2 varchar2(1000);
BEGIN
  doc := dbe_xmldom.newdomdocument('<bookstore category="web" cover="paperback">
     <book category="cooking"><title lang="en">Everyday Italian</title>
     <author>Giada De Laurentiis</author><year>2005</year>
     <price>30.00</price></book></bookstore>');
  elem := DBE_XMLDOM.GETDOCUMENTELEMENT(doc);
  node := DBE_XMLDOM.MAKENODE(elem);
  map := DBE_XMLDOM.GETATTRIBUTES(node);
  node2:= DBE_XMLDOM.GETNAMEDITEM(map,'category');
  DBE_XMLDOM.writeToBuffer(node2, buf2);
  dbe_output.print_line(buf2);
END;
/
-- Expected result:
web
ANONYMOUS BLOCK EXECUTE

-- 2. Return the node specified by name and namespace.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  root DBE_XMLDOM.DOMElement;
  elem DBE_XMLDOM.DOMElement;
  map DBE_XMLDOM.DOMNAMEDNODEMAP;
  node DBE_XMLDOM.DOMNODE;
  buf varchar2(1000);
  buf2 varchar2(1000);
BEGIN
  doc := dbe_xmldom.newdomdocument('<h:table xmlns:h="http://www.w3.org/TR/html4/">
      <h:tr h:id="10"><h:td >Apples</h:td>
      <h:td>Bananas</h:td></h:tr></h:table>');
  root := DBE_XMLDOM.getDocumentElement(doc);
  node := DBE_XMLDOM.MAKENODE(root);
  node := dbe_xmldom.GETFIRSTCHILD(node);
  map := DBE_XMLDOM.GETATTRIBUTES(node);
  node := DBE_XMLDOM.GETNAMEDITEM(map,'id','http://www.w3.org/TR/html4/');
  DBE_XMLDOM.writeToBuffer(node, buf2);
  dbe_output.print_line(buf2);
END;
/
-- Expected result:
10
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETNEXTSIBLING

  Returns the next node. The prototype of the DBE_XMLDOM.GETNEXTSIBLING function is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_XMLDOM.GETNEXTSIBLING(
  n  IN  DOMNODE)
RETURN DOMNODE;
```

Table 2-444 DBE_XMLDOM.GETNEXTSIBLING API parameters

Parameter	Description
n	Specified DOMNode node.

Example:

```
-- Obtain the first subnode after the DOC tree is converted into the DOMNode type, and obtain the name of the first subnode of the obtained first DOMNode node. Then, obtain the name of the next node through DBE_XMLDOM.GETNEXTSIBLING.
DECLARE
    doc dbe_xmldom.domdocument;
    doc_node dbe_xmldom.domnode;
    root_node dbe_xmldom.domnode;
    inside_node dbe_xmldom.domnode;
    node_name varchar2(1000);
    node_type integer;
BEGIN
   doc := dbe_xmldom.newdomdocument('<computer size="ITX">
        <cpu>Ryzen 9 3950X</cpu>
        <ram>32GBx2 DDR4 3200MHz</ram>
        <motherboard>X570i</motherboard>
    </computer>');
    doc_node := DBE_XMLDOM.MAKENODE(doc);
    root_node := DBE_XMLDOM.GETFIRSTCHILD(doc_node);
    node_name := DBE_XMLDOM.GETNODENAME(root_node);
    node_type := DBE_XMLDOM.GETNODETYPE(root_node);
    dbe_output.print_line(node_name);
    dbe_output.print_line(node_type);
    inside_node := DBE_XMLDOM.GETFIRSTCHILD(root_node);
    node_name := DBE_XMLDOM.GETNODENAME(inside_node);
    dbe_output.print_line(node_name);
    inside_node := DBE_XMLDOM.GETNEXTSIBLING(inside_node);
    node_name := DBE_XMLDOM.GETNODENAME(inside_node);
    dbe_output.print_line(node_name);
END;
/
-- Expected result:
computer
1
cpu
ram
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETNODENAME

  Returns the name of a node. The prototype of the DBE_XMLDOM.GETNODENAME function is as follows:

```
DBE_XMLDOM.GETNODENAME(
  n  IN  DOMNODE)
RETURN  VARCHAR2;
```

Table 2-445 DBE_XMLDOM.GETNODENAME API parameters

Parameter	Description
n	Specified DOMNode node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
-- Obtain the name of the specified DOMNode node from the DOC tree.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  root DBE_XMLDOM.DOMElement;
  root_node DBE_XMLDOM.DOMNode;
  inside_node DBE_XMLDOM.DOMNode;
  buf VARCHAR2(1000);
BEGIN
   doc := dbe_xmldom.newdomdocument('<bookstore category="web" cover="paperback">
      <book category="cooking"><title lang="en">Everyday Italian</title>
      <author>Giada De Laurentiis</author><year>2005</year>
      <price>30.00</price></book></bookstore>');
   root := DBE_XMLDOM.getDocumentElement(doc);
   root_node := DBE_XMLDOM.MAKENODE(root);
   inside_node := DBE_XMLDOM.GETFIRSTCHILD(root_node);
   buf := DBE_XMLDOM.GETNODENAME(inside_node);
   dbe_output.print_line(buf);
END;
/
-- Expected result:
book
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETNODETYPE

  Returns the type of a node. The prototype of the DBE_XMLDOM.GETNODETYPE function is as follows:

```
DBE_XMLDOM.GETNODETYPE(
   n  IN  DOMNODE)
RETURN  NUMBER;
```

Table 2-446 DBE_XMLDOM.GETNODETYPE API parameters

Parameter	Description
n	Specified DOMNode node.

Example:

```
-- Obtain the type of the specified DOMNode node from the DOC tree.
DECLARE
   doc DBE_XMLDOM.DOMDocument;
   doc_node DBE_XMLDOM.DOMNode;
   num number;
   buf varchar2(1000);
BEGIN
   doc := dbe_xmldom.newdomdocument('<bookstore category="web" cover="paperback">
      <book category="cooking"><title lang="en">Everyday Italian</title>
      <author>Giada De Laurentiis</author><year>2005</year>
      <price>30.00</price></book></bookstore>');
   doc_node := DBE_XMLDOM.makeNode(doc);
   num := DBE_XMLDOM.GETNODETYPE(doc_node);
   dbe_output.print_line(num);
   buf := DBE_XMLDOM.GETNODENAME(doc_node);
   dbe_output.print_line(buf);
END;
/
-- Expected result:
9
#document
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETNODEVALUE

  Returns the value of a DOMNode node. The prototype of the DBE_XMLDOM.GETNODEVALUE function is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_XMLDOM.GETNODEVALUE(
  n  IN  DOMNODE)
RETURN  VARCHAR2;
```

Table 2-447 DBE_XMLDOM.GETNODEVALUE API parameters

Parameter	Description
n	Specified DOMNode object.

Example:

```
-- Convert a DOMText node to a DOMNode node and obtain the value of the node.
DECLARE
  buf VARCHAR2(1000);
  doc DBE_XMLDOM.DOMDocument;
  text DBE_XMLDOM.DOMText;
  elem2 DBE_XMLDOM.DOMElement;
  node DBE_XMLDOM.DOMNode;
begin
  doc := DBE_XMLDOM.NEWDOMDOCUMENT();
  text := DBE_XMLDOM.createTextNode(doc, 'aaa');
  DBE_XMLDOM.SETNODEVALUE(DBE_XMLDOM.makeNode(text), 'ccc');
  buf := DBE_XMLDOM.GETNODEVALUE(DBE_XMLDOM.makeNode(text));
  DBE_OUTPUT.print_line(buf);
end;
/
-- Expected result:
ccc
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETPARENTNODE

  Returns the parent node of the given DOMNode node. The prototype of the DBE_XMLDOM.GETPARENTNODE function is as follows:

```
DBE_XMLDOM.GETPARENTNODE(
  n  IN  DOMNODE)
RETURN  DOMNODE;
```

Table 2-448 DBE_XMLDOM.GETPARENTNODE API parameters

Parameter	Description
n	Specified DOMNode object.

Example:

```
-- Add a node to the DOC tree and obtain the name of its parent node.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  doc1 DBE_XMLDOM.DOMDocument;
  root DBE_XMLDOM.DOMElement;
  child1 DBE_XMLDOM.DOMElement;
  child2 DBE_XMLDOM.DOMElement;
  attr DBE_XMLDOM.DOMAttr;
  text DBE_XMLDOM.DOMTEXT;
  node DBE_XMLDOM.DOMNode;
  parent DBE_XMLDOM.DOMNode;
  buf varchar2(1000);
BEGIN
  doc := DBE_XMLDOM.newDOMDocument();
  root := DBE_XMLDOM.createElement(doc, 'root');
  node := DBE_XMLDOM.appendChild(DBE_xmldom.makeNode(doc),DBE_xmldom.makeNode(root));
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  child1 := DBE_XMLDOM.createElement(doc, 'child1');
  node := DBE_XMLDOM.appendChild(DBE_XMLDOM.makeNode(root),
DBE_XMLDOM.makeNode(child1));
  child2 := DBE_XMLDOM.createElement(doc, 'child2');
  node := DBE_XMLDOM.appendChild(DBE_XMLDOM.makeNode(child1),
DBE_XMLDOM.makeNode(child2));
  parent := DBE_XMLDOM.GETPARENTNODE(DBE_XMLDOM.makeNode(child2));
  buf := DBE_XMLDOM.GETNODENAME(parent);
  DBE_OUTPUT.print_line(buf);
END;
/
-- Expected result:
child1
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETTAGNAME

  Returns the tag name of the specified DOMElement node. The prototype of the DBE_XMLDOM.GETTAGNAME function is as follows:

```
DBE_XMLDOM.GETTAGNAME(
  elem  IN  DOMELEMENT)
RETURN  VARCHAR2;
```

Table 2-449 DBE_XMLDOM.GETTAGNAME API parameters

Parameter	Description
elem	Specified DOMElement node.

Example:

```
-- Obtain the tag name of a DOMElement node.
DECLARE
  doc dbe_xmldom.domdocument;
  elem dbe_xmldom.domelement;
  buffer varchar2(1010);
  value  varchar(1000);
BEGIN
  doc := dbe_xmldom.newDOMDocument();
  elem := DBE_XMLDOM.CREATEELEMENT(DBE_XMLDOM.NEWDOMDOCUMENT(), 'root');
  value := DBE_XMLDOM.gettagname(elem);
  dbe_output.print_line('value: ');
  dbe_output.print_line(value);
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
END;
/
-- Expected result:
value:
root
buffer:
<?xml version="1.0" encoding="UTF-8"?>

ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.HASCHILDNODES

  Checks whether the DOMNode object has any subnode. The prototype of the DBE_XMLDOM.HASCHILDNODES function is as follows:

```
DBE_XMLDOM.HASCHILDNODES(
  n  IN  DOMNODE)
RETURN  BOOLEAN;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-450 DBE_XMLDOM.HASCHILDNODES API parameters

Parameter	Description
n	Specified DOMNode object.

Example:

```
-- Create a node named child1, mount it to the DOC tree, and add a node to child1. Then, check
whether the child1 node has any subnode.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  doc1 DBE_XMLDOM.DOMDocument;
  root DBE_XMLDOM.DOMElement;
  child1 DBE_XMLDOM.DOMElement;
  child2 DBE_XMLDOM.DOMElement;
  attr DBE_XMLDOM.DOMAttr;
  text DBE_XMLDOM.DOMTEXT;
  node DBE_XMLDOM.DOMNode;
  buf varchar2(1000);
BEGIN
  doc := DBE_XMLDOM.newDOMDocument();
  root := DBE_XMLDOM.createElement(doc, 'root');
  node := DBE_XMLDOM.appendChild(DBE_xmldom.makeNode(doc),DBE_xmldom.makeNode(root));
  child1 := DBE_XMLDOM.createElement(doc, 'child1');
  node := DBE_XMLDOM.appendChild(DBE_XMLDOM.makeNode(root),
DBE_XMLDOM.makeNode(child1));
  child2 := DBE_XMLDOM.createElement(doc, 'child2');
  node := DBE_XMLDOM.appendChild(DBE_XMLDOM.makeNode(child1),
DBE_XMLDOM.makeNode(child2));
  IF DBE_XMLDOM.HASCHILDNODES(DBE_XMLDOM.makeNode(child1)) THEN
    DBE_OUTPUT.print_line('HAS CHILD NODES');
  ELSE
    DBE_OUTPUT.print_line('NOT HAS CHILD NODES ');
  END IF;
END;
/
-- Expected result:
HAS CHILD NODES
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.IMPORTNODE

  Copies a node to another node and mounts the copied node to a specified document. If the type of the copied node does not belong to the 12 types specified by constants of XML DOM, an exception indicating that the type is not supported will be thrown. The prototype of the DBE_XMLDOM.IMPORTNODE function is as follows:

```
DBE_XMLDOM.IMPORTNODE(
  doc IN DOMDOCUMENT,
  importedNode IN DOMNODE,
  deep IN BOOLEAN)
RETURN DOMNODE;
```

Table 2-451 DBE_XMLDOM.IMPORTNODE API parameters

Parameter	Description
doc	Document to which the node is mounted.
importedNode	Node to be imported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
deep	Specifies whether to perform recursive import. • If the value is **TRUE**, the node and all its subnodes are imported. • If the value is **FALSE**, the node itself is imported.

Example:

```
-- Obtain the root2_node node in the DOC2 tree, copy it, and mount it to the DOC tree.
DECLARE
    doc dbe_xmldom.domdocument;
    doc2 dbe_xmldom.domdocument;
    doc_node dbe_xmldom.domnode;
    doc2_node dbe_xmldom.domnode;
     root_node dbe_xmldom.domnode;
    root2_node dbe_xmldom.domnode;
    import_node dbe_xmldom.domnode;
    result_node dbe_xmldom.domnode;
    buffer varchar2(1010);
BEGIN
    doc := dbe_xmldom.newdomdocument('<bookstore category="web" cover="paperback">
       <book category="cooking"><title lang="en">Everyday Italian</title>
       <author>Giada De Laurentiis</author><year>2005</year>
       <price>30.00</price></book></bookstore>');
    doc2 := dbe_xmldom.newdomdocument('<case>LIANLI TU150</case>');
    doc_node := DBE_XMLDOM.MAKENODE(doc);
    doc2_node := DBE_XMLDOM.MAKENODE(doc2);
    root_node := DBE_XMLDOM.GETFIRSTCHILD(doc_node);
    root2_node := DBE_XMLDOM.GETFIRSTCHILD(doc2_node);
    DBE_XMLDOM.WRITETOBUFFER(doc, buffer);
    dbe_output.print_line(buffer);
    import_node := DBE_XMLDOM.IMPORTNODE(doc, root2_node, TRUE);
    result_node := DBE_XMLDOM.APPENDCHILD(root_node, import_node);
    DBE_XMLDOM.WRITETOBUFFER(doc, buffer);
    dbe_output.print_line(buffer);
END;
/
-- Expected result:
<?xml version="1.0" encoding="UTF-8"?>
<bookstore category="web" cover="paperback">
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
</bookstore>

<?xml version="1.0" encoding="UTF-8"?>
<bookstore category="web" cover="paperback">
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <case>LIANLI TU150</case>
</bookstore>

ANONYMOUS BLOCK EXECUTE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_XMLDOM.ISNULL

  Checks whether the given DOMAttr node is null. If so, **TRUE** is returned. If not, **FALSE** is returned. The prototype of the DBE_XMLDOM.ISNULL function is as follows:

  ```
  DBE_XMLDOM.ISNULL(
    a       IN     DOMATTR)
  RETURN BOOLEAN;
  ```

  Checks whether the given DOMDocument node is null. If so, **TRUE** is returned. If not, **FALSE** is returned. The prototype of the DBE_XMLDOM.ISNULL function is as follows:

  ```
  DBE_XMLDOM.ISNULL(
    doc       IN     DOMDOCUMENT)
  RETURN BOOLEAN;
  ```

  Checks whether the given DOMElement node is null. If so, **TRUE** is returned. If not, **FALSE** is returned. The prototype of the DBE_XMLDOM.ISNULL function is as follows:

  ```
  DBE_XMLDOM.ISNULL(
    elem     IN     DOMELEMENT)
  RETURN BOOLEAN;
  ```

  Checks whether the given DOMNamedNodeMap node is null. If so, **TRUE** is returned. If not, **FALSE** is returned. The prototype of the DBE_XMLDOM.ISNULL function is as follows:

  ```
  DBE_XMLDOM.ISNULL(
    nnm       IN     DOMNAMEDNODEMAP)
  RETURN BOOLEAN;
  ```

  Checks whether the given DOMNode node is null. If so, **TRUE** is returned. If not, **FALSE** is returned. The prototype of the DBE_XMLDOM.ISNULL function is as follows:

  ```
  DBE_XMLDOM.ISNULL(
    n      IN     DOMNODE)
  RETURN BOOLEAN;
  ```

  Checks whether the given DOMNodeList node is null. If so, **TRUE** is returned. If not, **FALSE** is returned. The prototype of the DBE_XMLDOM.ISNULL function is as follows:

  ```
  DBE_XMLDOM.ISNULL(
    nl       IN     DOMNODELIST)
  RETURN BOOLEAN;
  ```

  Checks whether the given DOMText node is null. If so, **TRUE** is returned. If not, **FALSE** is returned. The prototype of the DBE_XMLDOM.ISNULL function is as follows:

  ```
  DBE_XMLDOM.ISNULL(
    t       IN     DOMTEXT)
  RETURN BOOLEAN;
  ```

Table 2-452 DBE_XMLDOM.ISNULL API parameters

Parameter	Description
a	Specified DOMAttr node.
doc	Specified DOMDocument node.
elem	Specified DOMElement node.
nnm	Specified DOMNamedNodeMap node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
n	Specified DOMNode node.
nl	Specified DOMNodeList node.
t	Specified DOMText node.

NOTE

Due to the implementation difference of DBE_XMLDOM.FREEDOCUMENT, an error is reported when the DBE_XMLDOM.ISNULL API calls the freed DOMDocument node.

Example:

```
-- 1. Use createAttribute to create a DOMAttr node and check whether the node is empty.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  attr DBE_XMLDOM.DOMATTR;
  buf VARCHAR2(1000);
BEGIN
  doc := DBE_xmldom.newdomdocument('<?xml version="1.0"?>
    <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)>
    <!ATTLIST note color CDATA #REQUIRED>
    <!ELEMENT to (#PCDATA)>
    <!ELEMENT from (#PCDATA)>
    <!ELEMENT heading (#PCDATA)>
    <!ELEMENT body (#PCDATA)>]>
    <note color="red"><to>Chinese</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don''t forget me this weekend!</body>
    </note>');
  attr := DBE_XMLDOM.CREATEATTRIBUTE (doc, 'length');
  if DBE_XMLDOM.ISNULL(attr) then
    DBE_OUTPUT.print_line('null');
  else
    DBE_OUTPUT.print_line('not null');
  end if;
END;
/
-- Expected result:
not null
ANONYMOUS BLOCK EXECUTE

-- 2. Declare (but not initialize) a DOMElement node and check whether the node is empty.
DECLARE
  docelem   DBE_XMLDOM.DOMElement;
BEGIN
  if DBE_XMLDOM.ISNULL(docelem) then
    DBE_OUTPUT.print_line('null');
  else
    DBE_OUTPUT.print_line('not null');
  end if;
END;
/
-- Expected result:
null
ANONYMOUS BLOCK EXECUTE

-- 3. Use newDomdocument to construct a good DOMDocument node and check whether the node is
empty.
Declare
 doc dbe_xmldom.domdocument;
BEGIN
 doc := DBE_xmldom.newdomdocument('<?xml version="1.0"?>
```

```
    <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)>
    <!ATTLIST note color CDATA #REQUIRED>
    <!ELEMENT to (#PCDATA)>
    <!ELEMENT from (#PCDATA)>
    <!ELEMENT heading (#PCDATA)>
    <!ELEMENT body (#PCDATA)>]>
    <note color="red"><to>Chinese</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don''t forget me this weekend!</body>
    </note>');
 if DBE_XMLDOM.ISNULL(doc) then
    DBE_OUTPUT.print_line('null');
  else
    DBE_OUTPUT.print_line('not null');
  end if;
END;
/
-- Expected result:
not null
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.ITEM

  Returns the element corresponding to the index in a list based on the index. The prototype of the DBE_XMLDOM.ITEM function is as follows:

```
DBE_XMLDOM.ITEM(
  nl IN DOMNODELIST,
  index IN NUMBER)
RETURN DOMNODE;
```

  Returns the element corresponding to the index in a map based on the index. The prototype of the DBE_XMLDOM.ITEM function is as follows:

```
DBE_XMLDOM.ITEM(
  nnm IN DOMNAMEDNODEMAP,
  index IN NUMBER)
RETURN DOMNODE;
```

Table 2-453 DBE_XMLDOM.ITEM API parameters

Parameter	Description
nl	Specified DOMNodeList object.
nnm	Specified DOMNamedNodeMap object.
index	Index of the element to be retrieved.

> **NOTE**
>
> For improper input parameters such as Boolean and CLOB, the item function of the map type points to the value of the first index by default.

Example:

```
-- 1. Return the element corresponding to the index in a map based on the index.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  elem DBE_XMLDOM.DOMElement;
  map DBE_XMLDOM.DOMNAMEDNODEMAP;
  node DBE_XMLDOM.DOMNODE;
  node2 DBE_XMLDOM.DOMNODE;
  buf varchar2(1000);
BEGIN
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  doc := dbe_xmldom.newdomdocument('<bookstore category="web" cover="paperback"><book
category="cooking">
      <title lang="en">Everyday Italian</title><author>Giada De Laurentiis</author>
      <year>2005</year><price>30.00</price></book></bookstore>');
  elem := DBE_XMLDOM.GETDOCUMENTELEMENT(doc);
  node := DBE_XMLDOM.MAKENODE(elem);
  map := DBE_XMLDOM.GETATTRIBUTES(DBE_XMLDOM.getFirstChild(node));
  node2:= DBE_XMLDOM.item(map,0);
  DBE_XMLDOM.writeToBuffer(node2, buf);
  dbe_output.print_line(buf);
  dbe_xmldom.freedocument(doc);
  RAISE NOTICE '%', buf;
END;
/
-- Expected result:
cooking
NOTICE:  cooking
ANONYMOUS BLOCK EXECUTE

-- 2. Return the element corresponding to the index in a list based on the index.
DECLARE
  doc dbe_xmldom.domdocument;
  node dbe_xmldom.domnode;
  node1 dbe_xmldom.domnode;
  nodelist DBE_XMLDOM.DOMNODELIST;
  len INTEGER;
  buffer1 varchar2(1010);
BEGIN
  doc := dbe_xmldom.newdomdocument('<bookstore category="web" cover="paperback"><book
category="cooking">
      <title lang="en">Everyday Italian</title><author>Giada De Laurentiis</author>
      <year>2005</year><price>30.00</price></book></bookstore>');
  node := dbe_xmldom.makenode(doc);
  node := dbe_xmldom.GETFIRSTCHILD(node);
  node := dbe_xmldom.GETFIRSTCHILD(node);
  nodelist := DBE_XMLDOM.GETCHILDNODES(node);
  len := DBE_XMLDOM.GETLENGTH(nodelist);
  RAISE NOTICE 'len :  %', len;
  node1 := DBE_XMLDOM.ITEM(nodelist, 0);
  IF DBE_XMLDOM.ISNULL(node1) THEN
      dbe_output.print_line('IS NULL');
  ELSE
      dbe_output.print_line('NOT NULL');
  END IF;
  dbe_xmldom.writetobuffer(node1, buffer1);
  dbe_output.print_line('buffer1: ');
  dbe_output.print_line(buffer1);
END;
/
-- Expected result:
NOTICE:  len :  4
NOT NULL
buffer1:
<title lang="en">Everyday Italian</title>
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.MAKEELEMENT

  Returns the DOMElement object after conversion. The prototype of the DBE_XMLDOM.MAKEELEMENT function is as follows:

```
DBE_XMLDOM.MAKEELEMENT(
  n IN DOMNODE)
RETURN DOMELEMENT;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-454 DBE_XMLDOM.MAKEELEMENT API parameters

Parameter	Description
n	Specified DOMNode object.

Example:

```
-- Forcibly convert the DOMNode node converted from the DOMElement type back to the
DOMElement type.
DECLARE
  buf VARCHAR2(1000);
  doc DBE_XMLDOM.DOMDocument;
  elem DBE_XMLDOM.DOMElement;
  elem2 DBE_XMLDOM.DOMElement;
  node DBE_XMLDOM.DOMNode;
BEGIN
  doc := DBE_XMLDOM.NEWDOMDOCUMENT();
  elem := DBE_XMLDOM.createElement(doc, 'aaa');
  node := DBE_XMLDOM.makeNode(elem);
  elem2 := DBE_XMLDOM.makeElement(node);
  buf := DBE_XMLDOM.GETNODENAME(DBE_XMLDOM.makeNode(elem2));
  DBE_OUTPUT.print_line(buf);
END;
/
-- Expected result:
aaa
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.MAKENODE

  Forcibly converts a specified DOMAttr node to a DOMNode node and returns the DOMNode node. The prototype of the DBE_XMLDOM.MAKENODE function is as follows:

  ```
  DBE_XMLDOM.MAKENODE(
    a        IN     DOMATTR)
   RETURN DOMNODE;
  ```

  Forcibly converts a specified DOMDocument node to a DOMNode node and returns the DOMNode node. The prototype of the DBE_XMLDOM.MAKENODE function is as follows:

  ```
  DBE_XMLDOM.MAKENODE(
    doc      IN     DOMDOCUMENT)
   RETURN DOMNODE;
  ```

  Forcibly converts a specified DOMElement node to a DOMNode node and returns the DOMNode node. The prototype of the DBE_XMLDOM.MAKENODE function is as follows:

  ```
  DBE_XMLDOM.MAKENODE(
    elem       IN     DOMELEMENT)
   RETURN DOMNODE;
  ```

  Forcibly converts a specified DOMText node to a DOMNode node and returns the DOMNode node. The prototype of the DBE_XMLDOM.MAKENODE function is as follows:

  ```
  DBE_XMLDOM.MAKENODE(
    t       IN     DOMTEXT)
   RETURN DOMNODE;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-455 DBE_XMLDOM.MAKENODE API parameters

Parameter	Description
a	Specified DOMAttr node.
doc	Specified DOMDocument node.
elem	Specified DOMElement node.
t	Specified DOMText node.

NOTE

Due to syntax restrictions, when DBE_XMLDOM.MAKENODE is used as the return value of a function, it cannot be directly implemented by running the following command:

```
return DBE_XMLDOM.MAKENODE(doc);
```

You are advised to run the following command:

```
tmp_node := DBE_XMLDOM.MAKENODE(doc );
return tmp_node;
```

Example:

```
-- 1. Use createattr to generate ATTR and convert it to a node.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  attr DBE_XMLDOM.DOMATTR;
  dom_node DBE_XMLDOM.DOMNode;
  buf VARCHAR2(1000);
BEGIN
  doc := DBE_xmldom.newdomdocument('<?xml version="1.0"?>
      <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)>
      <!ELEMENT to (#PCDATA)>
      <!ELEMENT from (#PCDATA)>
      <!ELEMENT heading (#PCDATA)>
      <!ELEMENT body (#PCDATA)>]>
      <note><to>Chinese</to>
      <from>Jani</from>
      <heading>Reminder</heading>
      <body>Don''t forget me this weekend!</body>
      </note>');
   attr := DBE_XMLDOM.CREATEATTRIBUTE (doc, 'length');
   dom_node := DBE_XMLDOM.makeNode(attr);
   buf := DBE_XMLDOM.getNodeName(dom_node);
   DBE_OUTPUT.print_line(buf);
END;
/
-- Expected result:
length
ANONYMOUS BLOCK EXECUTE

-- 2. Use the getdocumentelement function to generate an elem node and then perform makenode.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  root DBE_XMLDOM.DOMElement;
  attr DBE_XMLDOM.DOMATTR;
  node DBE_XMLDOM.DOMNODE;
  buf VARCHAR2(1000);
BEGIN
  doc := DBE_xmldom.newdomdocument('<?xml version="1.0"?>
      <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)>
      <!ATTLIST note color CDATA #REQUIRED>
      <!ELEMENT to (#PCDATA)>
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    <!ELEMENT from (#PCDATA)>
    <!ELEMENT heading (#PCDATA)>
    <!ELEMENT body (#PCDATA)>]>
    <note color="red"><to>Chinese</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don''t forget me this weekend!</body>
    </note>');
  root := DBE_XMLDOM.getDocumentElement(doc);
  node := DBE_XMLDOM.makenode(root);
  DBE_OUTPUT.print_line(DBE_XMLDOM.GETNODENAME(node));
END;
/
-- Expected result:
note
ANONYMOUS BLOCK EXECUTE

-- 3. Use newdomdocument to create a parameter of the DOMDocument type. The parameter is not
empty and is used as the input parameter of MAKENODE.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  buf VARCHAR2(1000);
  dom_node DBE_XMLDOM.DOMNODE;
BEGIN
  doc := DBE_xmldom.newdomdocument('<?xml version="1.0"?>
            <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)>
            <!ELEMENT to (#PCDATA)>
            <!ELEMENT from (#PCDATA)>
            <!ELEMENT heading (#PCDATA)>
            <!ELEMENT body (#PCDATA)>]>
     <note><to>Chinese</to>
            <from>Jani</from>
            <heading>Reminder</heading>
            <body>Don''t forget me this weekend!</body>
            </note>');
  DBE_OUTPUT.print_line('doc.id: ');
  DBE_OUTPUT.print_line(doc.id);
  dom_node := DBE_XMLDOM.makeNode(doc);
  DBE_OUTPUT.print_line('dom_node.id: ');
  DBE_OUTPUT.print_line(dom_node.id);
  buf := DBE_XMLDOM.GETNODENAME(dom_node);
  DBE_OUTPUT.print_line(buf);
END;
/
-- Expected result (if the XMLDOM API has been executed before the current session, the result is
uncertain):
doc.id:
19000000000000001B00000001
dom_node.id:
19000000010000001B00000001
#document
ANONYMOUS BLOCK EXECUTE

-- 4. Declare (but not initialize) a DOMText variable, and use it as the input parameter of MAKENODE.
DECLARE
  text DBE_XMLDOM.DOMTEXT;
  buf VARCHAR2(1000);
  dom_node DBE_XMLDOM.DOMNODE;
BEGIN
  dom_node := DBE_XMLDOM.makeNode(text);
  buf := DBE_XMLDOM.GETNODENAME(dom_node);
  DBE_OUTPUT.print_line(buf);
END;
/
-- Expected result:
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.NEWDOMDOCUMENT

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Returns a new DOMDocument object. The prototype of the DBE_XMLDOM.NEWDOMDOCUMENT function is as follows:

```
DBE_XMLDOM.NEWDOMDOCUMENT
RETURN DOMDOCUMENT;
```

Returns a new DOMDocument instance object created from the specified XMLType type. The prototype of the DBE_XMLDOM.NEWDOMDOCUMENT function is as follows:

```
DBE_XMLDOM.NEWDOMDOCUMENT(
  xmldoc    IN SYS.XMLTYPE)
RETURN DOMDOCUMENT;
```

Returns a new DOMDocument instance object created from the specified CLOB type. The prototype of the DBE_XMLDOM.NEWDOMDOCUMENT function is as follows:

```
DBE_XMLDOM.NEWDOMDOCUMENT(
  cl       IN    CLOB)
RETURN DOMDOCUMENT;
```

Table 2-456 DBE_XMLDOM.NEWDOMDOCUMENT API parameters

Parameter	Description
xmldoc	Specified XMLType type.
cl	Specified CLOB type.

NOTE

- The size of the input parameter must be less than 1 GB.
- Currently, external DTD parsing is not supported.
- The document created by NEWDOMDOCUMENT uses the UTF-8 character set by default.
- Each document parsed from the same XMLType instance is independent, and the modification of the document does not affect the XMLType.
- For details about the differences between our database and database ORA, see **DBE_XMLPARSER.PARSECLOB**.

Example:

```
-- 1. Return a new DOMDocument object.
DECLARE
  doc dbe_xmldom.domdocument;
  buffer varchar2(1010);
BEGIN
  doc := dbe_xmldom.newdomdocument();
  dbe_xmldom.setdoctype(doc, 'note', 'sysid', 'pubid');
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
  dbe_xmldom.freedocument(doc);
END;
/
-- Expected result:
buffer:
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note PUBLIC "pubid" "sysid">

ANONYMOUS BLOCK EXECUTE

-- 2. Return a new DOMDocument instance object created from the specified CLOB type.
DECLARE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  doc dbe_xmldom.domdocument;
  buffer varchar2(1010);
BEGIN
  doc := dbe_xmldom.newdomdocument('<?xml version="1.0"?>
     <note><to>test</to><from>Jani</from><heading>Reminder</heading>
     <body>Don''t forget me this weekend!</body></note>');
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
  dbe_xmldom.freedocument(doc);
END;
/
-- Expected result:
buffer:
<?xml version="1.0" encoding="UTF-8"?>
<note>
 <to>test</to>
 <from>Jani</from>
 <heading>Reminder</heading>
 <body>Don't forget me this weekend!</body>
</note>

ANONYMOUS BLOCK EXECUTE

-- 3. Return a new DOMDocument instance object created from the specified XMLType type.
DECLARE
doc dbe_xmldom.domdocument;
  xt xmltype;
  buffer varchar2(1010);
BEGIN
  xt := xmltype('<h:data xmlns:h="http://www.w3.org/TR/html4/">
    <h:da1 len="10">test namespace</h:da1>
    <h:da1>bbbbbbbbbb</h:da1>
    </h:data>');
  doc := dbe_xmldom.newdomdocument(xt);
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
  dbe_xmldom.freedocument(doc);
END;
/
-- Expected result:
buffer:
<?xml version="1.0" encoding="UTF-8"?>
<h:data xmlns:h="http://www.w3.org/TR/html4/">
 <h:da1 len="10">test namespace</h:da1>
 <h:da1>bbbbbbbbbb</h:da1>
</h:data>

ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.SETATTRIBUTE

  Sets the value of the DOMElement attribute by name. The prototype of the DBE_XMLDOM.SETATTRIBUTE function is as follows:

```
DBE_XMLDOM.SETATTRIBUTE(
  elem    IN  DOMELEMENT,
  name    IN  VARCHAR2,
  value   IN  VARCHAR2);
```

  Sets the attribute values of a DOMElement object by name and namespace URI. The prototype of the DBE_XMLDOM.SETATTRIBUTE function is as follows:

```
DBE_XMLDOM.SETATTRIBUTE(
  elem    IN  DOMELEMENT,
  name    IN  VARCHAR2,
  value   IN  VARCHAR2,
  ns      IN  VARCHAR2);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-457 DBE_XMLDOM.SETATTRIBUTE API parameters

Parameter	Description
elem	Specified DOMElement node.
name	Attribute name.
value	Attribute value.
ns	Namespace.

NOTE

Multiple attributes can be added through the DBE_XMLDOM.SETATTRIBUTE API. The attribute name cannot be null, and attributes with the same name cannot exist in the same DOMElement node. If you want to add attributes with the same name, you should explicitly set a namespace for each attribute with the same name, but you are advised not to perform such operations. If an attribute exists in a namespace, the specified namespace must be displayed when you modify the attribute. Otherwise, the attribute with the same name is added.

Example:

```
-- 1. Set the value of the DOMElement attribute by name.
DECLARE
  doc dbe_xmldom.domdocument;
  elem dbe_xmldom.domelement;
  docnode DBE_XMLDOM.DOMNode;
  buffer varchar2(1010);
  value  varchar(1000);
BEGIN
  doc := dbe_xmldom.newDOMDocument();
  elem := DBE_XMLDOM.CREATEELEMENT(doc, 'root');
  DBE_XMLDOM.setattribute(elem, 'len', '50cm');
  docnode := DBE_XMLDOM.appendChild(DBE_XMLDOM.makeNode(doc),
DBE_XMLDOM.makeNode(elem));
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
END;
/
-- Expected result:
buffer:
<?xml version="1.0" encoding="UTF-8"?>
<root len="50cm"/>

ANONYMOUS BLOCK EXECUTE

-- 2. Set the attribute values of a DOMElement object by name and namespace URI.
DECLARE
  doc dbe_xmldom.domdocument;
  elem dbe_xmldom.domelement;
  docnode DBE_XMLDOM.DOMNode;
  buffer varchar2(1010);
  value  varchar(1000);
begin
  doc := dbe_xmldom.newDOMDocument();
  elem := DBE_XMLDOM.CREATEELEMENT(doc, 'root');
  DBE_XMLDOM.setattribute(elem, 'len', '50cm', 'www.xxxxx.com');
  docnode := DBE_XMLDOM.appendChild(DBE_XMLDOM.makeNode(doc),
DBE_XMLDOM.makeNode(elem));
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
END;
/
-- Expected result:
buffer:
<?xml version="1.0" encoding="UTF-8"?>
<root len="50cm"/>

ANONYMOUS BLOCK EXECUTE

-- 3. Change the values of the DOMElement attributes by name.
DECLARE
  doc dbe_xmldom.domdocument;
  elem dbe_xmldom.domelement;
  docnode DBE_XMLDOM.DOMNode;
  buffer varchar2(1010);
  value  varchar(1000);
BEGIN
  doc := dbe_xmldom.newDOMDocument();
  elem := DBE_XMLDOM.CREATEELEMENT(doc, 'root');
  DBE_XMLDOM.setattribute(elem, 'len', '50cm');
  DBE_XMLDOM.setattribute(elem, 'len', '55cm');
  docnode := DBE_XMLDOM.appendChild(DBE_XMLDOM.makeNode(doc),
DBE_XMLDOM.makeNode(elem));
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
END;
/
-- Expected result:
buffer:
<?xml version="1.0" encoding="UTF-8"?>
<root len="55cm"/>

ANONYMOUS BLOCK EXECUTE

-- 4. Change the values of the DOMElement attributes by name and namespace URI.
DECLARE
  doc dbe_xmldom.domdocument;
  elem dbe_xmldom.domelement;
  docnode DBE_XMLDOM.DOMNode;
  buffer varchar2(1010);
  value  varchar(1000);
begin
  doc := dbe_xmldom.newDOMDocument();
  elem := DBE_XMLDOM.CREATEELEMENT(doc, 'root');
  DBE_XMLDOM.setattribute(elem, 'len', '50cm', 'www.xxxxx.com');
  DBE_XMLDOM.setattribute(elem, 'len', '55cm', 'www.xxxxx.com');
  docnode := DBE_XMLDOM.appendChild(DBE_XMLDOM.makeNode(doc),
DBE_XMLDOM.makeNode(elem));
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
END;
/
-- Expected result:
buffer:
<?xml version="1.0" encoding="UTF-8"?>
<root len="55cm"/>

ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.SETCHARSET

  Sets the character set for a DOMDocument object. The prototype of the DBE_XMLDOM.SETCHARSET function is as follows:

```
DBE_XMLDOM.SETCHARSET(
  doc      IN     DOMDocument,
  charset  IN     VARCHAR2);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-458 DBE_XMLDOM.SETCHARSET API parameters

Parameter	Description
doc	Specified DOMDocument node.
charset	Character set.

NOTE

- The value of **charset** contains a maximum of 60 bytes.
- Currently, the following character sets are supported: UTF-8, UCS-4, UCS-2, ISO-8859-1, ISO-8859-2, ISO-8859-3, ISO-8859-4, ISO-8859-5, ISO-8859-6, ISO-8859-7, ISO-8859-8, ISO-8859-9, ISO-2022-JP, Shift_JIS, EUC-JP and ASCII. If you enter other character sets, an error is reported or garbled characters may be displayed.

Example:

```
-- Set the UTF-8 character set for the DOC tree and print the DOC tree to the buffer.
DECLARE
  doc dbe_xmldom.domdocument;
  buffer varchar2(1010);
BEGIN
  doc := dbe_xmldom.newdomdocument('<?xml version="1.0"?>
      <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)><!ELEMENT to (#PCDATA)>
      <!ELEMENT from (#PCDATA)><!ELEMENT heading (#PCDATA)><!ELEMENT body (#PCDATA)>]>
      <note><to>test</to><from>Jani</from><heading>Reminder</heading>
      <body>Don''t forget me this weekend!</body></note>');
  dbe_xmldom.setcharset(doc, 'utf-8');
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
  dbe_xmldom.freedocument(doc);
END;
/
-- Expected result:
buffer:
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note [
<!ELEMENT note (to , from , heading , body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<note>
  <to>test</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>

ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.SETDOCTYPE

  Sets the external DTD of a DOMDocument object. The prototype of the DBE_XMLDOM.SETDOCTYPE function is as follows:

```
DBE_XMLDOM.SETDOCTYPE(
  doc     IN   DOMDocument,
  name    IN   VARCHAR2,
  sysid   IN   VARCHAR2,
  pubid   IN   VARCHAR2);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-459 DBE_XMLDOM.SETDOCTYPE API parameters

Parameter	Description
doc	Specified DOMDocument node.
name	Name of the DOCType to be initialized.
sysid	ID of the system whose DOCType needs to be initialized.
pubid	Public ID of the DOCType to be initialized.

NOTE

The total length of **name**, **sysid**, and **pubid** cannot exceed 32500 bytes.

Example:

```
-- After the initial system ID, public ID, and name are set for the external DTD of the DOMDocument, the DOC tree modified each time is output to the buffer.
DECLARE
  doc dbe_xmldom.domdocument;
  buffer varchar2(1010);
begin
  doc := dbe_xmldom.newdomdocument('<?xml version="1.0"?>
      <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)><!ELEMENT to (#PCDATA)>
      <!ELEMENT from (#PCDATA)><!ELEMENT heading (#PCDATA)><!ELEMENT body (#PCDATA)>]>
      <note><to>test</to><from>Jani</from><heading>Reminder</heading>
      <body>Don''t forget me this weekend!</body></note>');
  dbe_xmldom.setdoctype(doc, 'note', 'sysid', 'pubid');
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
  dbe_output.print_line('-------------------------------------------------');
  dbe_xmldom.setdoctype(doc, 'n0te', NULL, '');
  dbe_xmldom.setdoctype(doc, 'n0t1e', NULL, '');
  dbe_xmldom.writetobuffer(doc, buffer);
  dbe_output.print_line('buffer: ');
  dbe_output.print_line(buffer);
  dbe_xmldom.freedocument(doc);
END;
/
-- Expected result:
buffer:
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note PUBLIC "pubid" "sysid" [
<!ELEMENT note (to , from , heading , body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<note>
  <to>test</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>

-------------------------------------------------
buffer:
<?xml version="1.0" encoding="UTF-8"?>
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
<!DOCTYPE n0t1e [
<!ELEMENT note (to , from , heading , body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<note>
  <to>test</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>

ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.SETNODEVALUE

  Sets the value of a node in the DOMNode object. The prototype of the DBE_XMLDOM.SETNODEVALUE function is as follows:

```
DBE_XMLDOM.SETNODEVALUE(
 n IN DOMNODE,
 nodeValue IN VARCHAR2);
```

Table 2-460 DBE_XMLDOM.SETNODEVALUE API parameters

Parameter	Description
n	Specified DOMNode object.
nodeValue	String to be set in the DOMNode object.

NOTE

- You can enter an empty string or null value for **nodeValue**, but the node value will not be changed.
- **nodeValue** supports the escape character '&', which is automatically escaped during serialization.
- The default maximum length of **nodeValue** is restricted by the VARCHAR2 type and is 32767 bytes. If the length exceeds 32767 bytes, an exception is thrown.

Example:

```
-- After setting a node value different from the initial value for the DOMNode node that is converted from DOMText, obtain and output the node value.
DECLARE
  buf VARCHAR2(1000);
  doc DBE_XMLDOM.DOMDocument;
  text DBE_XMLDOM.DOMText;
  elem2 DBE_XMLDOM.DOMElement;
  node DBE_XMLDOM.DOMNode;
BEGIN
  doc := DBE_XMLDOM.NEWDOMDOCUMENT();
  text := DBE_XMLDOM.createTextNode(doc, 'aaa');
  DBE_XMLDOM.SETNODEVALUE(DBE_XMLDOM.makeNode(text), 'ccc');
  buf := DBE_XMLDOM.GETNODEVALUE(DBE_XMLDOM.makeNode(text));
  DBE_OUTPUT.print_line(buf);
END;
/
-- Expected result:
ccc
ANONYMOUS BLOCK EXECUTE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- DBE_XMLDOM.WRITETOBUFFER

  Writes an XML node to a specified buffer using the database character set. The prototype of the DBE_XMLDOM.WRITETOBUFFER function is as follows:

```
DBE_XMLDOM.WRITETOBUFFER(
  doc      IN      DOMDOCUMENT,
  buffer   INOUT  VARCHAR2);
```

  Writes an XML document to a specified buffer using the database character set. The prototype of the DBE_XMLDOM.WRITETOBUFFER function is as follows:

```
DBE_XMLDOM.WRITETOBUFFER(
  n        IN      DOMNODE,
  buffer   INOUT  VARCHAR2);
```

Table 2-461 DBE_XMLDOM.WRITETOBUFFER API parameters

Parameter	Description
doc	Specified DOMDocument node.
buffer	Buffer for the write operation.
n	Specified DOMNode node.

NOTE

- The buffer to which the writetobuffer function writes is less than 1 GB.
- This function adds content such as indentation to format the output. The output document will contain the XML declaration version and encoding.
- By default, XML files are output in the UTF-8 character set.

Example:

```
-- 1. Enter a parameter of the DOMNode type.
DECLARE
   doc dbe_xmldom.domdocument;
   elem DBE_XMLDOM.DOMELEMENT;
   buf varchar2(1000);
BEGIN
    doc := dbe_xmldom.newdomdocument();
    elem := dbe_xmldom.createelement(doc,'elem');
    DBE_XMLDOM.WRITETOBUFFER(dbe_xmldom.makenode(elem), buf);
    DBE_OUTPUT.print_line(buf);
END;
/
-- Expected result:
<elem/>
ANONYMOUS BLOCK EXECUTE

-- 2. Enter a parameter of the DOMDocument type.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  buf VARCHAR2(1000);
BEGIN
  doc := dbe_xmldom.newdomdocument('<?xml version="1.0"?>
      <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)><!ELEMENT to (#PCDATA)>
      <!ELEMENT from (#PCDATA)><!ELEMENT heading (#PCDATA)><!ELEMENT body (#PCDATA)>]>
      <note><to>test</to><from>Jani</from><heading>Reminder</heading>
      <body>Don''t forget me this weekend!</body></note>');
  DBE_XMLDOM.WRITETOBUFFER(doc, buf);
  DBE_OUTPUT.print_line('doc: ');
  DBE_OUTPUT.print_line(buf);
  DBE_XMLDOM.FREEDOCUMENT(doc);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
END;
/
-- Expected result:
doc:
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note [
<!ELEMENT note (to , from , heading , body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<note>
  <to>test</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>

ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.WRITETOCLOB

  Writes an XML node to a specified CLOB using the database character set. The prototype of the DBE_XMLDOM.WRITETOCLOB function is as follows:

```
DBE_XMLDOM.WRITETOCLOB(
  doc     IN      DOMDOCUMENT,
  cl      INOUT  CLOB);
```

  Writes an XML node to a specified CLOB using the database character set. The prototype of the DBE_XMLDOM.WRITETOCLOB function is as follows:

```
DBE_XMLDOM.WRITETOCLOB(
  n       IN      DOMNODE,
  cl      INOUT  CLOB);
```

Table 2-462 DBE_XMLDOM.WRITETOCLOB API parameters

Parameter	Description
doc	Specified DOMDocument node.
cl	CLOB to be written.
n	Specified DOMNode node.

NOTE

- The **doc** parameter is an input parameter. The CLOB to which the writetoclob function writes is less than 1 GB.
- This function adds content such as indentation to format the output. The output document will contain the XML declaration version and encoding.
- By default, XML files are output in the UTF-8 character set.

Example:

```
-- 1. Enter a parameter of the DOMNode type.
DECLARE
  CL  CLOB;
  N   DBE_XMLDOM.DOMNODE;
BEGIN
  DBE_XMLDOM.WRITETOCLOB(N, CL);
  DBE_OUTPUT.PRINT_LINE(CL);
END;
/
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Expected result:
ANONYMOUS BLOCK EXECUTE

-- 2. Enter a parameter of the DOMDocument type.
DECLARE
    doc dbe_xmldom.domdocument;
    mclob clob;
BEGIN
    doc := dbe_xmldom.newdomdocument('<?xml version="1.0"?>
       <!DOCTYPE note [<!ELEMENT note (to,from,heading,body)><!ELEMENT to (#PCDATA)>
       <!ELEMENT from (#PCDATA)><!ELEMENT heading (#PCDATA)><!ELEMENT body (#PCDATA)>]>
       <note><to>test</to><from>Jani</from><heading>Reminder</heading>
       <body>Don''t forget me this weekend!</body></note>');
    dbe_xmldom.writetoclob(doc, mclob);
    dbe_output.print_line('mclob: ');
    dbe_output.print_line(mclob);
    dbe_xmldom.freedocument(doc);
END;
/
-- Expected result:
mclob:
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note [
<!ELEMENT note (to , from , heading , body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<note>
  <to>test</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>

ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.WRITETOFILE

  Writes an XML node to a specified file using the database character set. The prototype of the DBE_XMLDOM.WRITETOFILE function is as follows:

  ```
  DBE_XMLDOM.WRITETOCLOB(
    doc     IN      DOMDOCUMENT,
    fileName   IN      VARCHAR2);
  ```

  Writes an XML node to a specified file using the database character set. The prototype of the DBE_XMLDOM.WRITETOFILE function is as follows:

  ```
  DBE_XMLDOM.WRITETOCLOB(
    n       IN      DOMNODE,
    fileName   IN      VARCHAR2);
  ```

  Writes an XML document to a specified file using the specified character set. The prototype of the DBE_XMLDOM.WRITETOFILE function is as follows:

  ```
  DBE_XMLDOM.WRITETOCLOB(
    doc     IN      DOMDOCUMENT,
    fileName   IN      VARCHAR2,
    charset   IN   VARCHAR2);
  ```

  Writes an XML document to a specified file using the specified character set. The prototype of the DBE_XMLDOM.WRITETOFILE function is as follows:

  ```
  DBE_XMLDOM.WRITETOCLOB(
    n       IN      DOMNODE,
    fileName   IN      VARCHAR2,
    charset   IN   VARCHAR2);
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-463 DBE_XMLDOM.WRITETOFILE API parameters

Parameter	Description
doc	Specified DOMDocument node.
fileName	File to be written.
n	Specified DOMNode node.
charset	Specified character set.

NOTE

- The **doc** parameter is an input parameter. The value of **filename** can contain a maximum of 255 bytes, and the value of **charset** can contain a maximum of 60 bytes. For details about the character sets supported by **charset**, see the **DBE_XMLDOM.SETCHARSET** API.
- This function adds content such as indentation to format the output. The output document will contain the XML declaration version and encoding.
- If newdomdocument() is used to create a document without parameters, no error is reported when **charset** is not specified. The UTF-8 character set is used by default.
- The **filename** must be in the path created in **pg_directory**. The backslash (\) in the filename will be converted to a slash (/). Only one slash (/) is allowed. The file name must be in the **pg_directory_name/file_name.xml** format. The output file must be in the XML format.
- When the GUC parameter **safe_data_path** is enabled, you can only use an advanced package to read and write files in the file path specified by **safe_data_path**.
- Before creating a directory, ensure that the directory exists in the OS and the user has the read and write permissions on the directory. For details about how to create a directory, see **CREATE DIRECTORY**.

Example:

```
-- Before creating a directory, ensure that the directory exists in the OS and the user has the read and write permissions on the directory.
create directory dir as '/tmp';
-- 1. Write an XML node to a specified file using the database character set.
DECLARE
  FPATH VARCHAR2(1000);
  DOC   DBE_XMLDOM.DOMDOCUMENT;
BEGIN
  DOC := DBE_XMLDOM.NEWDOMDOCUMENT('<ROOT>
    <A ATTR1="A_VALUE">
        <ACHILD>ACHILD TXT</ACHILD>
    </A>
    <B>B TXT</B>
    <C/>
  </ROOT>');
  FPATH := 'dir/simplexml.xml';
  DBE_XMLDOM.WRITETOFILE(DOC, FPATH);
END;
/
-- Expected result:
ANONYMOUS BLOCK EXECUTE

-- 2. Write an XML document to a specified file using the specified character set.
DECLARE
  SRC   VARCHAR(1000);
  FPATH VARCHAR2(1000);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  DOC   DBE_XMLDOM.DOMDOCUMENT;
  ELE   DBE_XMLDOM.DOMELEMENT;
BEGIN
  FPATH := 'dir/simplexml.xml';
  SRC := '<ROOT>
    <A ATTR1="A_VALUE">
        <ACHILD>ACHILD TXT</ACHILD>
    </A>
    <B>B TXT</B>
    <C/>
  </ROOT>';
  DOC := DBE_XMLDOM.NEWDOMDOCUMENT(SRC);
  ELE := DBE_XMLDOM.GETDOCUMENTELEMENT(DOC);
  DBE_XMLDOM.WRITETOFILE(DBE_XMLDOM.MAKENODE(ELE), FPATH, 'ASCII');
  DBE_XMLDOM.FREEDOCUMENT(DOC);
END;
/
-- Expected result:
ANONYMOUS BLOCK EXECUTE

-- Clean up the environment.
drop directory dir;
```

- DBE_XMLDOM.GETSESSIONTREENUM

  Queries the number of DOM trees of all types in the current session. The prototype of the DBE_XMLDOM.GETSESSIONTREENUM function is as follows:

  ```
  DBE_XMLDOM.GETSESSIONTREENUM()
  RETURN INTEGER;
  ```

  NOTE

  For DOM trees that have used FREEElement and FREENode, this function still counts them.

  Example:

  ```
  -- Create three documents and obtain the number of DOM trees in the current session.
  DECLARE
    doc DBE_XMLDOM.DOMDocument;
    doc2 DBE_XMLDOM.DOMDocument;
    doc3 DBE_XMLDOM.DOMDocument;

    buffer varchar2(1010);
  BEGIN
  -- Create three documents.
    doc := DBE_XMLDOM.NEWDOMDOCUMENT('<?xml version="1.0"?>
  <root>
     <elem1 attr="attrtest">
        <elem2>Im text</elem2>
        <elem3>Im text too</elem3>
     </elem1>
     <elem4>Text</elem4>
  </root>
  ');
    doc2 := DBE_XMLDOM.NEWDOMDOCUMENT('<?xml version="1.0"?>
  <computer size="ITX" price="19999">
     <cpu>Ryzen 9 3950X</cpu>
     <ram>32GBx2 DDR4 3200MHz</ram>
     <motherboard>ROG X570i</motherboard>
     <gpu>RTX2070 Super</gpu>
     <ssd>1TB NVMe Toshiba + 2TB NVMe WD Black</ssd>
     <hdd>12TB WD Digital</hdd>
     <psu>CORSAIR SF750</psu>
     <case>LIANLI TU150</case>
  </computer>
  ');
    doc3 := DBE_XMLDOM.NEWDOMDOCUMENT('<?xml version="1.0"?>
  <bookstore>
     <book genre="autobiography" publicationdate="1981" ISBN="1-861003-11-0">
        <title>The Autobiography of Benjamin Franklin</title>
  ```

```
      <author>
         <first-name>Benjamin</first-name>
         <last-name>Franklin</last-name>
      </author>
      <price>8.99</price>
   </book>
   <book genre="novel" publicationdate="1967" ISBN="0-201-63361-2">
      <title>The Confidence Man</title>
      <author>
         <first-name>Herman</first-name>
         <last-name>Melville</last-name>
      </author>
      <price>11.99</price>
   </book>
   <book genre="philosophy" publicationdate="1991" ISBN="1-861001-57-6">
      <title>The Gorgias</title>
      <author>
         <name>Plato</name>
      </author>
      <price>9.99</price>
   </book>
</bookstore>
');
-- Print IDs.
  DBE_OUTPUT.PRINT_LINE(doc.id);
  DBE_OUTPUT.PRINT_LINE(doc2.id);
  DBE_OUTPUT.PRINT_LINE(doc3.id);
-- Call functions and print them.
  DBE_OUTPUT.PRINT_LINE(DBE_XMLDOM.GETSESSIONTREENUM());
-- Release the documents.
  DBE_XMLDOM.FREEDOCUMENT(doc);
  DBE_XMLDOM.FREEDOCUMENT(doc2);
  DBE_XMLDOM.FREEDOCUMENT(doc3);
END;
/
-- Expected result (if the XMLDOM API has been executed before the current session, the result is
uncertain):
00000000000000000200000001
01000000000000000300000001
02000000000000000400000001
3
ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETDOCTREESINFO

  Queries the DOM tree information of the document type in the current session, such as the memory usage. The prototype of the DBE_XMLDOM.GETDOCTREESINFO function is as follows:

  ```
  DBE_XMLDOM.GETDOCTREESINFO()
  RETURN VARCHAR2;
  ```

  NOTE

  This function collects statistics only on DOM tree nodes of the document type.

  Example:

  ```
  -- Create three documents and obtain the information about the document tree in the current session.
  DECLARE
    doc DBE_XMLDOM.DOMDocument;
    doc2 DBE_XMLDOM.DOMDocument;
    doc3 DBE_XMLDOM.DOMDocument;

    buffer varchar2(1010);
  BEGIN
  -- Create three documents.
    doc := DBE_XMLDOM.NEWDOMDOCUMENT('<?xml version="1.0"?>
  <root>
     <elem1 attr="attrtest">
        <elem2>Im text</elem2>
        <elem3>Im text too</elem3>
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    </elem1>
    <elem4>Text</elem4>
</root>
');
  doc2 := DBE_XMLDOM.NEWDOMDOCUMENT('<?xml version="1.0"?>
<computer size="ITX" price="19999">
    <cpu>Ryzen 9 3950X</cpu>
    <ram>32GBx2 DDR4 3200MHz</ram>
    <motherboard>ROG X570i</motherboard>
    <gpu>RTX2070 Super</gpu>
    <ssd>1TB NVMe Toshiba + 2TB NVMe WD Black</ssd>
    <hdd>12TB WD Digital</hdd>
    <psu>CORSAIR SF750</psu>
    <case>LIANLI TU150</case>
</computer>
');
  doc3 := DBE_XMLDOM.NEWDOMDOCUMENT('<?xml version="1.0"?>
<bookstore>
    <book genre="autobiography" publicationdate="1981" ISBN="1-861003-11-0">
        <title>The Autobiography of Benjamin Franklin</title>
        <author>
            <first-name>Benjamin</first-name>
            <last-name>Franklin</last-name>
        </author>
        <price>8.99</price>
    </book>
    <book genre="novel" publicationdate="1967" ISBN="0-201-63361-2">
        <title>The Confidence Man</title>
        <author>
            <first-name>Herman</first-name>
            <last-name>Melville</last-name>
        </author>
        <price>11.99</price>
    </book>
    <book genre="philosophy" publicationdate="1991" ISBN="1-861001-57-6">
        <title>The Gorgias</title>
        <author>
            <name>Plato</name>
        </author>
        <price>9.99</price>
    </book>
</bookstore>
');
-- Print IDs.
  DBE_OUTPUT.PRINT_LINE(doc.id);
  DBE_OUTPUT.PRINT_LINE(doc2.id);
  DBE_OUTPUT.PRINT_LINE(doc3.id);
-- Call functions and print them.
  DBE_OUTPUT.PRINT_LINE(DBE_XMLDOM.GETDOCTREESINFO());
-- Release the documents.
  DBE_XMLDOM.FREEDOCUMENT(doc);
  DBE_XMLDOM.FREEDOCUMENT(doc2);
  DBE_XMLDOM.FREEDOCUMENT(doc3);
END;
/
-- Expected result (if the XMLDOM API has been executed before the current session, the result is
uncertain):
00000000000000000200000001
01000000000000000300000001
02000000000000000400000001
|ID:00000000000000000200000001   |Node count:11   |Memory used:151 byte   |
|ID:01000000000000000300000001   |Node count:22   |Memory used:322 byte   |
|ID:02000000000000000400000001   |Node count:48   |Memory used:654 byte   |

ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETDETAILDOCTREEINFO

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Queries the number of subnodes of each type in the specified document. The prototype of the DBE_XMLDOM.GETDETAILDOCTREEINFO function is as follows:

```
DBE_XMLDOM.GETDETAILDOCTREEINFO(
   doc     IN      DOMDOCUMENT
)
RETURN VARCHAR2;
```

Table 2-464 DBE_XMLDOM.GETDETAILDOCTREEINFO API parameters

Parameter	Description
doc	Specified DOMDocument node.

NOTE

This function collects statistics only on DOM tree nodes of the document type.

Example:

```
-- Create three documents and use this function to obtain the number of nodes of each type in each
document.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  doc2 DBE_XMLDOM.DOMDocument;
  doc3 DBE_XMLDOM.DOMDocument;

  buffer varchar2(1010);
BEGIN
-- Create three documents.
  doc := DBE_XMLDOM.NEWDOMDOCUMENT('<?xml version="1.0"?>
<root>
   <elem1 attr="attrtest">
      <elem2>Im text</elem2>
      <elem3>Im text too</elem3>
   </elem1>
   <elem4>Text</elem4>
</root>
');
  doc2 := DBE_XMLDOM.NEWDOMDOCUMENT('<?xml version="1.0"?>
<computer size="ITX" price="19999">
   <cpu>Ryzen 9 3950X</cpu>
   <ram>32GBx2 DDR4 3200MHz</ram>
   <motherboard>ROG X570i</motherboard>
   <gpu>RTX2070 Super</gpu>
   <ssd>1TB NVMe Toshiba + 2TB NVMe WD Black</ssd>
   <hdd>12TB WD Digital</hdd>
   <psu>CORSAIR SF750</psu>
   <case>LIANLI TU150</case>
</computer>
');
  doc3 := DBE_XMLDOM.NEWDOMDOCUMENT('<?xml version="1.0"?>
<bookstore>
   <book genre="autobiography" publicationdate="1981" ISBN="1-861003-11-0">
      <title>The Autobiography of Benjamin Franklin</title>
      <author>
         <first-name>Benjamin</first-name>
         <last-name>Franklin</last-name>
      </author>
      <price>8.99</price>
   </book>
   <book genre="novel" publicationdate="1967" ISBN="0-201-63361-2">
      <title>The Confidence Man</title>
      <author>
         <first-name>Herman</first-name>
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
        <last-name>Melville</last-name>
      </author>
      <price>11.99</price>
   </book>
   <book genre="philosophy" publicationdate="1991" ISBN="1-861001-57-6">
      <title>The Gorgias</title>
      <author>
         <name>Plato</name>
      </author>
      <price>9.99</price>
   </book>
</bookstore>
');
-- Print IDs.
  DBE_OUTPUT.PRINT_LINE(doc.id);
  DBE_OUTPUT.PRINT_LINE(doc2.id);
  DBE_OUTPUT.PRINT_LINE(doc3.id);
-- Call functions and print them.
  buffer := DBE_XMLDOM.GETDETAILDOCTREEINFO(doc);
  DBE_OUTPUT.PRINT_LINE(buffer);
  buffer := DBE_XMLDOM.GETDETAILDOCTREEINFO(doc2);
  DBE_OUTPUT.PRINT_LINE(buffer);
  buffer := DBE_XMLDOM.GETDETAILDOCTREEINFO(doc3);
  DBE_OUTPUT.PRINT_LINE(buffer);
-- Release the documents.
  DBE_XMLDOM.FREEDOCUMENT(doc);
  DBE_XMLDOM.FREEDOCUMENT(doc2);
  DBE_XMLDOM.FREEDOCUMENT(doc3);
END;
/
-- Expected result (if the XMLDOM API has been executed before the current session, the result is
uncertain):
000000000000000000200000001
010000000000000000300000001
020000000000000000400000001
|ID:000000000000000000200000001    |Element count:5    |Attribute count:1    |Text count:4    |
|ID:010000000000000000300000001    |Element count:9    |Attribute count:2    |Text count:10    |
|ID:020000000000000000400000001    |Element count:18    |Attribute count:9    |Text count:20    |

ANONYMOUS BLOCK EXECUTE
```

- DBE_XMLDOM.GETELEMENTSBYTAGNAM

  Searches for the DOMNodelist node by name in the XML file and returns the node. The prototype of the DBE_XMLDOM.GETELEMENTSBYTAGNAME function is:

  ```
  DBE_XMLDOM.GETELEMENTSBYTAGNAME(
    doc        IN      DOMDOCUMENT,
    tagname    IN      VARCHAR2)
  RETURN DOMNODELIST;
  ```

  Searches for the DOMNodelist node by name in the XML file and returns the node. The prototype of the DBE_XMLDOM.GETELEMENTSBYTAGNAME function is:

  ```
  DBE_XMLDOM.GETELEMENTSBYTAGNAME(
    elem       IN      DOMELEMENT,
    tagname       IN      VARCHAR2)
  RETURN DOMNODELIST;
  ```

  Searches for the DOMNodelist node by name and namespace in the XML file and returns the node. The prototype of the DBE_XMLDOM.GETELEMENTSBYTAGNAME function is:

  ```
  DBE_XMLDOM.GETELEMENTSBYTAGNAME(
    elem       IN      DOMELEMENT,
    tagname       IN      VARCHAR2,
    ns         IN      VARCHAR2)
   RETURN DOMNODELIST;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-465 DBE_XMLDOM.GETELEMENTSBYTAGNAME API parameters

Parameter	Description
doc	Specified DOMDocument node.
elem	Specified DOMElement node.
tagname	Tag name. Using the wildcard (*) will match any tag.
ns	Namespace. Using the wildcard (*) will match any namespace.

Example:

```
-- 1. Search the DOMDocument node by TAGNAME and return the matched DOMNodelist node list.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  root_elem DBE_XMLDOM.DOMElement;
  child_node DBE_XMLDOM.DOMNODE;
  node_list DBE_XMLDOM.DOMNODELIST;
  buffer VARCHAR2(1000);
BEGIN
  doc := DBE_XMLDOM.NEWDOMDOCUMENT('<?xml version="1.0" encoding="UTF-8"?>
<computer size="ITX" price="19999">
    <cpu>Ryzen 9 3950X</cpu>
    <cpu>Ryzen 9 5950X_1</cpu>
    <ram>32GBx2 DDR4 3200MHz<cpu>Ryzen <cpu>Ryzen 9 5950X_2</cpu></cpu></ram>
    <motherboard>ROG X570i</motherboard>
    <gpu>RTX2070 Super</gpu>
    <ssd>1TB NVMe Toshiba + 2TB NVMe WD Black</ssd>
    <hdd>12TB WD Digital</hdd>
    <psu>CORSAIR SF750</psu>
    <case>LIANLI TU150</case>
</computer>');
  root_elem := DBE_XMLDOM.GETDOCUMENTELEMENT(doc);
  node_list := DBE_XMLDOM.GETELEMENTSBYTAGNAME(doc, 'cpu');
  child_node := DBE_XMLDOM.ITEM(node_list, 2);
  DBE_XMLDOM.WRITETOBUFFER(child_node, buffer);
  DBE_OUTPUT.PRINT_LINE(buffer);
END;
/
-- Expected result:
<cpu>Ryzen <cpu>Ryzen 9 5950X_2</cpu></cpu>
ANONYMOUS BLOCK EXECUTE

-- 2. Search the DOMElement node by TAGNAME and return the matched DOMNodelist node list.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  root_elem DBE_XMLDOM.DOMElement;
  child_node DBE_XMLDOM.DOMNODE;
  node_list DBE_XMLDOM.DOMNODELIST;
  buffer VARCHAR2(1000);
BEGIN
  doc := DBE_XMLDOM.NEWDOMDOCUMENT(
    '<?xml version="1.0" encoding="UTF-8"?>' ||
    '<computer size="ITX" price="19999">' ||
    '<cpu>Ryzen 9 3950X</cpu>' ||
    '<cpu>Ryzen 9 5950X_1</cpu>' ||
    '<ram>32GBx2 DDR4 3200MHz' ||
        '<cpu>Ryzen 9 5950X_2' ||
            '<cpu>Ryzen 9 5950X_3' ||
                '<cpu>Ryzen 9 5950X_4</cpu>' ||
            '</cpu>' ||
```

```
        '</cpu>' ||
    '</ram>' ||
    '</computer>');
  root_elem := DBE_XMLDOM.GETDOCUMENTELEMENT(doc);
  node_list := DBE_XMLDOM.GETELEMENTSBYTAGNAME(root_elem, 'cpu');
  child_node := DBE_XMLDOM.ITEM(node_list, 3);
  DBE_XMLDOM.WRITETOBUFFER(child_node, buffer);
  DBE_OUTPUT.PRINT_LINE(buffer);
END;
/
-- Expected result:
<cpu>Ryzen 9 5950X_3<cpu>Ryzen 9 5950X_4</cpu></cpu>
ANONYMOUS BLOCK EXECUTE

-- 3. Search the DOMElement node by TAGNAME and NAMESPACE, and return the matched
DOMNodelist node list.
DECLARE
  doc DBE_XMLDOM.DOMDocument;
  root_elem DBE_XMLDOM.DOMElement;
  child_node DBE_XMLDOM.DOMNODE;
  node_list DBE_XMLDOM.DOMNODELIST;
  buffer VARCHAR2(1000);
BEGIN
  doc := DBE_XMLDOM.NEWDOMDOCUMENT('<?xml version="1.0" encoding="UTF-8"?>
<computer size="ITX" price="19999" xmlns:h="www.xxxxx.com">
    <cpu>Ryzen 9 3950X</cpu>
    <cpu>Ryzen 9 5950X_1</cpu>
    <h:cpu>ns Ryzen 9 5950X_2</h:cpu>
    <ram>32GBx2 DDR4 3200MHz<cpu>Ryzen 9 5950X_3<cpu>Ryzen 9 5950X_4<cpu>Ryzen 9
5950X_5</cpu></cpu></cpu></ram>
    <motherboard>ROG X570i</motherboard>
    <gpu>RTX2070 Super</gpu>
    <ssd>1TB NVMe Toshiba + 2TB NVMe WD Black</ssd>
    <hdd>12TB WD Digital</hdd>
    <psu>CORSAIR SF750</psu>
    <case>LIANLI TU150</case>
</computer>');
  root_elem := DBE_XMLDOM.GETDOCUMENTELEMENT(doc);
  node_list := DBE_XMLDOM.GETELEMENTSBYTAGNAME(root_elem, 'cpu', 'www.xxxxx.com');
  child_node := DBE_XMLDOM.ITEM(node_list, 0);
  DBE_XMLDOM.WRITETOBUFFER(child_node, buffer);
  DBE_OUTPUT.PRINT_LINE(buffer);
END;
/
-- Expected result:
<h:cpu>ns Ryzen 9 5950X_2</h:cpu>
ANONYMOUS BLOCK EXECUTE
```

2.12.2.22 DBE_XMLGEN

APIs

The DBE_XMLGEN system package converts the result of an SQL query into a standard XML format and returns the result. For details about all the supported APIs, see **Table 2-467**.

Table 2-466 DBE_XMLGEN data type

Type	Description
DBE_XMLGEN.CTXHANDLE	Data type used to store the XML output status.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- A maximum of 65,535 context handles can exist in a session. Closing a context handle does not reclaim the resource.
- The CONNECT BY statement is not supported in the distributed system. Therefore, the NEWCONTEXTFROMHIERARCHY function cannot be constructed.
- In a distributed system, the cursor cannot be moved reversely. Therefore, the RESTARTQUERY function is unavailable.
- The case of the table column and type in the output XML file is the same as that of the table column and type created by the user. If you need to write a large column and type name, use double quotation marks ("") to specify the column and type name.
- During NEWCONTEXTFROMHIERARCHY initialization, the SETNULLHANDLING, USENULLATTRIBUTEINDICATOR, and SETCONVERTSPECIALCHARS methods can be used to set this parameter, but the setting does not take effect.
- If ORDER BY is not added for query, the query result is disordered, which affects the return result of the GETXML function.
- When the EXECUTE DIRECT ON syntax is used to execute functions in the DBE_XMLGEN package, if the GUC parameter **enable_udf_execute_direct** is set to **on**, the functions can be executed; if the parameter is set to **off**, the functions cannot be executed.

Table 2-467 DBE_XMLGEN

API	Description
DBE_XMLGEN.CONVERT	Encodes or decodes the input character string in XML format.
DBE_XMLGEN.NEWCONTEXT	Initializes the common context handle.
DBE_XMLGEN.NEWCONTEXTFROMHIERARCHY	Initializes the context handle with recursive elements.
DBE_XMLGEN.SETCONVERTSPECIALCHARS	Specifies whether the output XML needs to be encoded.
DBE_XMLGEN.SETNULLHANDLING	Sets how to display the null value in the XML file.
DBE_XMLGEN.SETROWSETTAG	Sets the name of the XML root node.
DBE_XMLGEN.SETROWTAG	Sets the tag name of each row of data in the XML file.
DBE_XMLGEN.USENULLATTRIBUTEINDICATOR	Adds the **xsi:nil="true"** attribute to the element where the **null** value is located in the XML file.
DBE_XMLGEN.USEITEMTAGSFORCOLL	Adds the suffix **_ITEM** to the element where the variable of the array type is located.
DBE_XMLGEN.GETNUMROWSPROCESSED	Views the number of data rows returned by GETXML or GETXMLTYPE last time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_XMLGEN.SETMAXROWS	Sets the maximum number of rows returned for a GETXML operation.
DBE_XMLGEN.SETSKIPROWS	Sets the number of SQL rows to be skipped.
DBE_XMLGEN.RESTARTQUERY	Restarts the SQL.
DBE_XMLGEN.GETXMLTYPE	Returns the XML text of the XMLTYPE type.
DBE_XMLGEN.GETXML	Returns the XML text of the CLOB type.
DBE_XMLGEN.CLOSECONTEXT	Disables the context handle.

- DBE_XMLGEN.CONVERT

  Encodes or decodes the input character string in XML format. The conversion is performed according to the following rules.

  Table 2-468 XML encoding rules

Original Value	Code Value
&	&
<	<
>	>
"	"
'	'

  The prototype of the DBE_XMLGEN.CONVERT function is as follows:

```
DBE_XMLGEN.CONVERT(XMLSTR IN VARCHAR2, FLAG IN NUMBER := 0) RETURNS VARCHAR2;
DBE_XMLGEN.CONVERT(XMLCLOB IN CLOB, FLAG IN NUMBER := 0) RETURNS CLOB;
```

  Table 2-469 DBE_XMLGEN.CONVERT API parameters

Parameter	Description
XMLSTR	XML character string to be converted. The value is of the VARCHAR2 type.
XMLCLOB	XML character string to be converted. The value is of the CLOB type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
FLAG	Transcodes or decodes character strings. **0**: encoding. **1**: decoding.

Example:

```
-- XML decoding
SELECT DBE_XMLGEN.CONVERT('<foo/>', 1);
 convert
---------
 <foo/>
(1 row)
-- XML encoding
SELECT DBE_XMLGEN.CONVERT('<foo><qwe</foo>', 0);
           convert
--------------------------------
 &lt;foo&gt;&lt;qwe&lt;/foo&gt;
(1 row)
```

- DBE_XMLGEN.NEWCONTEXT

  Initializes a common context handle. The context handle can be used only in one transaction. Otherwise, an error is reported when **DBE_XMLGEN.RESTARTQUERY** and **DBE_XMLGEN.CLOSECONTEXT** are used to close the context handle.

  The prototype of the DBE_XMLGEN.NEWCONTEXT function is as follows:

  ```
  DBE_XMLGEN.NEWCONTEXT(QUERYSTRING IN VARCHAR2) RETURNS DBE_XMLGEN.CTXHANDLE;
  DBE_XMLGEN.NEWCONTEXT(QUERYSTRING IN SYS_REFCURSOR) RETURNS
  DBE_XMLGEN.CTXHANDLE;
  ```

  Table 2-470 DBE_XMLGEN.NEWCONTEXT API parameters

Parameter	Description
QUERYSTRING	SQL statement or SYS_REFCURSOR used to generate XML files.

- DBE_XMLGEN.NEWCONTEXTFROMHIERARCHY

  Initializes the context handle with recursive elements. The data format must contain two columns. The first column is of the numeric type, and the second column is of the XML or XMLTYPE type. Generally, the value is generated by the CONNECT BY statement. The first column specifies the generation level. The distributed system does not support the connect by statement. Therefore, this function case cannot be constructed.

  NOTE

  The number of nested layers in the generated XML file cannot exceed 50 million.

  The prototype of the DBE_XMLGEN.NEWCONTEXTFROMHIERARCHY function is as follows:

  ```
  DBE_XMLGEN.NEWCONTEXTFROMHIERARCHY(QUERYSTRING IN VARCHAR2);
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-471 DBE_XMLGEN.NEWCONTEXTFROMHIERARCHY API parameters

Parameter	Description
QUERYSTRING	XML character string to be converted. The value is of the VARCHAR2 type.

- DBE_XMLGEN.SETCONVERTSPECIALCHARS

  Specifies whether the output XML needs to be encoded. If XML encoding is canceled, XML injection may occur. Due to performance issues, if the XML is secured, XML encoding can be skipped.

  The prototype of the DBE_XMLGEN.SETCONVERTSPECIALCHARS function is as follows:

  ```
  DBE_XMLGEN.SETCONVERTSPECIALCHARS(CTX IN DBE_XMLGEN.CTXHANDLE, CONV IN BOOLEAN);
  ```

  Table 2-472 DBE_XMLGEN.SETCONVERTSPECIALCHARS API parameters

Parameter	Description
CTX	Context handle.
CONV	Specifies whether to encode the output XML file. ● **true**: yes. ● **false**: no.

- DBE_XMLGEN.SETNULLHANDLING

  Sets how to display the null value in the XML file.

  The prototype of the DBE_XMLGEN.SETNULLHANDLING function is as follows:

  ```
  DBE_XMLGEN.SETNULLHANDLING(CTX IN DBE_XMLGEN.CTXHANDLE, FLAG IN NUMBER := 0);
  ```

  Table 2-473 DBE_XMLGEN.SETNULLHANDLING API parameters

Parameter	Description
CTX	Context handle.
FLAG	Null value display format. ● **0**: The element is not displayed. ● **1**: The **xsi:nil="true"** attribute is added to the element. ● **2**: Self-closed elements are displayed.

- DBE_XMLGEN.SETROWSETTAG

  Sets the name of the XML root node.

  The prototype of the DBE_XMLGEN.SETROWSETTAG function is as follows:

  ```
  DBE_XMLGEN.SETROWSETTAG(CTX IN DBE_XMLGEN.CTXHANDLE, ROWSETTAGNAME IN VARCHAR2);
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-474 DBE_XMLGEN.SETROWSETTAG API parameters

Parameter	Description
CTX	Context handle.
ROWSETTAGNAME	Name of the XML root node.

- DBE_XMLGEN.SETROWTAG

  Sets the tag name of each row of data in the XML file.

  The prototype of the DBE_XMLGEN.SETROWTAG function is as follows:

  ```
  DBE_XMLGEN.SETROWTAG(CTX IN DBE_XMLGEN.CTXHANDLE, ROWTAGNAME IN VARCHAR2);
  ```

  Table 2-475 DBE_XMLGEN.SETROWTAG API parameters

Parameter	Description
CTX	Context handle.
ROWTAGNAME	Tag name of each row of data.

- DBE_XMLGEN.USENULLATTRIBUTEINDICATOR

  Adds the **xsi:nil="true"** attribute to the element where the **null** value is located in the XML file.

  The prototype of the DBE_XMLGEN.USENULLATTRIBUTEINDICATOR function is as follows:

  ```
  DBE_XMLGEN.USENULLATTRIBUTEINDICATOR(CTX IN DBE_XMLGEN.CTXHANDLE, ATTRIND IN
  BOOLEAN);
  ```

  Table 2-476 DBE_XMLGEN.USENULLATTRIBUTEINDICATOR API parameters

Parameter	Description
CTX	Context handle.
ATTRIND	None.

- DBE_XMLGEN.USEITEMTAGSFORCOLL

  Adds the suffix **_ITEM** to the element where the variable of the array type is located.

  The prototype of the DBE_XMLGEN.USEITEMTAGSFORCOLL function is as follows:

  ```
  DBE_XMLGEN.USEITEMTAGSFORCOLL(CTX IN DBE_XMLGEN.CTXHANDLE);
  ```

  Table 2-477 DBE_XMLGEN.USEITEMTAGSFORCOLL API parameters

Parameter	Description
CTX	Context handle.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
-- Preset data.
CREATE TABLE test_for_array(idd number, id INT[]);
INSERT INTO test_for_array VALUES(1, ARRAY[1,2,3]);
SELECT DBE_XMLGEN.GETXML('SELECT * from test_for_array');
-- Add the suffix _ITEM to the array type.
DECLARE
qryctx DBE_XMLGEN.CTXHANDLE;
result CLOB;
BEGIN
    qryctx := DBE_XMLGEN.NEWCONTEXT('SELECT id from test_for_array');
    DBE_XMLGEN.useItemTagsForColl(qryctx);
    result:=DBE_XMLGEN.GETXML(qryctx);
    DBE_XMLGEN.CLOSECONTEXT(qryctx);
    DBE_OUTPUT.PUT_LINE(result);
END;
/
<?xml version="1.0"?>
<ROWSET>
<ROW>
  <id>
    <int4_ITEM>1</int4_ITEM>
    <int4_ITEM>2</int4_ITEM>
    <int4_ITEM>3</int4_ITEM>
  </id>
</ROW>
</ROWSET>
ANONYMOUS BLOCK EXECUTE
DROP TABLE test_for_array;
```

- DBE_XMLGEN.GETNUMROWSPROCESSED

  Views the number of data rows returned by GETXML or GETXMLTYPE last time.

  The prototype of the DBE_XMLGEN.GETNUMROWSPROCESSED function is as follows:

  ```
  DBE_XMLGEN.GETNUMROWSPROCESSED(CTX IN DBE_XMLGEN.CTXHANDLE);
  ```

  Table 2-478 DBE_XMLGEN.GETNUMROWSPROCESSED API parameters

Parameter	Description
CTX	Context handle.

- DBE_XMLGEN.SETMAXROWS

  Sets maximum number of rows returned by each GETXML operation.

  The prototype of the DBE_XMLGEN.SETMAXROWS function is as follows:

  ```
  DBE_XMLGEN.SETMAXROWS(CTX IN DBE_XMLGEN.CTXHANDLE, MAXROWS IN NUMBER);
  ```

  Table 2-479 DBE_XMLGEN.SETMAXROWS API parameters

Parameter	Description
CTX	Context handle.
MAXROWS	Maximum number of rows returned by each GETXML operation.

- DBE_XMLGEN.SETSKIPROWS

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Sets the number of SQL rows to be skipped.

The prototype of the DBE_XMLGEN.SETSKIPROWS function is as follows:

```
DBE_XMLGEN.SETSKIPROWS(CTX IN DBE_XMLGEN.CTXHANDLE, SKIPROWS IN NUMBER);
```

Table 2-480 DBE_XMLGEN.SETSKIPROWS API parameters

Parameter	Description
CTX	Context handle.
SKIPROWS	Number of SQL header rows that are skipped.

- DBE_XMLGEN.RESTARTQUERY

  Restarts the SQL. This API is not supported in distributed mode.

  The prototype of the DBE_XMLGEN.RESTARTQUERY function is as follows:

  ```
  DBE_XMLGEN.RESTARTQUERY(CTX IN DBE_XMLGEN.CTXHANDLE);
  ```

  Table 2-481 DBE_XMLGEN.RESTARTQUERY API parameters

Parameter	Description
CTX	Context handle.

- DBE_XMLGEN.GETXMLTYPE

  Returns the XML text of the XMLTYPE type.

  The prototype of the DBE_XMLGEN.GETXMLTYPE function is as follows:

  ```
  DBE_XMLGEN.GETXMLTYPE(SQLQUERY IN VARCHAR2, DTDORSCHEMA IN NUMBER := 0) RETURNS
  XMLTYPE;
  DBE_XMLGEN.GETXMLTYPE(CTX IN DBE_XMLGEN.CTXHANDLE, DTDORSCHEMA IN NUMBER := 0)
  RETURNS XMLTYPE;
  ```

  Table 2-482 DBE_XMLGEN.GETXMLTYPE API parameters

Parameter	Description
SQLQUERY	SQL statements that need to be converted into XML files.
DTDORSCHEMA	Meaningless.
CTX	Context handle.

- DBE_XMLGEN.GETXML

  Returns the XML text of the CLOB type.

  The prototype of the DBE_XMLGEN.GETXML function is as follows:

  ```
  DBE_XMLGEN.GETXML(SQLQUERY IN VARCHAR2, DTDORSCHEMA IN NUMBER := 0) RETURNS CLOB;
  DBE_XMLGEN.GETXML(CTX IN DBE_XMLGEN.CTXHANDLE, DTDORSCHEMA IN NUMBER := 0)
  RETURNS CLOB;
  DBE_XMLGEN.GETXML(CTX IN DBE_XMLGEN.CTXHANDLE, TMPCLOB INOUT CLOB, DTDORSCHEMA
  IN NUMBER := 0);
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-483 DBE_XMLGEN.GETXML API parameters

Parameter	Description
SQLQUERY	SQL statements that need to be converted into XML files.
DTDORSCHEMA	Meaningless.
CTX	Context handle.
TMPCLOB	CLOB variable for storing the output XML.

- DBE_XMLGEN.CLOSECONTEXT

  Disables the context handle.

  The prototype of the DBE_XMLGEN.CLOSECONTEXT function is as follows:

```
DBE_XMLGEN.CLOSECONTEXT(CTX IN DBE_XMLGEN.CTXHANDLE);
```

Table 2-484 DBE_XMLGEN.CLOSECONTEXT API parameters

Parameter	Description
CTX	Context handle.

Example:

```
CREATE TABLE IF NOT EXISTS department(department_id NUMBER, department_name
VARCHAR2(30), manager NUMBER, location NUMBER);
INSERT INTO department VALUES(10, 'administrator', 200, 1700);
INSERT INTO department VALUES(11, 'aaa', 200, 1700);
INSERT INTO department VALUES(12, 'bbb', 300, 1600);
INSERT INTO department VALUES(13, 'ccc', 400, 1600);
INSERT INTO department VALUES(14, 'aaa</row><a>asd</a><row>', 400, 1600);
INSERT INTO department VALUES(15, NULL, 500,1600);
INSERT INTO department VALUES(16, '!@#$ %^&*()+-=<>/\"a3_a', 400, 1600);
-- Disable the context.
DECLARE
    qryctx DBE_XMLGEN.CTXHANDLE;
    result CLOB;
BEGIN
    qryctx := DBE_XMLGEN.NEWCONTEXT('SELECT * from department where department_id=14');
    DBE_XMLGEN.SETCONVERTSPECIALCHARS(qryctx, false);
    result:=DBE_XMLGEN.GETXML(qryctx);
    DBE_XMLGEN.CLOSECONTEXT(qryctx);
    DBE_OUTPUT.PUT_LINE(result);
END;
/
<?xml version="1.0"?>
<ROWSET>
<ROW>
  <department_id>14</department_id>
  <department_name>aaa</row><a>asd</a><row></department_name>
  <manager>400</manager>
  <location>1600</location>
</ROW>
</ROWSET>
ANONYMOUS BLOCK EXECUTE
DROP TABLE department;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2.12.2.23 DBE_XMLPARSER

APIs

DBE_XMLPARSER is used to deserialize XML character strings and convert the character strings that store XML documents to document nodes. **Table 2-485** lists APIs supported by the DBE_XMLPARSER advanced package.

The XMLPARSER data type is used to store XML data. The maximum number of XML data records that can be stored is 16777215. The XMLPARSER data type can parse and create DOMDocument nodes based on the input string. The advanced package also provides the corresponding SET and GET APIs to parse the constraint attributes of the process.

NOTE

- When the DBE_XMLPARSER advanced package is used in the database whose character set is set to **SQL_ASCII**, an error is reported if characters beyond the ASCII range are input.
- The DBE_XMLPARSER advanced package supports only the ORA-compatible mode.

Table 2-485 DBE_XMLPARSER

API	Description
DBE_XMLPARSER.FREEPARSER	Frees a parser.
DBE_XMLPARSER.GETDOCUMENT	Obtains the parsed document node.
DBE_XMLPARSER.GETVALIDATIONMODE	Obtains the validation attribute.
DBE_XMLPARSER.NEWPARSER	Creates a parser instance.
DBE_XMLPARSER.PARSEBUFFER	Parses the VARCHAR string.
DBE_XMLPARSER.PARSECLOB	Parses the CLOB string.
DBE_XMLPARSER.SETVALIDATIONMODE	Sets the validation attribute.

- DBE_XMLPARSER.FREEPARSER

  Frees a given parser object.

  The prototype of DBE_XMLPARSER.FREEPARSER is as follows:

```
DBE_XMLPARSER.FREEPARSER (
 p    IN    parser);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-486 DBE_XMLPARSER.FREEPARSER API parameters

Parameter	Description
p	Parser object.

Example:

```
-- Create a parser and release it.
DECLARE
  l_parser dbe_xmlparser.parser;
  BEGIN
  l_parser := dbe_xmlparser.newparser();
-- Directly release the l_parser instance.
  dbe_xmlparser.freeparser(l_parser);
END;
/
```

- DBE_XMLPARSER.GETDOCUMENT

  Returns the root node of the DOM tree document constructed by the parser. This function can be called only after the document is parsed.

  The prototype of DBE_XMLPARSER.GETDOCUMENT is as follows:

```
DBE_XMLPARSER.GETDOCUMENT (
  p     IN     parser)
 RETURN DOMDocument;
```

Table 2-487 DBE_XMLPARSER.GETDOCUMENT API parameters

Parameter	Description
p	Parser object.

NOTE

- If the GETDOCUMENT function has no input parameter, an error is reported.
- If the **PARSER** parameter of the GETDOCUMENT function is null, null is returned.
- If **PARSER** input by the GETDOCUMENT function has not parsed any document, null is returned.

Example:

```
-- Create a parser to parse character strings and print the obtained document.
DECLARE
  l_parser dbe_xmlparser.parser;
  l_doc dbe_xmldom.domdocument;
  buffer varchar2 :=
'<?xml version="1.0" encoding="UTF-8"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Do not forget me this weekend!</body>
</note>';
  buffer2 varchar2;
  BEGIN
  l_parser := dbe_xmlparser.newparser();
 -- l_parser parses the character string and obtains the DOMDocument node through GETDOCUMENT.
  dbe_xmlparser.PARSEBUFFER(l_parser, buffer);
  l_doc := dbe_xmlparser.getdocument(l_parser);
 -- Print the content in l_doc.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  dbe_xmldom.writetobuffer(l_doc, buffer2);
  RAISE NOTICE '%', buffer2;

  dbe_xmlparser.freeparser(l_parser);
  dbe_xmldom.freedocument(l_doc);
END;
/
```

Execution result:

```
NOTICE:  <?xml version="1.0" encoding="UTF-8"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Do not forget me this weekend!</body>
</note>
```

- DBE_XMLPARSER.GETVALIDATIONMODE

  Obtains the parsing validation mode of a specified parser. If DTD validation is enabled, **true** is returned. Otherwise, **false** is returned.

  The prototype of DBE_XMLPARSER.GETVALIDATIONMODE is as follows:

  ```
  DBE_XMLPARSER.GETVALIDATIONMODE (
   p     IN     parser)
   RETURN BOOLEAN;
  ```

  Table 2-488 DBE_XMLPARSER.GETVALIDATIONMODE API parameters

Parameter	Description
p	Parser object.

  Example:

  ```
  -- Create a parser and use GETVALIDATIONMODE to check whether the parser validation mode is
  enabled.
  DECLARE
    l_parser dbe_xmlparser.parser;
  BEGIN
    l_parser := dbe_xmlparser.newparser();
      IF (dbe_xmlparser.GETVALIDATIONMODE(l_parser) = true) THEN
   RAISE NOTICE 'validation';
      ELSE
      RAISE NOTICE 'no validation';
    END IF;
    dbe_xmlparser.freeparser(l_parser);
  END;
  /
  ```

  Execution result:

  ```
  NOTICE:  validation
  ```

- DBE_XMLPARSER.NEWPARSER

  Creates a parser object and returns a new parser instance.

  The prototype of DBE_XMLPARSER.NEWPARSER is as follows:

  ```
  DBE_XMLPARSER.NEWPARSER
   RETURN Parser;
  ```

  Example:

  ```
  -- Create a parser to parse character strings and then free the parser.
  DECLARE
    -- Create a parser.
    l_parser dbe_xmlparser.parser;
    l_doc dbe_xmldom.domdocument;
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 buffer varchar2(1000) :=
  '<?xml version="1.0" encoding="UTF-8"?>
  <note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Do not forget me this weekend!</body>
  </note>';
 buffer2 varchar2(1000);
 BEGIN
 l_parser := dbe_xmlparser.newparser();
 -- Parse the document and create a new DOM document.
 dbe_xmlparser.PARSEBUFFER(l_parser, buffer);

 dbe_xmlparser.freeparser(l_parser);
END;
/
```

- DBE_XMLPARSER.PARSEBUFFER

  Parses XML documents stored in strings.

  The prototype of DBE_XMLPARSER.PARSEBUFFER is as follows:

```
DBE_XMLPARSER.PARSEBUFFER (
  p     IN     parser,
  doc   IN VARCHAR2);
```

Table 2-489 DBE_XMLPARSER.PARSEBUFFER API parameters

Parameter	Description
p	Parser object.
doc	A string that stores XML documents.

NOTE

- The maximum length of a character string that can be parsed by the PARSEBUFFER function is 32767. If the length exceeds the maximum, an error is reported.
- Different from the ORA database, this database supports only UTF-8 in terms of character encoding, and **version** can only be set to **1.0**. If versions 1.0 to 1.9 are parsed, the execution is normal but a warning appears. For versions later than 1.9, an error is reported.
- DTD validation differences:
  - **!ATTLIST to type (CHECK|check|Check) "Ch..."** reports an error because the default value **"Ch..."** is not an enumerated value in the brackets. However, the ORA database does not report this error.
  - **<!ENTITY baidu "www.baidu.com">..... &Baidu;&writer** reports an error because the letters are case-sensitive. **Baidu** cannot correspond to **baidu**. However, the ORA database does not report this error.
- Namespace validation difference: Undeclared namespace tags are parsed. However, the ORA database reports an error.
- Difference in parsing XML predefined entities: **'** and **"** are parsed and translated into ' and ". However, predefined entities in ORA database are not translated into characters.

Example:

```
-- Create a parser to parse character strings and print the obtained document.
DECLARE
 l_parser dbe_xmlparser.parser;
 l_doc dbe_xmldom.domdocument;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  buffer varchar2 :=
'<?xml version="1.0" encoding="UTF-8"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Do not forget me this weekend!</body>
</note>';
  buffer2 varchar2;
  BEGIN
  l_parser := dbe_xmlparser.newparser();
  dbe_xmlparser.PARSEBUFFER(l_parser, buffer);
  l_doc := dbe_xmlparser.getdocument(l_parser);

  dbe_xmldom.writetobuffer(l_doc, buffer2);
  RAISE NOTICE '%', buffer2;

  dbe_xmlparser.freeparser(l_parser);
  dbe_xmldom.freedocument(l_doc);
END;
/
```

Execution result:

```
NOTICE:  <?xml version="1.0" encoding="UTF-8"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Do not forget me this weekend!</body>
</note>
```

- DBE_XMLPARSER.PARSECLOB

  Parses XML documents stored in a CLOB.

  The prototype of DBE_XMLPARSER.PARSECLOB is as follows:

  ```
  DBE_XMLPARSER.PARSECLOB (
    p     IN     parser,
    doc   IN CLOB);
  ```

  Table 2-490 DBE_XMLPARSER.PARSECLOB API parameters

Parameter	Description
p	Parser object.
doc	A CLOB that stores XML documents.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- PARSECLOB cannot parse CLOBs larger than 1 GB.
- Different from the ORA database, this database supports only UTF-8 in terms of character encoding, and **version** can only be set to **1.0**. If versions 1.0 to 1.9 are parsed, the execution is normal but a warning appears. For versions later than 1.9, an error is reported.
- DTD validation differences:
  - **!ATTLIST to type (CHECK|check|Check) "Ch..."** reports an error because the default value **"Ch..."** is not an enumerated value in the brackets. However, the ORA database does not report this error.
  - **<!ENTITY baidu "www.baidu.com">..... &Baidu;&writer** reports an error because the letters are case-sensitive. **Baidu** cannot correspond to **baidu**. However, the ORA database does not report this error.
- Namespace validation difference: Undeclared namespace tags are parsed. However, the ORA database reports an error.
- Difference in parsing XML predefined entities: **'** and **"** are parsed and translated into ' and ". However, predefined entities in ORA database are not translated into characters.

Example:

```
-- Create a parser to parse character strings and print the obtained document.
DECLARE

 l_clob clob :=
'<?xml version="1.0" encoding="UTF-8"?>
 <note>
 <to>Tove</to>
 <from>Jani</from>
 <heading>Reminder</heading>
 <body>this weekend!</body>
 </note>';
 -- Create a parser.
 l_parser dbe_xmlparser.parser;
 l_doc dbe_xmldom.domdocument;
 buffer varchar2(1000);
 BEGIN
 l_parser := dbe_xmlparser.newparser();
 -- Parse the document and create a new DOM document.
 dbe_xmlparser.parseclob(l_parser, l_clob);
 l_doc := dbe_xmlparser.getdocument(l_parser);
 dbe_xmldom.writetobuffer(l_doc, buffer);
 RAISE NOTICE '%',buffer;

 dbe_xmlparser.freeparser(l_parser);
 dbe_xmldom.freedocument(l_doc);

 END;
 /
```

Execution result:

```
NOTICE:  <?xml version="1.0" encoding="UTF-8"?>
<note>
 <to>Tove</to>
 <from>Jani</from>
 <heading>Reminder</heading>
 <body>this weekend!</body>
</note>
```

- DBE_XMLPARSER.SETVALIDATIONMODE

  Sets the parsing validation mode of a specified parser.

  The prototype of DBE_XMLPARSER.SETVALIDATIONMODE is as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
DBE_XMLPARSER.SETVALIDATIONMODE(
 p     IN     parser)
 yes   IN BOOLEAN);
```

Table 2-491 DBE_XMLPARSER.SETVALIDATIONMODE API parameters

Parameter	Description
p	Parser object.
yes	Mode to be set: • **true**: DTD validation is enabled. • **false**: DTD validation is disabled.

NOTE

- If the input parameter **yes** of the SETVALIDATIONMODE function is null, the parsing validation mode of the parser is not changed.
- By default, the DTD validation is enabled during parser initialization.

Example 1:

```
-- Create a parser. The XML character string to be parsed does not match the DTD format.
-- If setValidationMode is set to false, the string can be parsed. If setValidationMode is set to true,
an error is reported during parsing.
DECLARE
 l_clob clob :=
'<!DOCTYPE note [
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<table>
<name attr1="WEB" attr2="web2">African Coffee Table</name>
<width>80</width>
<length>120</length>
</table>';
 l_parser dbe_xmlparser.parser;
 l_doc dbe_xmldom.domdocument;
buffer varchar2(1000);
 BEGIN
 l_parser := dbe_xmlparser.newparser();
 -- Set it to false for parsing.
   dbe_xmlparser.setValidationMode(l_parser, false);
   dbe_xmlparser.parseclob(l_parser, l_clob);
   l_doc := dbe_xmlparser.getdocument(l_parser);
   dbe_xmldom.writetobuffer(l_doc, buffer);
   RAISE NOTICE '%', buffer;
   dbe_xmlparser.freeparser(l_parser);
   dbe_xmldom.freedocument(l_doc);
 END;
 /
```

Execution result:

```
NOTICE:  <?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE note [
<!ELEMENT note (to , from , heading , body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
]>
<table>
 <name attr1="WEB" attr2="web2">African Coffee Table</name>
 <width>80</width>
 <length>120</length>
 </table>
```

Example 2:

```
-- Create a parser. The XML character string to be parsed does not match the DTD format.
-- An error is reported during parsing after setValidationMode is set to true.
DECLARE
  l_clob clob :=
 '<!DOCTYPE note [
 <!ELEMENT note (to,from,heading,body)>
 <!ELEMENT to (#PCDATA)>
 <!ELEMENT from (#PCDATA)>
 <!ELEMENT heading (#PCDATA)>
 <!ELEMENT body (#PCDATA)>
 ]>
 <table>
 <name attr1="WEB" attr2="web2">African Coffee Table</name>
 <width>80</width>
 <length>120</length>
 </table>';
  l_parser dbe_xmlparser.parser;
  l_doc dbe_xmldom.domdocument;
 buffer varchar2(1000);
  BEGIN
  l_parser := dbe_xmlparser.newparser();
  -- Set it to true for parsing.
  -- The XML character string does not match the DTD format and is expected to trigger an error.
    dbe_xmlparser.setValidationMode(l_parser, true);
    dbe_xmlparser.parseclob(l_parser, l_clob);
    l_doc := dbe_xmlparser.getdocument(l_parser);
    dbe_xmldom.writetobuffer(l_doc, buffer);
    dbe_xmlparser.freeparser(l_parser);
    dbe_xmldom.freedocument(l_doc);
  END;
  /
```

Execution result:

```
An error is reported during xmlparser parsing.
ERROR:  invalid XML document
```

2.12.2.24 PRVT_ILM

The PRVT_ILM API is an internal API of the ILM feature and cannot be directly called by users. This section lists only the API names and does not provide detailed API prototype description.

API	Description
be_active_ado_window	Triggers a maintenance window and updates the ilmadowindow triggering time and action.
be_create_ado_window_for_each_db	Triggers a maintenance window and creates ilmadowindow in each database of the instance.
be_execute_ilm	Executes ilmadowind for automatic evaluation.
be_execute_ilm_dn	Performs automatic evaluation on distributed DNs. This operation affects the overall ILM background scheduling of the cluster.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
flush_task_executestate	Updates the automatic scheduling task status.
generate_taskoid	Generates a task ID for this scheduling on the CN.
evaluate_obj_policy	Triggers evaluation on data objects after scheduling.
change_be_ilm	Calls the DBE_ILM_ADMIN.ENABLE_ILM() and DISABLE_ILM () help functions. This operation affects the overall ILM background scheduling of the cluster.
get_compression_ratio	Calls the DBE_COMPRESSION.GET_COMPRESSION_RATIO() help function.
get_compression_type	Calls the DBE_COMPRESSION.GET_COMPRESSION_TYPE() help function.
get_lastmodified_time	Calls the DBE_HEAT_MAP.ROW_HEAT_MAP() help function.
ilm_job_action	Executes compression tasks, which is called on distributed DNs.
delete_expired_ilm_log	Clears expired logs in distributed mode.
delete_gs_ilm_task	Operates a feature-related system catalog, which is called on distributed DNs.
insert_gs_ilm_jobdetail	Operates a feature-related system catalog, which is called on distributed DNs.
insert_gs_ilm_task	Operates a feature-related system catalog, which is called on distributed DNs.
insert_gs_ilm_taskdetail	Operates a feature-related system catalog, which is called on distributed DNs.
update_gs_ilm_jobdetail	Operates a feature-related system catalog, which is called on distributed DNs.
update_gs_ilm_object	Operates a feature-related system catalog, which is called on distributed DNs.
update_gs_ilm_param	Operates a feature-related system catalog, which is called on distributed DNs. This operation affects the overall ILM background scheduling of the cluster.
update_gs_ilm_task	Operates a feature-related system catalog, which is called on distributed DNs.
gs_ilm_ticker	Triggers a maintenance window and ticks once to record the mapping between an LSN and time.
compress_blocks	Calls the PRVT_ILM.ILM_JOB_ACTION() help function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
compress_block_single	This API has been deprecated. Do not use it.
get_job_status	Calls the DBE_ILM.STOP_ILM help function.
insert_gs_ilm_param	Operates a feature-related system catalog.
query_unfinishedjob_num	Checks whether there are jobs that are not complete, including jobs in the initial or running state.
update_gs_ilm_object_flag2	Operates a feature-related system catalog, which is called on distributed DNs.
ilm_seq_nextval	Obtains the next sequence value of the ILM.
ilm_seq_setval	Sets the current sequence value of the ILM.
check_compatibility	Calls a help function to determine the database compatibility mode in a unified manner.
set_pg_settings	Calls a help function to set the database compatibility mode using a GUC parameter.
check_seq	Checks whether the current value of **GSILMTASK_SEQ** is greater than the maximum value of **taskoid** in the GS_ILM_TASK catalog, which is called on distributed CNs.
get_actual_compression_ratio	Calls the DBE_COMPRESSION.GET_ACTUAL_COMPRESSION_RATIO() help function.
get__hole_ratio	Calls the DBE_COMPRESSION.GET_HOLE_RATIO() help function.
turbo_compress_buffer_stats	Calls the DBE_COMPRESSION.TURBO_COMPRESS_BUFFER_STATS() help function.
get_index_compression_type	Calls the DBE_COMPRESSION.GET_INDEX_COMPRESSION_TYPE() help function.
create_ilm_db_policy	Calls the DBE_ILM_ADMIN.CREATE_ILM_DB_POLICY() help function.
delete_ilm_db_policy	Calls the DBE_ILM_ADMIN.DELETE_ILM_DB_POLICY() help function.
compress_indexes	Calls the PRVT_ILM.ILM_JOB_ACTION() help function.
eval_comp_relgrowth	This API has been deprecated. Do not use it.
delete_gs_ilm_jobdetail	Operates a feature-related system catalog, which is called on distributed DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

2.12.3 Internal Functions

The following APIs are internal APIs of advanced packages. You are advised not to call them directly. Otherwise, unexpected behavior or running effect may occur.

DBE_DESCRIBE

Table 2-492 describes the internal APIs contained in DBE_DESCRIBE.

Table 2-492 Internal APIs of the DBE_DESCRIBE package

API	Description
DBE_DESCRIBE.GET_PROCEDURE_NAME	Extracts the name of a stored procedure or function from the original input of a user.
DBE_DESCRIBE.IS_NUMBER_TYPE	Checks whether the type is numeric.

DBE_UTILITY

Table 2-493 describes the internal APIs contained in DBE_UTILITY.

Table 2-493 Internal APIs of the DBE_UTILITY package

API	Description
DBE_UTILITY.EXPAND_SQL_TEXT	Expands SQL statements that contain views or synonyms into equivalent queries that reference only the base table and join.
DBE_UTILITY.CANONICALIZE_RET	Standardizes the formatting of database object names (such as tables, views, columns, and users).
DBE_UTILITY.COMMA_TO_TABLE_FUNC	Converts a comma-separated list of strings into a PL/SQL index table and returns the number of elements.
DBE_UTILITY.COMPILE_SCHEMA	Compiles invalid or all PL/SQL objects such as stored procedures, functions, and views in a specified schema in batches. pkg_util.gs_compile_schema is recommended.
DBE_UTILITY.NAME_SEPARATE	Parses an object name based on the database object naming rules and obtains the schema name, object name, and database link.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_UTILITY.NAME_TOKENIZE_FUNC	Parses a database object name into five parts (schema name, table name, column name, and total length).
DBE_UTILITY.NAME_TOKENIZE_LOWER	Parses a database object name into five parts (schema name, table name, column name, and total length), and converts all parts into lowercase letters.
DBE_UTILITY.NAME_TOKENIZE_LOWER_FUNC	Equivalent to DBE_UTILITY.NAME_TOKENIZE_LOWER.
DBE_UTILITY.PRIVILEGE_CHECK	Checks the permissions of the current user on a specified database object.
DBE_UTILITY.SEARCH_CLASS_WITH_NSPOID_ONAME_TYPE	Searches for database objects by name and type in the specified namespace, and returns the OIDs and quantity of the objects.
DBE_UTILITY.SEARCH_OBJECTS	Searches for database objects that match the schema name, object name, and object type in the database, and returns the OID, object type, and whether the object is a synonym.
DBE_UTILITY.SEARCH_OBJECTS_SYNONYM_FILL_SCHEMA	Searches for database objects with the specified name and type in multiple possible schemas based on the specified schema search type, and returns information about the found objects.
DBE_UTILITY.SEARCH_PROCEDURE_WITH_NSPOID_ONAME	Retrieves functions, stored procedures, and packages corresponding to the specified schema OID and name.
DBE_UTILITY.SEARCH_SYNONYM_WITH_NSPOID_ONAME	Retrieves the synonym corresponding to the specified schema OID and name.
DBE_UTILITY.TABLE_TO_COMMA_FUNC	Converts an array into a string.
DBE_UTILITY.USER_NAME	Returns the username of the current database.
DBE_UTILITY.SEARCH_OBJECTS_SYNONYM_FILL_SECHEMA	Has the same function as DBE_UTILITY.SEARCH_OBJECTS_SYNONYM_FILL_SCHEMA. You are advised to use DBE_UTILITY.SEARCH_OBJECTS_SYNONYM_FILL_SCHEMA.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_UTILITY.SEARCH_SYNONM_WITH_NSPOID_ONAME	Has the same function as DBE_UTILITY.SEARCH_SYNONYM_WITH_NSPOID_ONAME. You are advised to use DBE_UTILITY.SEARCH_SYNONYM_WITH_NSPOID_ONAME.

2.13 Retry

Retry is a process in which the database executes an SQL statement or stored procedure (including anonymous block) again in the case of execution failure, improving the execution success rate and user experience. In addition, the database checks the error code and retry configuration to determine whether to retry.

If the execution fails, the system rolls back the executed statements and executes the stored procedure again.

Example:

```
gaussdb=# CREATE OR REPLACE PROCEDURE retry_basic ( IN  x INT)
AS
BEGIN
  INSERT INTO t1 (a) VALUES (x);
  INSERT INTO t1 (a) VALUES (x+1);
END;
/

gaussdb=# CALL retry_basic(1);
```

2.14 Package

A package is a combination of PL/SQL programs, such as stored procedures, functions, variables, constants, and cursors. It is object-oriented and can encapsulate PL/SQL program design elements. Functions in a package are created, deleted, and modified in a unified manner.

A package contains two parts: package specifications and package body. The declaration contained in the package specifications can be accessed by external functions and anonymous blocks. The declaration contained in the package body cannot be accessed by external functions or anonymous blocks, but can be accessed only by functions and stored procedures in the package body.

For details about how to create a package, see **CREATE PACKAGE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- Cross-package variables cannot be used as control variables in the FOR loop.
- Types defined in a package cannot be deleted or modified, and cannot be used to define tables.
- Cursor variables cannot be referenced in the SCHEMA.PACKAGE.CURSOR format.
- A cursor with parameters can be opened only in the current package.
- In distributed mode, PACKAGE can be used only in an ORA-compatible database.
- In distributed mode, if an SQL statement contains variables or functions in a package, pushdown is not supported.
- A package variable cannot be used as the default value of a function or stored procedure parameter.
- If the input parameters of a stored procedure in a package contain pseudo parameters (for example, **anyelement**), variable-length parameters cannot be specified during a call.
- If the **in**, **out**, and **inout** parameters of a function in the package header and package body are different, the system identifies the functions to be different.

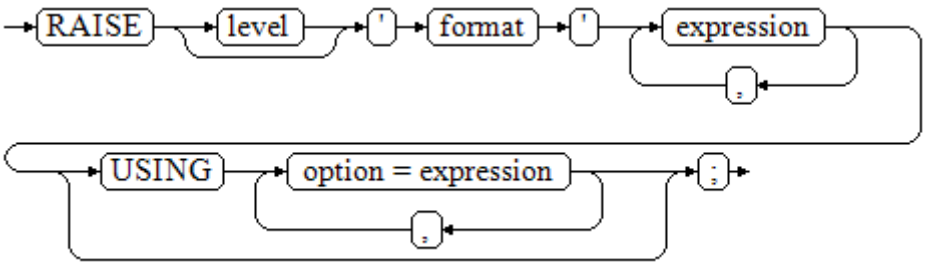
2.15 Debugging

Syntax

RAISE has the following five syntax formats.

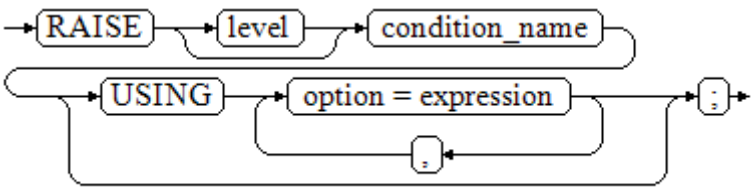
- The syntax of raise_format is shown in **Figure 2-51**.

  Figure 2-51 raise_format::=

- The syntax of raise_condition is shown in **Figure 2-52**.

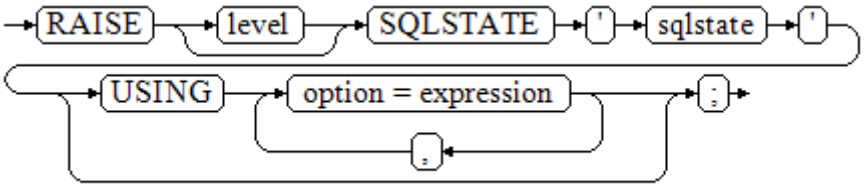
  Figure 2-52 raise_condition::=

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

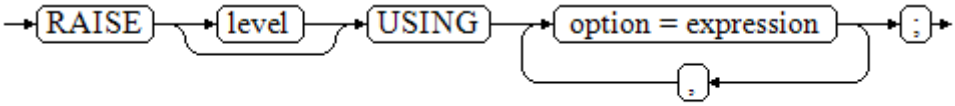
- The syntax of raise_sqlstate is shown in **Figure 2-53**.

  Figure 2-53 raise_sqlstate::=

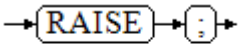
- The syntax of raise_option is shown in **Figure 2-54**.

  Figure 2-54 raise_option::=

- The syntax of raise is shown in **Figure 2-55**.

  Figure 2-55 raise::=

Parameters:

- **level**: specifies the error level, that is, **DEBUG**, **LOG**, **INFO**, **NOTICE**, **WARNING**, or **EXCEPTION** (default). **EXCEPTION** throws an error that normally terminates the current transaction and the others only generate information at their levels. The GUC parameters **log_min_messages** and **client_min_messages** control whether the error messages of specific levels are reported to the client and are written to the server log.
- **format**: specifies the error message text to be reported, which is a format string. The format string can be appended with an expression for insertion to the message text. In a format string, **%** is replaced by the parameter value attached to format and **%%** is used to print **%**. For example:

  ```
  -- v_job_id replaces % in the string.
  RAISE NOTICE 'Calling cs_create_job(%)',v_job_id;
  ```

- **option = expression**: inserts additional information to an error report. The **keyword** option can be **MESSAGE**, **DETAIL**, **HINT**, or **ERRCODE**, and each expression can be any string.
  - **MESSAGE**: specifies the error message text. This option cannot be used in a RAISE statement that contains a format character string in front of USING.
  - **DETAIL**: specifies detailed information of an error.
  - **HINT**: prints hint information.
  - **ERRCODE**: designates an error code (SQLSTATE) in a report. A condition name or a five-character SQLSTATE error code can be used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **condition_name**: specifies the condition name corresponding to the error code.
- **sqlstate**: specifies the error code.

If neither a condition name nor an SQLSTATE is specified in the **RAISE EXCEPTION** command, **RAISE EXCEPTION (P0001)** is used by default. If no message text is specified, the condition name or SQLSTATE is used as the message text by default.

NOTICE

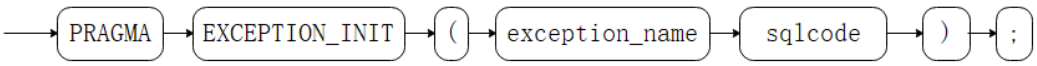
- If **SQLSTATE** specifies an error code, the error code is not limited to a defined error code. It can be any error code containing five digits or ASCII uppercase rather than **00000**. Do not use error codes ended with three zeros because such error codes are category codes and can be captured by the whole category.
- In ORA-compatible mode, **SQLCODE** is equivalent to **SQLSTATE**.
- The syntax described in **Figure 2-55** does not append any parameter. This form is used only for the EXCEPTION statement in a BEGIN block so that the error can be re-processed.

EXCEPTION_INIT

In ORA-compatible mode, EXCEPTION_INIT can be used to define the SQLCODE error code.

Figure 2-56 shows the syntax diagram.

Figure 2-56 exception_init::=



Parameters:

- **exception_name** indicates the name of the exception declared by the user. The EXCEPTION_INIT syntax must follow the declared exception.
- **sqlcode** is a user-defined SQL code, which must be a negative integer ranging from –2147483647 to –1.

Precautions:

- This function can be used in ORA-compatible mode but cannot be bound to system error codes.
- Only exception_init can be used to assign a value to **sqlcode** associated with exception. Other value assignment modes are invalid.
- Cross-database.schema.package.exception calling is not supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- When **EXCEPTION_INIT** is used to define an SQL code, **SQLSTATE** is equivalent to **SQLCODE**, and **SQLERRM** is in the format of *xxx*: **non-GaussDB Exception**. For example, if the user-defined SQL code is **–1**, **SQLSTATE** is **–1** and **SQLERRM** is **1: non-GaussDB Exception**.
- In the same stored procedure or function, if EXCEPTION_INIT is used to set the same error code, the Oracle Database does not support capturing exceptions separately as conditions. GaussDB supports this behavior. For details, see **Examples**.

Examples

Display error and hint information when a transaction terminates.

```
CREATE OR REPLACE PROCEDURE proc_raise1(user_id in integer)
AS
BEGIN
RAISE EXCEPTION 'Noexistent ID --> %',user_id USING HINT = 'Please check your user ID';
END;
/

call proc_raise1(300011);

-- Execution result
ERROR:  Noexistent ID --> 300011
HINT:  Please check your user ID
```

Two methods are available for setting **SQLSTATE**.

```
CREATE OR REPLACE PROCEDURE proc_raise2(user_id in integer)
AS
BEGIN
RAISE 'Duplicate user ID: %',user_id USING ERRCODE = 'unique_violation';
END;
/

\set VERBOSITY verbose
call proc_raise2(300011);

-- Execution result
ERROR:  Duplicate user ID: 300011
SQLSTATE: 23505
```

If the main parameter is a condition name or **SQLSTATE**, the following applies:

```
RAISE division_by_zero;
RAISE SQLSTATE '22012';
```

For example:

```
CREATE OR REPLACE PROCEDURE division(div in integer, dividend in integer)
AS
DECLARE
res int;
   BEGIN
   IF dividend=0 THEN
      RAISE division_by_zero;
      RETURN;
   ELSE
      res := div/dividend;
      RAISE INFO 'division result: %', res;
      RETURN;
   END IF;
   END;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
/
call division(3,0);

-- Execution result
ERROR:  division_by_zero
```

Alternatively:

```
RAISE unique_violation USING MESSAGE = 'Duplicate user ID: ' || user_id;
```

In the same stored procedure or function, if EXCEPTION_INIT is used to set the same error code, Oracle Database does not support exception capture as a condition, but GaussDB supports this behavior.

```
gaussdb=# CREATE OR REPLACE PACKAGE pkg IS
gaussdb$#   err1 EXCEPTION;
gaussdb$#   err2 EXCEPTION;
gaussdb$#   PRAGMA EXCEPTION_INIT(err1, -10001);
gaussdb$#   PRAGMA EXCEPTION_INIT(err2, -10001);
gaussdb$# PROCEDURE pro;
gaussdb$# END pkg;
gaussdb$# /
CREATE PACKAGE
gaussdb=# CREATE OR REPLACE PACKAGE BODY pkg IS
gaussdb$# PROCEDURE pro IS
gaussdb$# BEGIN
gaussdb$#     FOR i IN 1..10 LOOP
gaussdb$#         BEGIN
gaussdb$#         IF i = 1 THEN
gaussdb$#             raise err1;
gaussdb$#         END IF;
gaussdb$#         IF i = 2 THEN
gaussdb$#             raise err2;
gaussdb$#         END IF;
gaussdb$#         EXCEPTION
gaussdb$# -- The same error code is split as two different conditions of EXCEPTION.
gaussdb$#             WHEN err1 THEN
gaussdb$#                 CONTINUE;
gaussdb$#             WHEN err2 THEN
gaussdb$#                 EXIT;
gaussdb$#         END;
gaussdb$#     END LOOP;
gaussdb$# END pro;
gaussdb$# END pkg;
gaussdb$# /
CREATE PACKAGE BODY
gaussdb=# CALL pkg.pro();
 pro
-----

(1 row)
gaussdb=# DROP PACKAGE pkg;
NOTICE:  drop cascades to function public.pro()
DROP PACKAGE
```

2.16 Keyword

NOTE

Keywords are identifiers with special meanings in the PL/SQL and are case-insensitive. Keywords described in **Table 2-494** are PL/SQL keywords. Note that PL/SQL reserved keywords and non-reserved keywords are different from those in SQL. For details about SQL keywords, see **Keywords**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 2-494 PL/SQL Keywords

Keyword	Keyword Type	Whether It Is an SQL Keyword
ABSOLUTE	Non-reserved	Non-reserved
ALIAS	Non-reserved	-
ALL	Reserved	Reserved
ALTER	Non-reserved	Non-reserved
ARRAY	Non-reserved	Reserved
AS	Non-reserved	Reserved
BACKWARD	Non-reserved	Non-reserved
BULK	Non-reserved	-
BEGIN	Reserved	Non-reserved
BY	Reserved	Non-reserved
CASE	Reserved	Reserved
CLOSE	Reserved	Non-reserved
COLLATE	Reserved	Reserved
COLLECT	Non-reserved	-
COMMIT	Non-reserved	Non-reserved
CONSTANT	Non-reserved	Non-reserved
CONTINUE	Non-reserved	Non-reserved
CURSOR	Non-reserved	Non-reserved
CURRENT	Non-reserved	Non-reserved
DEBUG	Non-reserved	-
DECLARE	Reserved	Non-reserved
DEFAULT	Reserved	Reserved
DELETE	Reserved	Non-reserved
DETAIL	Non-reserved	-
DIAGNOSTICS	Reserved	-
DISTINCT	Non-reserved	Reserved
DUMP	Non-reserved	-
ELSE	Reserved	Reserved
ELSEIF	Reserved	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	Keyword Type	Whether It Is an SQL Keyword
ELSIF	Reserved	-
END	Reserved	Reserved
ERRCODE	Non-reserved	-
ERROR	Non-reserved	Non-reserved
EXCEPT	Non-reserved	Reserved
EXCEPTION	Reserved	-
EXCEPTIONS	Non-reserved	-
EXECUTE	Reserved	Non-reserved
EXIT	Reserved	-
FETCH	Reserved	Reserved
FIRST	Non-reserved	Non-reserved
FOR	Reserved	Reserved
FORALL	Reserved	-
FOREACH	Reserved	-
FORWARD	Non-reserved	Non-reserved
FROM	Reserved	Reserved
FUNCTION	Reserved	Non-reserved
GET	Reserved	-
GOTO	Reserved	-
HINT	Non-reserved	-
IF	Reserved	Non-reserved
IGNORE	Non-reserved	Non-reserved
IMMEDIATE	Non-reserved	Non-reserved
IN	Reserved	Reserved
INDEX	Non-reserved	Non-reserved
INFO	Non-reserved	-
INSERT	Reserved	Non-reserved
INSTANTIATION	Non-reserved	-
INTERSECT	Non-reserved	Reserved
INTO	Reserved	Reserved

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	Keyword Type	Whether It Is an SQL Keyword
IS	Non-reserved	Reserved
LAST	Non-reserved	Non-reserved
LIMIT	Reserved	Reserved
LOAD	Non-reserved	Non-reserved
LOG	Non-reserved	Non-reserved
LOOP	Reserved	Non-reserved
MERGE	Non-reserved	Non-reserved
MESSAGE	Non-reserved	-
MESSAGE_TEXT	Non-reserved	-
MOVE	Reserved	Non-reserved
MULTISET	Non-reserved	-
NEXT	Non-reserved	Non-reserved
NO	Non-reserved	Non-reserved
NOT	Reserved	Reserved
NOTICE	Non-reserved	-
NULL	Reserved	Reserved
OF	Reserved	Non-reserved
OPEN	Reserved	Non-reserved
OPTION	Non-reserved	Non-reserved
OR	Reserved	Reserved
OUT	Reserved	Non-reserved (cannot be functions or types)
PACKAGE	Non-reserved	Non-reserved
PERFORM	Non-reserved	-
PG_EXCEPTION_CONTEXT	Non-reserved	-
PG_EXCEPTION_DETAIL	Non-reserved	-
PG_EXCEPTION_HINT	Non-reserved	-
PIPE	Non-reserved	-
PIPELINED	Non-reserved	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	Keyword Type	Whether It Is an SQL Keyword
PRAGMA	Non-reserved	-
PRIOR	Non-reserved	Non-reserved
PROCEDURE	Reserved	Reserved
QUERY	Non-reserved	Non-reserved
RAISE	Reserved	-
RANGE	Non-reserved	Non-reserved
RECORD	Non-reserved	-
REF	Reserved	Non-reserved
RELATIVE	Non-reserved	Non-reserved
RELEASE	Non-reserved	Non-reserved
REPLACE	Non-reserved	Non-reserved
RESULT_OID	Non-reserved	-
RETURN	Reserved	Non-reserved
RETURNED_SQLSTATE	Non-reserved	-
REVERSE	Non-reserved	-
ROLLBACK	Non-reserved	Non-reserved
ROW_COUNT	Non-reserved	-
ROWTYPE	Non-reserved	Non-reserved
SAVE	Non-reserved	-
SAVEPOINT	Non-reserved	Non-reserved
SCROLL	Non-reserved	Non-reserved
SELECT	Reserved	Reserved
SLICE	Non-reserved	Non-reserved
SQLSTATE	Non-reserved	-
STACKED	Non-reserved	-
STRICT	Reserved	Non-reserved
SUBTYPE	Non-reserved	-
SYS_REFCURSOR	Non-reserved	Non-reserved
TABLE	Non-reserved	Reserved

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	Keyword Type	Whether It Is an SQL Keyword
THEN	Reserved	Reserved
TO	Reserved	Reserved
UNION	Non-reserved	Reserved
UPDATE	Reserved	Non-reserved
USING	Reserved	Reserved
USE_COLUMN	Non-reserved	-
USE_VARIABLE	Non-reserved	-
VARIABLE_CONFLICT	Non-reserved	-
VARRAY	Non-reserved	-
WARNING	Non-reserved	-
WHEN	Reserved	Reserved
WHILE	Reserved	-
WITH	Non-reserved	Reserved

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3 Compatibility Description

3.1 Overview of GaussDB Compatibility

In GaussDB, you can create a database with a compatibility mode of **ORA** (Oracle-compatible), **MYSQL** (MySQL-compatible), **TD** (Teradata-compatible), **PG** (Postgres-compatible), or **M** (M-compatible). The compatibility mode directly affects SQL syntax, data types, system functions, stored procedures, and more. Some compatibility APIs may be supported only in certain compatibility modes.

For details about Oracle-compatible mode, M-compatible mode, and MySQL-compatible mode, see **Oracle Compatibility Description**, **M-compatible Mode**, and **MYSQL-compatible Mode**. You can select a proper compatibility mode based on actual requirements.

3.2 Oracle Compatibility Description

3.2.1 Overview of Oracle Database Compatibility

GaussDB is generally compatible with Oracle Database in terms of basic functions (such as data types, SQL statements, and database objects) and PL/SQL. However, due to architecture design differences, there are still some incompatible items.

This chapter compares the Oracle-compatible mode in GaussDB with Oracle Database 19c.

3.2.2 Basic SQL Elements

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.2.2.1 Data Types

Table 3-1 Numeric types

Oracle Database	GaussDB	Difference
NUMBER [(p [, s])]	Supported, with differences.	The precision and usage are different. • When NUMBER contains parameters, the maximum boundary values of precision *p* and scale *s* in GaussDB are greater than those in Oracle Database. • In GaussDB, the default value of *p* when NUMBER does not contain parameters is much greater than the maximum boundary value when NUMBER contains parameters. However, in Oracle Database, the former is equal to the latter. • In GaussDB, the value of *s* cannot be negative. In Oracle Database, a negative *s* value is accurate to an integer. • In GaussDB, NaN values can be inserted. However, in Oracle Database, an error will be reported if NaN values are inserted. • If the scale *s* is not specified in GaussDB, the scale is not limited. In Oracle Database, if the scale *s* is not specified, the default scale is 0. • GaussDB does not support the number(*, s) format. Oracle Database supports this format, where the asterisk (*) indicates that the maximum precision is used (that is, the precision is not limited).
FLOAT [(p)]	Supported.	-
BINARY_FLOAT	Not supported.	-
BINARY_DOUBLE	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-2 Date and time types

Oracle Database	GaussDB	Difference
DATE	Supported, with differences.	If the **mapping_date_to_datea** parameter is not enabled, the precision of GaussDB is different from that in Oracle Database, and GaussDB supports a larger A.D. time range. If **mapping_date_to_datea** is enabled, the syntax, operators, and format control supported by the DATE type in GaussDB are different from those in Oracle Database.
TIMESTAMP [(fractional_seconds_precision)]	Supported, with differences.	-
TIMESTAMP [(fractional_seconds_precision)] WITH TIME ZONE	Supported, with differences.	The timestamptz type of GaussDB is equivalent to the timestampwithlocaltimezone type of Oracle Database. The type corresponding to timestamptz of Oracle Database is missing. Time zone update: In some countries or regions, the time zone information is often updated. Therefore, the database system often needs to modify the time zone file accordingly to ensure that the time is correct. Currently, the GaussDB time zone type involves only timestamp with timezone. When a new time zone file takes effect, the existing data is not changed, and the new data is adjusted based on the time zone file information. Data capabilities of GaussDB are different from those of Oracle Database.
TIMESTAMP [(fractional_seconds_precision)] WITH LOCAL TIME ZONE	Not supported.	-
INTERVAL YEAR [(year_precision)] TO MONTH	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
INTERVAL DAY [(day_precision)] TO SECOND [(fractional_seconds_precision)]	Supported.	-

NOTE

- In ORA-compatible mode, the calendar systems are different between GaussDB and Oracle Database as follows:
  - Oracle Database has two calendar systems: the Julian calendar and the Gregorian calendar. Before October 5, 1582, the Julian calendar is used. The leap year calculation rule is that the year must be divisible by 4. After October 5, 1582, the Gregorian calendar is used. The leap year calculation rule is that the year must be divisible by 4 and cannot be divisible by 100, or can be divisible by 400. GaussDB uses only the Gregorian calendar system. The leap year calculation rule is that the year must be divisible by 4 and not divisible by 100, or divisible by 400.
  - In Oracle Database, the year 1582 comprises 355 days, with the missing 10 days spanning from October 5, 1582, to October 14, 1582. The day following October 4, 1582, is October 15, 1582. In GaussDB, however, the year 1582 contains 365 days.
  - The Oracle Database has the year A.D. 0, so the period from '-0001-01-01 00:00:00' to '0001-01-01 00:00:00' contains 731 days. GaussDB does not have the year A.D. 0, so the period from '-0001-01-01 00:00:00' to '0001-01-01 00:00:00' contains 366 days.
- In ORA-compatible mode, when the **mapping_date_to_datea** parameter is not enabled, the DATE type is replaced by TIMESTAMP(0) WITHOUT TIME ZONE. The difference of this type between the two databases is the same as that of TIMESTAMP (0) WITHOUT TIME ZONE. When a timestamp value with a decimal point, for example, **timestamp '2023-01-01 23:59:59.5'**, is inserted into a DATE column, GaussDB rounds the value, and the result inserted into the table is **'2023-01-02 00:00:00'**. Oracle Database truncates the value, and the result inserted into the table is **'2023-01-01 23:59:59'**.
- In ORA-compatible mode, when the **mapping_date_to_datea** parameter is enabled, the DATE type is replaced by DATEA. The differences are the same as those of DATEA.
- In terms of TIMESTAMP [(fractional_seconds_precision)] and TIMESTAMP [(fractional_seconds_precision)] WITH TIME ZONE, the differences between GaussDB and Oracle Database are as follows:
  - The value of **fractional_seconds_precision** ranges from 0 to 6 in GaussDB, but ranges from 0 to 9 in Oracle Database.
  - GaussDB uses **DateStyle** to set the display format of date and time values and the rules of resolving ambiguous values. For details, see the description of date input in **Date/Time Types**. Generally, the input format verification and output display in Oracle Database are controlled by the **NLS_TIMESTAMP_FORMAT** and **NLS_TIMESTAMP_TZ_FORMAT** parameters.
  - By default, GaussDB removes zeros from the end of the decimal part of the second. Oracle Database controls the display of the decimal part based on the setting (**FF/FF1-FF9**) of the formatting parameter. For example, '2017-09-01 10:32:19.212000' is displayed as '2017-09-01 10:32:19.212' in GaussDB. In Oracle Database, it is displayed as '2017-09-01 10:32:19.212' if the **format** parameter contains FF, or '2017-09-01 10:32:19.212000000' if the **format** parameter contains **FF9**.
  - GaussDB supports a wider time range than Oracle Database.
  - For literals with more than six digits of precision, the seventh digit is rounded according to the banker's rounding rule—if the seventh digit is 5, the sixth digit is the nearest even number. For those within six digits of precision, the rounding rule baseline is 2000-01-01 00:00:00. If the time is earlier than the baseline, the banker's rounding rule is used. If the time is later than the baseline, the standard rounding rule is used. The time baseline of TIMESTAMP WITH TIME ZONE is calculated according to the UTC-0 time zone.
- In ORA-compatible mode, when the **mapping_date_to_datea** parameter is enabled, there are the following differences between its DATE type and Oracle Database's DATE type:
  - Literals can be defined as the DATEA type, for example, by running **SELECT datea '2024-10-01' ;**. In GaussDB, literal strings are verified and should comply with the format specified by **nls_date_format**; otherwise, an error is reported or incorrect

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

data is output. However, in Oracle Database, literal strings are verified and should be in the fixed format YYYY-MM-DD.

- Using CREATE TABLE, CREATE VIEW, or PREPARE to create objects are different between GaussDB and Oracle Database.

  In GaussDB, if an UNKNOWN type exists, it is converted to a definite type. If the conversion fails, an error is reported for object creation. If the conversion is successful, the value is saved as a value of the definite type, and subsequent operations are not affected by additional parameters. For example, the following is a typical scenario:

  ```
  WHERE datea_column > 'unknown_text'
  'unknown_text'::datea
  cast('unknown_text' as datea)
  datea'unknown text'
  ```

  Loose verification is used when the original definition contains other expressions such as functions. When defining an object, you can create it successfully as long as the return value type of the expression can be converted to the target type, and executing the expression is not needed during definition. For example:

  ```
  CREATE TABLE t (a DATE DEFAULT to_date('2024-10-01'));
  ```

  The format of the preceding statement will not be verified by **nls_date_format** for table creation.

- For a value of the UNKNOWN, CHAR, CHARACTER, NCHAR, BPCHAR, VARCHAR, CHARACTER VARYING, VARCHAR2, NVARCHAR2, TEXT, or CLOB type in the numeric format, GaussDB supports a DATEA value plus (the left and right operands are interchangeable) or minus the value, but does not support the value minus a DATEA value. For a value of the CHAR, CHARACTER, NCHAR, VARCHAR, CHARACTER VARYING, VARCHAR2, or NVARCHAR2 type in numeric format, Oracle Database supports a DATE value plus (the left and right operands can be exchanged) or minus the value, but does not support the value minus a DATE value.
- For a value of the UNKNOWN, CHAR, CHARACTER, NCHAR, BPCHAR, VARCHAR, CHARACTER VARYING, VARCHAR2, NVARCHAR2, TEXT, or CLOB type in the time format, GaussDB supports the value minus a DATEA value, but does not support a DATEA value plus (the left and right operands can be exchanged) or minus the value. For a value of the CHAR, CHARACTER, NCHAR, VARCHAR, CHARACTER VARYING, VARCHAR2, or NVARCHAR2 type in the time format, Oracle Database supports neither a DATE value plus (the left and right operands can be exchanged) or minus the value, nor the value minus a DATE value.
- In GaussDB, only the **nls_date_format** parameter can control the DATEA format. In Oracle Database, the format can be controlled by **nls_date_format** or **nls_date_language**.
- In GaussDB, implicit conversion between DATEA and CLOB types is supported. Oracle Database does not support this feature in certain scenarios.
- When Oracle Database performs arithmetic operations (+ and –) between the date/time type and numeric type, the operand of the date/time type is implicitly converted to the DATE type, and the operation result is also of the DATE type. GaussDB complies with its own type promotion rules. The type of the operation result is the same as that of the left operand, and implicit downgrade to the DATEA type is not performed.
- GaussDB supports the comparison between the DATEA type and the INTERVAL type, numeric type, or RAW type, but Oracle Database does not. For details, see **Comparison Operators**.
- The behaviors of **nls_date_format** supported by GaussDB differ from that supported by Oracle Database in the following ways:
  1. GaussDB does not support E/EE, but Oracle Database supports.
  2. DL/DS/TS cannot be used for to_date in GaussDB, but they are supported in Oracle Database.
  3. An input string should not contain trailing spaces. For example, an error will be reported when **nls_date_format = 'yyyy-mm-dd';** is executed to convert the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

following string to the date type: **SELECT date '2024 -10-01' FROM dual;**. However, it is allowed in Oracle Database.

4. In Oracle Database, the input of the "text" separator requires verification; however, in GaussDB, only its output is affected, rather than its input.
5. FX and FM offer the same functions but have different behaviors in GaussDB and Oracle Database. FM suppresses leading zeros and trailing spaces for converting a character string. It can only be used as a prefix and applied in each basic element in GaussDB, which is different from the applicable scope in Oracle Database. FX precisely verifies whether the NLS format is used. In GaussDB, if FX appears once, all subsequent elements need to be precisely verified until FM is encountered, which is different from the applicable scope in Oracle Database.
6. GaussDB supports TH as the suffix, but it does not take effect in Oracle Database.
7. GaussDB does not support THSP and SPTH as the suffix. SP can be set as the suffix, but it does not take effect.
8. In GaussDB, prefixes and suffixes are not allowed to appear consecutively for multiple times within a formatted model element, such as FMFMYYYY or YYYYTHTH.

- The DATEA type in GaussDB supports only B-tree, UB-tree, PSORT, and CB-tree indexes, but does not support GIN, UGIN, and GiST indexes.
- The DATEA type in GaussDB supports only the JDBC, ODBC, and libpq drivers and does not support the Go and Python drivers.
- The DATEA type in GaussDB cannot be used as a range type.
- The arithmetic operators of the DATEA type in GaussDB (including subtraction between DATEA types and addition and subtraction between DATEA types and numeric types) use float8 (double-precision floating point) for intermediate calculation and then convert the calculation result to the output type. The valid precision of float8 is about 15 decimal digits. When the calculation result contains a large number of decimal places, the precision of the last digit may be lost. For example, when the result of subtraction between DATEA types is returned as a numeric type, the maximum number of valid digits is about 15. When the result of addition or subtraction between DATEA types and numeric types is returned as a DATEA type, the hour, minute, and second parts in the result may have microsecond-level precision deviation. In Oracle Database, the arithmetic operation of the DATE type returns the NUMBER type, and there is no such precision restriction. If high-precision date calculation is required, you are advised to use functions such as EXTRACT to extract time elements and then perform calculation.
- The gsql of GaussDB does not display DATEA data in the format specified by **nls_date_format**. Instead, it displays data in the fixed format 'YYYY-MM-DD HH24:MI:SS AD/BC'. The sqlplus of the Oracle Database displays DATE data in the format specified by **nls_date_format**.

- In ORA-compatible mode, the AT TIME ZONE syntax is supported, but its semantics are different from those in Oracle Database. In GaussDB, AT TIME ZONE is a type conversion operator. **timestamp AT TIME ZONE zone** interprets the timestamp without time zone as the UTC time point based on the specified time zone and returns the timestamptz type (converted to the time zone of the current session). **timestamptz AT TIME ZONE zone** converts the timestamp with time zone to the local time of the specified time zone and returns the timestamp type. However, in Oracle Database, AT TIME ZONE is an operator for reinterpreting time semantics. It keeps the return type unchanged (TIMESTAMP WITH TIME ZONE input still returns TIMESTAMP WITH TIME ZONE) and only displays the time value in the target time zone.
- In ORA-compatible mode, there are differences between GaussDB and Oracle Database when you use CREATE TABLE to create objects.

  GaussDB does not support the **REF** keyword, but Oracle Database supports it.

```
CREATE TABLE test_floor_ref (
  ref_id NUMBER,
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    col_ref REF obj_ref_test_floor
);
```

Table 3-3 Character types

Oracle Database	GaussDB	Difference
VARCHAR2 [(size [BYTE | CHAR])]	Supported, with differences.	In GaussDB, the unit of size is bytes, and the maximum value is **10M**. In Oracle Database, if **MAX_STRING_SIZE** is set to **EXTENDED**, the maximum length is 32767 bytes; if **MAX_STRING_SIZE** is set to **STANDARD**, the maximum length is 4000 bytes.
NVARCHAR2 (size)	Supported, with differences.	In GaussDB, the unit of size is bytes and the maximum value is **10M**. In Oracle Database, if **MAX_STRING_SIZE** is set to **EXTENDED**, the maximum length is 32767 bytes; if **MAX_STRING_SIZE** is set to **STANDARD**, the maximum length is 4000 bytes. For both databases, the actual number of characters that can be contained depends on the character set in use. In GaussDB, trailing spaces in an NVARCHAR2 value are ignored during insertion. In Oracle Database, trailing spaces in an NVARCHAR2 value are not ignored during insertion.
CHAR [(size [BYTE | CHAR])]	Supported, with differences.	In GaussDB, the unit of size is bytes, and the maximum value is **10MB**. In Oracle Database, the maximum length is 2000 bytes.
NCHAR [(size)]	Supported, with differences.	In GaussDB, the size unit is bytes and the maximum value is 10 MB. The maximum number of characters that can be contained depends on the used character set. In Oracle Database, the size unit is characters and the maximum capacity is 2000 bytes. NCHAR is a data type used to store fixed-length Unicode characters.
CLOB	Supported, with differences.	Locators are not supported.
NCLOB	Not supported.	-
LONG	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-4 Binary types

Oracle Database	GaussDB	Difference
RAW (size)	Supported, with differences.	In GaussDB, **size** indicates the recommended byte length and is not used to verify the byte length of the input raw type.
LONG RAW	Not supported.	-
BLOB	Supported.	-
BFILE	Not supported.	-

Table 3-5 ROWID types

Oracle Database	GaussDB	Difference
ROWID	Supported, with differences.	In GaussDB, the ROWID type is compared based on the values of the **node_slice**, **table_oid**, **bucket_id**, and **row_no** fields in sequence. In Oracle Database, the ROWID type is compared based on the values of the **data object number**, **relate file number**, **block number**, and **row number** fields in sequence. For details about the comparison rules of ROWID operators, see **Operators**.
UROWID	Not supported.	-

Table 3-6 User-defined types

Oracle Database	GaussDB
Object types	Not supported.
REF data types	Not supported.
Variable arrays	Supported.
Nested tables	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-7 Pseudo-types

Oracle Database	GaussDB
anytype	Not supported.
anydata	Not supported.
anydataset	Not supported.

Table 3-8 XML types

Oracle Database	GaussDB	Difference
XMLType	Supported, with differences.	GaussDB does not support some operations. For example, the XMLELEMENT function is used to convert a character string to the XML type instead of the XMLType type. For details, see **XMLType**.
URITYPE	Not supported.	-

Table 3-9 Spatial types

Oracle Database	GaussDB
SDO_GEOMETRY	Not supported.
SDO_TOPO_GEOMETRY	Not supported.
SDO_GEORASTER	Not supported.

Table 3-10 Lock modes

Oracle Database	GaussDB
none	-
null	AccessShare
RS	RowShare
RX	RowExclusive
S	ShareUpdateExclusive
SRX	Share
-	ShareRowExclusive
X	Exclusive

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
-	AccessExclusive
-	INVALID **NOTE** INVALID of GaussDB indicates that an invalid lock is assigned. An invalid lock is assigned only when a lock that cannot be identified by GaussDB occurs during system running.

3.2.2.2 Data Type Comparison Rules

Data type comparison (collation) rules apply only when values of the same data type are compared (collated).

Table 3-11 Comparison rules

Oracle Database	GaussDB	Difference
Numeric values	Supported.	-
Datetime values	Supported.	-
Binary values	Supported.	-
Character values	Supported, with differences.	• GaussDB and Oracle Database support different comparison rules, and the names of the same comparison rules may be different. • GaussDB and Oracle Database differ in specifying comparison rules. For example, table-level comparison rules cannot be specified in GaussDB, but can be specified in Oracle Database. • The syntax for specifying comparison rules in GaussDB and Oracle Database is different. For example, in GaussDB, the **ENCODING**, **LC_CTYPE**, and **LC_COLLATE** parameters are used to determine the character set, collation, and comparison rule used for creating a database. For details, see **CREATE DATABASE**. In Oracle Database, comparison rules at different levels are usually specified by a series of parameters with the NLS prefix.
Object values	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
Varrays and nested tables	Supported, with differences.	Both GaussDB and Oracle Database support the comparison of varrays. Different from Oracle Database, GaussDB not only supports the comparison of the number of elements in two varrays, but also supports the comparison between varrays of the same type.
Data type precedence	Supported.	-
Explicit/ Implicit data type conversion	Supported.	-

3.2.2.3 Literals

Table 3-12 Literals

Oracle Database	GaussDB
Text literals	Supported.
Numeric literals	Supported, with the following difference: Numeric literals of 123.4f and 123.4d (with the f/d suffix) cannot be used as function input parameters.
Datetime literals	Supported.
Interval literals	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.2.2.4 Format Models

Table 3-13 Formats

Oracle Database	GaussDB	Difference
Number formats	Supported, with differences.	GaussDB supports the $, C, TM, TM9, TME, and U formats only when **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1**. In addition, this parameter does not support the TH, PL, or SG format. For details about the support of GaussDB, see formats for the number type in **Type Conversion Functions**.
Datetime formats	Supported, with differences.	GaussDB: Parameters used for time truncation and rounding are valid only when **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1**. For details about the support of GaussDB, see formats for formatting date and time in **Date and Time Processing Functions and Operators**.
Format model modifiers	Supported.	-
String-to-date conversion rules	Supported, with differences.	GaussDB: The to_timestamp_tz function is valid only when **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1**. For details about the support of GaussDB, see "to_date/to_timestamp/to_timestamp_tz" in **Type Conversion Functions**.
XML format models	Not supported.	-

3.2.2.5 Nulls

Table 3-14 Nulls

Oracle Database	GaussDB
IS NULL and IS NOT NULL	Supported.
NULLS in conditions	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.2.2.6 Comments

Table 3-15 Comments

Oracle Database	GaussDB	Difference
A slash and an asterisk (/*)	Supported.	-
Two hyphens (--)	Supported.	-
COMMENT command	Supported.	-
HINT	Supported, with differences.	GaussDB does not support the '--+' hint format. For details, see "Hint-based Optimization" in *SQL Optimization Guide*.

3.2.2.7 Database Objects

Table 3-16 Schema objects

Oracle Database	GaussDB	Difference
Analytic views	Not supported.	-
Attribute dimensions	Not supported.	-
Clusters	Supported.	-
Constraints	Supported.	-
Database links	Supported.	-
Database triggers	Supported.	-
Dimensions	Supported.	-
External procedure libraries	Not supported.	-
Hierarchies	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
Index-organized tables	Not supported.	-
Indexes	Supported.	If the unique index involves composite columns, the processing logic is NULL == NULL in Oracle Database, but it is NULL != NULL in GaussDB, which follows the SQL standard by default. To make the latter compatible with the former in terms of this behavior, enable the GUC parameter **enable_uniq_idx_a_compat**. This parameter is disabled by default in GaussDB ORA-compatible mode.
Index types	Not supported.	-
Java classes	Not supported.	-
Java resources	Not supported.	-
Java source code	Not supported.	-
Join groups	Not supported.	-
Materialized views	Supported.	-
Materialized view logs	Not supported.	-
Mining models	Not supported.	-
Object tables	Not supported.	-
Object types	Not supported.	-
Object views	Not supported.	-
Operators	Supported.	-
Packages	Supported.	-
Sequences	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
Storage functions	Supported.	-
Stored procedures	Supported.	-
Synonyms	Supported, with differences.	The names of Oracle Database objects in the same namespace must be unique. In GaussDB, the name of a synonym can be the same as that of a table, view, function, or package in the same namespace. In this case, GaussDB preferentially accesses the table, view, function, or package object with the same name. If no such object is found, GaussDB searches for the synonym object. In addition, the PUBLIC synonym is searched only when the schema name of the object to which the synonym points is a username. For details about the search sequence, see the precautions in **CREATE SYNONYM**. The search process of synonyms is as follows:
Tables	Supported.	-
Views	Supported.	-
Zone map	Not supported.	-

Table 3-17 Non-schema objects

Oracle Database	GaussDB
Contexts	Not supported.
Directories	Supported.
Editions	Not supported.
Flashback archives	Not supported.
Lockdown profiles	Not supported.
Profiles	Not supported.
Restore points	Supported.
Roles	Supported.
Rollback segments	• Ustore supports rollback segments. • Astore does not support rollback segments.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
Tablespaces	Supported.
Tablespace sets	Not supported.
Unified audit policies	Supported.
Users	Supported.

3.2.2.8 Database Object Names and Qualifiers

Table 3-18 Naming rules

Oracle Database	GaussDB	Difference
Database object naming rules	Supported, with differences.	GaussDB uses lowercase letters by default.
Schema object naming rules	Supported.	-

3.2.2.9 Syntax for Schema Objects and Parts in SQL Statements

Table 3-19 Object reference

Oracle Database	GaussDB
General syntax for referring to an object	Supported.
Resolving a reference to an object	Supported.
Referring to objects in other schemas	Supported.
Referring to objects in remote databases	Supported.
Referring to partitions and subpartitions of tables and indexes	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.2.3 Pseudocolumns

Hierarchical Query Pseudocolumns

Table 3-20 Hierarchical query pseudocolumns

Oracle Database	GaussDB
connect_by_iscycle	Supported.
connect_by_isleaf	Supported.
LEVEL pseudocolumn	Supported.

Sequence Pseudocolumns

Table 3-21 Sequences

Oracle Database	GaussDB	Difference
currval	Supported, with differences.	It is implemented as a function in GaussDB. The call mode is compatible with Oracle Database.
nextval	Supported, with differences.	It is implemented as a function in GaussDB. The call mode is compatible with Oracle Database.

ROWID Pseudocolumns

Table 3-22 rowid

Oracle Database	GaussDB	Difference
rowid	Supported, with differences.	Oracle Database supports ROWID pseudocolumns when a table is created. GaussDB supports ROWID pseudocolumns when the **hasrowid** option is used to create a table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

ROWNUM Pseudocolumn

Table 3-23 rownum

Oracle Database	GaussDB	Difference
rownum	Supported, with differences.	When Oracle Database uses rownum in the left, right, and full join conditions for filtering, the performance varies according to the conditions. The rownum condition may be ignored or partially ignored. However, GaussDB filters the results after left, right, and full join.

XMLDATA Pseudocolumn

Table 3-24 xmldata

Oracle Database	GaussDB
xmldata	Not supported.

3.2.4 Operators

GaussDB is compatible with operators except hierarchical query.

SQL Operators

Table 3-25 SQL operators

Oracle Database	GaussDB
Unary and binary operators	Supported.
Operator precedence	Supported.

Arithmetic Operators

Table 3-26 Arithmetic operators

Oracle Database	GaussDB
Unary operators: positive (+) and negative (–).	Supported.
Binary operators: addition (+) and subtraction (–).	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
Binary operators: multiplication (*) and division (/).	Supported.

COLLATE Operator

Table 3-27 COLLATE operator

Oracle Database	GaussDB
COLLATE collation_name	Supported.

Connection Operators

Table 3-28 Connection operators

Oracle Database	GaussDB	Difference
||	Supported, but with differences.	For the concatenation of invalid characters, GaussDB does not check the validity of characters and directly concatenates all characters; however, if Oracle Database detects invalid characters, it ignores them. Example: -- GaussDB gaussdb=# SET a_format_version='10c'; SET gaussdb=# SET a_format_dev_version='s2'; SET gaussdb=# SELECT dump(CHR(239) || 'test') FROM dual; dump ---------- Typ=25 Len=5: 239,116,101,115,116 (1 row) -- Oracle SQL> SELECT dump(CHR(239) || 'test') FROM dual; dump(CHR(239)||'TEST') ---------- Typ=1 Len=4: 116,101,115,116

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Hierarchical Query Operators

Table 3-29 Hierarchical query operators

Oracle Database	GaussDB	Difference
prior	Supported, with differences.	GaussDB: Only ordinary columns can be called. Functions cannot be called.
connect_by_root	Supported, with differences.	GaussDB: When connect_by_root is called, if parentheses are used to modify the operation value, the behavior is the same as that of Oracle Database. If parentheses are not used, this operator can be called only for ordinary columns.

Set Operators

Table 3-30 Set operators

Oracle Database	GaussDB
union	Supported.
union all	Supported.
intersect	Supported.
minus	Supported.

Multiset Operators

Table 3-31 Multiset operators

Oracle Database	GaussDB
multiset except	Supported.
multiset intersect	Supported.
multiset union	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

User-defined Operators

Table 3-32 User-defined operators

Oracle Database	GaussDB
CREATE OPERATOR	Not supported.

Comparison Operators

Oracle Database	GaussDB
< =	Supported.
< >	Supported.
> =	Supported.
^ =	Supported.
! =	Not supported. For !=, if there is a space between an exclamation mark (!) and an equal sign (=), the exclamation mark will be identified as factorial.

For comparison operators <=, <>, >=, and ^=, if there is a space between two symbols, it does not affect normal operations. For !=, if there is a space between an exclamation mark (!) and an equal sign (=), the exclamation mark will be identified as factorial, which may cause the result to be inconsistent with the expected result.

3.2.5 Expressions

GaussDB is compatible with most database expressions.

Table 3-33 Expressions

Oracle Database	GaussDB	Difference
Simple expressions	Supported.	-
Analytic view expressions	Not supported.	-
Compound expressions	Supported.	-
CASE expressions	Supported.	-
Column expressions	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
CURSOR expressions	Not supported.	-
Datetime expressions	Supported, with differences.	GaussDB command output does not contain time zone information, but Oracle Database contains time zone information similar to "PM AMERICA/LOS_ANGELES."
Function expressions	Supported.	-
Interval expressions	Partially supported.	GaussDB supports statements in the format of "SELECT INTERVAL '999999999 23:59:59.999' day(9) to second FROM DUAL;" but does not support statements in the format of "SELECT(SYSDATE-SYSDATE) DAY TO SECOND FROM DUAL;". They are supported in Oracle Database.
JSON object access expressions	Partially supported, with differences.	• GaussDB can extract values from JSON objects in "->'key'" mode, while Oracle Database can extract values in ".key" mode. • For JSON array objects, Oracle Database can extract values corresponding to all keys at a time in ".key" mode. However, GaussDB does not support this option.
Model expressions	Not supported.	-
Object expressions	Not supported.	-
Placeholder expressions	Partially supported.	GaussDB supports general placeholder expressions such as ":var", but does not support the combination of two general placeholder expressions using the INDICATOR keyword.
Scalar subquery expressions	Supported.	-
Type constructor expressions	Partially supported.	GaussDB cannot specify the NEW keyword before the type constructor, but Oracle Database can.
Expression lists	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

<table>
<tr><th>Oracle Database</th><th>GaussDB</th><th>Difference</th></tr>
<tr><td>Expressions with the addition (+) operator</td><td>Partially supported, with differences.</td><td>For an outer join expression connected by the OR operator, GaussDB supports only the outer join expression in which the same single table is used on the left and right sides of the OR operator.
When the addition (+) and OR operators are used together, the behavior of GaussDB is different from that of Oracle Database. The following describes the differences between GaussDB and Oracle Database 19c:
Common statements
-- Run the following statements to create tables and insert data into the tables:
CREATE TABLE a(id int, b int);
CREATE TABLE a2(id int, c int);
INSERT INTO a VALUES(1,1);
INSERT INTO a VALUES (null, 11);
INSERT INTO a VALUES (2, null);
INSERT INTO a VALUES (123,123);
INSERT INTO a VALUES (5, 500);
INSERT INTO a VALUES (null, 600);
INSERT INTO a2 VALUES(1,2);
INSERT INTO a2 VALUES (3,null);
INSERT INTO a2 VALUES (null, 22);
INSERT INTO a2 VALUES (234,234);
INSERT INTO a2 VALUES (5, 700);
INSERT INTO a2 VALUES (null, 800);
• Example 1
SELECT * FROM a,a2 WHERE (a.id(+) = 1 AND a.id(+) <= 2) OR a2.id(+) = 1 ORDER BY 1,2,3,4;
– GaussDB
gaussdb=# CREATE DATABASE gaussdb_o DBCOMPATIBILITY = 'ORA';
gaussdb=# \c gaussdb_o
-- Execute the common statement and then the query statement.
gaussdb_o=# SELECT * FROM a,a2 WHERE (a.id(+) = 1 AND a.id(+) <= 2) OR a2.id(+) = 1 ORDER BY 1,2,3,4;
ERROR: Operator "(+)" is not allowed used with "OR" together
-- Switch to the default database and drop the gaussdb_o database. Change the database name as required.
gaussdb_o=# \c postgres
gaussdb=# DROP DATABASE gaussdb_o;
– Oracle
-- Execute the common statement and then the query statement.
SQL> SELECT * FROM a,a2 WHERE (a.id(+) = 1 AND a.id(+) <= 2) OR a2.id(+) = 1 ORDER BY 1,2,3,4;

ID| B| ID| C
----------|----------|----------|----------
1| 1| 1| 2
1| 1| 3|</td></tr>
</table>

Oracle Database	GaussDB	Difference

		1| 1| 5| 700
		1| 1| 234| 234
		1| 1| | 22
		1| 1| | 800
		2| | 1| 2
		5| 500| 1| 2
		123| 123| 1| 2
		| 11| 1| 2
		| 600| 1| 2
		
		11 rows selected.
		-- Drop the data tables created in this example. (You need to modify the table names based on the actual situation.)
		SQL> DROP TABLE a;
		SQL> DROP TABLE a2;
		When the left and right sides of the OR operator in the outer join expression use different single tables, GaussDB throws an error because this breaks its rule. However, Oracle Database allows it without reporting an error.
		• Example 2
		SELECT * FROM a,a2 WHERE a.id(+) = a2.id AND a.id(+) != 2 OR a.id(+) IS NULL AND a.id(+) = a2.id ORDER BY 1,2,3,4;
		– GaussDB
		gaussdb=# CREATE DATABASE gaussdb_o DBCOMPATIBILITY = 'ORA';
		gaussdb=# \c gaussdb_o
		-- Execute the common statement and then the query statement.
		gaussdb_o=# SELECT * FROM a,a2 WHERE a.id(+) = a2.id OR a.id IS NULL AND a.id(+) = a2.id ORDER BY 1,2,3,4;
		id | b | id | c
		----+-----+-----+-----
		1 | 1 | 1 | 2
		5 | 500 | 5 | 700
		| | 3 |
		| | 234 | 234
		| | | 22
		| | | 800
		(6 rows)
		-- After extracting the common expression, execute the converted SQL statement.
		gaussdb_o=# SELECT * FROM a,a2 WHERE a.id(+) = a2.id AND (a.id(+) != 2 OR a.id(+) IS NULL) ORDER BY 1,2,3,4;
		id | b | id | c
		----+-----+-----+-----
		1 | 1 | 1 | 2
		5 | 500 | 5 | 700
		| | 3 |
		| | 234 | 234
		| | | 22
		| | | 800
		(6 rows)
		-- Switch to the default database and drop the *gaussdb_o* database. Change the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
		database name as required. gaussdb_o=# \c postgres gaussdb=# DROP DATABASE gaussdb_o; – Oracle -- Execute the common statement and then the query statement. SQL> SELECT * FROM a,a2 WHERE a.id(+) = a2.id OR a.id IS NULL AND a.id(+) = a2.id ORDER BY 1,2,3,4; ERROR at line 1: ORA-01719: outer join operator (+) not allowed in operand of OR or IN -- After extracting the common expression, execute the converted SQL statement. SQL> SELECT * FROM a,a2 WHERE a.id(+) = a2.id AND (a.id(+) != 2 OR a.id(+) IS NULL) ORDER BY 1,2,3,4; ID| B| ID| C ----------|----------|----------|---------- 1| 1| 1| 2 5| 500| 5| 700 | | 3| | | 234| 234 | | | 22 | | | 800 6 rows selected. -- Drop the data tables created in this example. (You need to modify the table names based on the actual situation.) SQL> DROP TABLE a; SQL> DROP TABLE a2; When the left and right sides of the OR operator in the outer join expression use the same single table, GaussDB does not throw an error because this complies with its rule. However, Oracle Database reports an error (ORA-01719). After extracting the common expression a.id(+) = a2.id, the converted SQL statement is as follows: SELECT * FROM a,a2 WHERE a.id(+) = a2.id AND (a.id(+) != 2 OR a.id(+) IS NULL) ORDER BY 1,2,3,4; The execution results are the same in the two databases.

3.2.6 Conditions

This chapter describes common compatible conditions. The conditions include comparison, floating-point, logical, model, multiset, pattern matching, **NULL** value, XML, SQL/JSON, composite, BETWEEN, EXISTS, IN, and IS OF TYPE, as shown in **Table 3-34**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-34 Conditions

Oracle Database	GaussDB	Difference
Comparison conditions	Supported, with differences.	Differences exist when statements contain the ANY, SOME, and ALL operators. Oracle Database supports operations on list objects, but GaussDB needs to convert list objects into array expressions before performing operations.
Floating-point conditions	Not supported.	-
Logical conditions	Supported.	-
Model conditions	Not supported.	-
Multiset conditions	Not supported.	-
Pattern-matching conditions	Supported.	-
NULL conditions	Supported.	-
XML conditions	Not supported.	-
SQL/JSON conditions	Partially supported, with differences.	• GaussDB does not support the IS JSON and JSON_TEXTCONTAINS conditions. • The JSONB_EQ condition in GaussDB is the same as the JSON_EQUAL condition in Oracle Database. However, GaussDB does not support the ERROR clause. • The JSONB_EXISTS condition in GaussDB is the same as the JSON_EXISTS condition in Oracle Database. However, GaussDB does not support the ERROR, EMPTY, or PASSING clauses.
Compound conditions	Supported.	-
BETWEEN condition	Supported.	-
EXISTS condition	Supported.	-
IN condition	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
IS OF TYPE condition	Not supported.	-

3.2.7 Drivers

3.2.7.1 JDBC

3.2.7.1.1 Array

This section describes the differences between Oracle Database and GaussDB when the JDBC driver of the java.sql.Array type is used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-35 Constructor reference

Constructor	Oracle Database	GaussDB	API Difference
1. Use the static constructor of ArrayDescriptor to construct an ArrayDescriptor object. 2. Use ArrayDescriptor to construct an array object.	String typeName = "XXX"; Connection conn = getConnection(); Object[] elements = null; ArrayDescriptor desc = ArrayDescriptor.createDescriptor(typeName, conn); Array array = new ARRAY(desc, conn, elements);	String typeName = "xxx"; Connection conn = getConnection(); Object[] elements = null; ArrayDescriptor desc = ArrayDescriptor.getDescriptor(typeName, conn); Array array = new GaussArray(desc, elements);	• Different names of the static constructor for ArrayDescriptor: In Oracle Database, it is createDescriptor, while in GaussDB it is getDescriptor. • Different constructor names for arrays: In Oracle Database, it is ARRAY(ArrayDescriptor, Connection, Object), while in GaussDB, it is GaussArray(ArrayDescriptor, Object). • Variable description: *typeName* indicates the type name and is case-sensitive: uppercase in Oracle Database, while lowercase in GaussDB. *conn* indicates the connection object for the corresponding database. *elements* indicates the element data of the corresponding type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

1. GaussDB currently does not support the constructors not listed in the aforementioned table.
2. If the element type is the character type and the length of the construction input string exceeds that defined by the element type, Oracle Database reports an error during input parameter binding.

   If the array type is the varray type and the number of elements exceeds the maximum length of varray, Oracle Database reports an error during input parameter binding.

   GaussDB does not verify type modifiers when constructing or binding input parameters. When receiving array objects and executing SQL statements, a database decides whether to report an error.

Table 3-36 API reference

Method	Return Type	Throws	GaussDB
getBaseTypeName()	String	SQLException	Supported.
getBaseType()	int	SQLException	Supported.
getArray()	Object	SQLException	Supported.
getArray(java.util.Map<String,Class<?>> map)	Object	SQLException	Not supported.
getArray(long index, int count)	Object	SQLException	Supported.
getArray(long index, int count, java.util.Map<String,Class<?>> map)	Object	SQLException	Not supported.
getResultSet()	ResultSet	SQLException	Not supported.
getResultSet(java.util.Map<String,Class<?>> map)	ResultSet	SQLException	Not supported.
getResultSet(long index, int count)	ResultSet	SQLException	Not supported.
getResultSet (long index, int count, java.util.Map<String,Class<?>> map)	ResultSet	SQLException	Not supported.
free()	void	SQLException	Not supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-37 Differences in the getArray() API

Element Database Type	Actual Return Type of the getArray API (Oracle Database)	Actual Return Type of the getArray API (GaussDB)
CHAR	java.lang.String[]	java.lang.String[]
VARCHAR/ VARCHAR2	java.lang.String[]	java.lang.String[]
NCHAR	java.lang.String[]	java.lang.String[]
NVARCHAR2	java.lang.String[]	java.lang.String[]
NUMBER	java.math.BigDecimal[]	java.math.BigDecimal[]
NUMERIC	java.math.BigDecimal[]	java.math.BigDecimal[]
DECIMAL	java.math.BigDecimal[]	java.math.BigDecimal[]
INTEGER	java.math.BigDecimal[]	java.lang.Integer[]
SMALLINT	java.math.BigDecimal[]	java.lang.Short[]
DOUBLE PRECISION	java.math.BigDecimal[]	java.lang.Double[]
FLOAT	java.math.BigDecimal[]	java.lang.Double[]
REAL	java.math.BigDecimal[]	java.lang.Float[]
BINARY_DOUBLE	java.lang.Double[]	java.lang.Double[]
BINARY_INTEGER	java.math.BigDecimal[]	java.lang.Integer[]
BOOLEAN	java.math.BigDecimal[]	java.lang.Boolean[]
TIMESTAMP	java.sql.Timestamp[]	java.sql.Timestamp[]
TIMESTAMP WITH TIME ZONE	java.time.OffsetDateTime[]	java.sql.Timestamp[]
BLOB	oracle.sql.BLOB[]	java.sql.Blob[]
CLOB	oracle.sql.CLOB[]	java.sql.Clob[]
Set/Array	java.lang.Object[]	java.sql.Array[]
RECORD	java.lang.Object[]	java.sql.Struct[]

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

1. GaussDB currently does not support the types not listed in the aforementioned table.
2. For details about the differences in the return values of the getArray(long index, int count) API, see the preceding table.
3. The differences in the **index** parameter of getArray(long index, int count) are as follows:
   - The value range supported by Oracle Database is [1, *Long.MAX_VALUE*]. GaussDB supports a range of [1, *Integer.MAX_VALUE*].
   - If the **index** value is greater than that of *Integer.MAX_VALUE*, it will be truncated in Oracle Database, while an error is reported in GaussDB.

Table 3-38 Differences in the getBaseType() API

Element Database Type	Return Value of getBaseType (Oracle Database)	Return Value of getBaseType (GaussDB)
CHAR	java.sql.Types.CHAR	java.sql.Types.CHAR
VARCHAR/ VARCHAR2	java.sql.Types.VARCHAR	java.sql.Types.VARCHAR
NCHAR	java.sql.Types.NCHAR	java.sql.Types.CHAR
NVARCHAR2	java.sql.Types.NVARCHAR	java.sql.Types.VARCHAR
NUMBER	java.sql.Types.NUMERIC	java.sql.Types.NUMERIC
NUMERIC	java.sql.Types.DECIMAL	java.sql.Types.NUMERIC
DECIMAL	java.sql.Types.DECIMAL	java.sql.Types.NUMERIC
INTEGER	java.sql.Types.NUMERIC	java.sql.Types.INTEGER
SMALLINT	java.sql.Types.NUMERIC	java.sql.Types.SMALLINT
DOUBLE PRECISION	java.sql.Types.FLOAT	java.sql.Types.DOUBLE
FLOAT	java.sql.Types.FLOAT	java.sql.Types.DOUBLE
REAL	java.sql.Types.FLOAT	java.sql.Types.REAL
BINARY_DOUBLE	oracle.jdbc.OracleTypes.BINARY_DOUBLE	java.sql.Types.DOUBLE
BINARY_INTEGER	java.sql.Types.NUMERIC	java.sql.Types.INTEGER
BOOLEAN	java.sql.Types.NUMERIC	java.sql.Types.BIT
TIMESTAMP	java.sql.Types.TIMESTAMP	java.sql.Types.TIMESTAMP
TIMESTAMP WITH TIME ZONE	oracle.jdbc.OracleTypes.TIMESTAMPTZ	java.sql.Types.TIMESTAMP
BLOB	java.sql.Types.BLOB	java.sql.Types.BLOB

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Element Database Type	Return Value of getBaseType (Oracle Database)	Return Value of getBaseType (GaussDB)
CLOB	java.sql.Types.CLOB	java.sql.Types.CLOB
Set/Array	java.sql.Types.ARRAY	java.sql.Types.ARRAY
RECORD	java.sql.Types.STRUCT	java.sql.Types.STRUCT

NOTE

GaussDB currently does not support the types not listed in the aforementioned table.

Table 3-39 Differences in the getBaseTypeName() API

Element Database Type	Return Value of getBaseTypeName (Oracle Database)	Return Value of getBaseTypeName (GaussDB)
CHAR	"CHAR"	"bpchar"
VARCHAR/ VARCHAR2	"VARCHAR"	"varchar"
NCHAR	"NCHAR"	"bpchar"
NVARCHAR2	"NVARCHAR"	"nvarchar2"
NUMBER	"NUMBER"	"numeric"
NUMERIC	"DECIMAL"	"numeric"
DECIMAL	"DECIMAL"	"numeric"
INTEGER	"NUMBER"	"int4"
SMALLINT	"NUMBER"	"int2"
DOUBLE PRECISION	"FLOAT"	"float8"
FLOAT	"FLOAT"	"float8"
REAL	"FLOAT"	"float4"
BINARY_DOUBLE	"BINARY_DOUBLE"	"float8"
BINARY_INTEGER	"NUMBER"	"int4"
BOOLEAN	"NUMBER"	"bool"
TIMESTAMP	"TIMESTAMP"	"timestamp"
TIMESTAMP WITH TIME ZONE	"TIMESTAMP WITH TIME ZONE"	"timestamptz"

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Element Database Type	Return Value of getBaseTypeName (Oracle Database)	Return Value of getBaseTypeName (GaussDB)
BLOB	"BLOB"	"blob"
CLOB	"CLOB"	"clob"
Set/Array	See the description below.	See the description below.
RECORD	See the description below.	See the description below.

NOTE

1. GaussDB currently does not support the types not listed in the aforementioned table.
2. When an element is of the set, array, or RECORD type that is defined within a package, the return rules for getBaseTypeName are as follows:
   - OJDBC11 returns *schemaName.packageName.typeName*.
   - OJDBC8 generally returns *schemaName.packageName.typeName*, or returns *"schemaName"."packageName.typeName"* in the following condition:

     Any of *schemaName*, *packageName*, or *typeName* violates the rule of starting with a letter and containing only letters, digits, and underscores (_).
   - GaussDB generally returns *schemaName.packageName.typeName*, or returns *"schemaName"."packageName"."typeName"* in the following condition:

     Any of *schemaName*, *packageName*, or *typeName* violates the rule of starting with a letter or underscore (_) and containing only letters, digits, and underscores.
3. When an element is of the set, array, or RECORD type that is defined in the schema (but not in the package), the return rules for getBaseTypeName are as follows:
   - OJDBC11 returns *schemaName.typeName*.
   - OJDBC8 generally returns *schemaName.typeName*, or returns *"schemaName"."typeName"* in the following condition:

     Either of *schemaName* or *typeName* violates the rule of starting with a letter and containing only letters, digits, and underscores (_).
   - GaussDB generally returns *schemaName.typeName*, or returns *"schemaName"."typeName"* in the following condition:

     Either *schemaName* or *typeName* violates the rule of starting with a letter or underscore (_) and containing only letters, digits, and underscores.
4. If no special processing is performed during element type creation, the getBaseTypeName API typically returns the type name in uppercase in Oracle Database, while returns the type name in lowercase in GaussDB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-40 Differences in array construction APIs

Element Database Type	Java Types Supported by Element Input Parameters (Oracle Database OJDBC8)	Java Types Supported by Element Input Parameters (GaussDB)	Difference
CHAR	Any Java type	Byte, Short, Integer, Long, BigInteger, BigDecimal, Float, Double, Character, Boolean, java.sql.Date, java.sql.Time, java.sql.Timestamp, and PGClob	The different types supported by element input parameters are listed in the table.
VARCHAR/ VARCHAR2	Any Java type	Byte, Short, Integer, Long, BigInteger, BigDecimal, Float, Double, Character, Boolean, java.sql.Date, java.sql.Time, java.sql.Timestamp, and PGClob	The different types supported by element input parameters are listed in the table.
NCHAR	Any Java type	Byte, Short, Integer, Long, BigInteger, BigDecimal, Float, Double, Character, Boolean, java.sql.Date, java.sql.Time, java.sql.Timestamp, and PGClob	The different types supported by element input parameters are listed in the table.
NVARCHAR2	Any Java type	Byte, Short, Integer, Long, BigInteger, BigDecimal, Float, Double, Character, Boolean, java.sql.Date, java.sql.Time, java.sql.Timestamp, and PGClob	The different types supported by element input parameters are listed in the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Element Database Type	Java Types Supported by Element Input Parameters (Oracle Database OJDBC8)	Java Types Supported by Element Input Parameters (GaussDB)	Difference
NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by element input parameters are listed in the table. 2. When the input parameter is of the Float, Double, BigDecimal, or String type and the decimal part is 0, Oracle Database truncates the decimal part, while GaussDB retains it.
NUMERIC	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by element input parameters are listed in the table. 2. When the input parameter is of the Float, Double, BigDecimal, or String type and the decimal part is 0, Oracle Database truncates the decimal part, while GaussDB retains it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Element Database Type	Java Types Supported by Element Input Parameters (Oracle Database OJDBC8)	Java Types Supported by Element Input Parameters (GaussDB)	Difference
DECIMAL	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by element input parameters are listed in the table. 2. When the input parameter is of the Float, Double, BigDecimal, or String type and the decimal part is 0, Oracle Database truncates the decimal part, while GaussDB retains it.
INTEGER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by element input parameters are listed in the table. 2. The target types are inconsistent. For details, see differences in the getArray() API. 3. When the input parameter value exceeds the integer range, GaussDB reports an error.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Element Database Type	Java Types Supported by Element Input Parameters (Oracle Database OJDBC8)	Java Types Supported by Element Input Parameters (GaussDB)	Difference
SMALLINT	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by element input parameters are listed in the table. 2. The target types are inconsistent. For details, see differences in the getArray() API. 3. When the input parameter exceeds the Short range, GaussDB reports an error.
DOUBLE PRECISION	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by element input parameters are listed in the table. 2. The target types are inconsistent. For details, see differences in the getArray() API. 3. Converting a higher-precision type to Double may result in precision loss.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Element Database Type	Java Types Supported by Element Input Parameters (Oracle Database OJDBC8)	Java Types Supported by Element Input Parameters (GaussDB)	Difference
FLOAT	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by element input parameters are listed in the table. 2. The target types are inconsistent. For details, see differences in the getArray() API. 3. Converting a higher-precision type to Double may result in precision loss.
REAL	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by element input parameters are listed in the table. 2. The target types are inconsistent. For details, see differences in the getArray() API. 3. Converting a higher-precision type to Float may result in precision loss.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Element Database Type	Java Types Supported by Element Input Parameters (Oracle Database OJDBC8)	Java Types Supported by Element Input Parameters (GaussDB)	Difference
BINARY_DOUBLE	Double and oracle.sql.BINARY_DOUBLE	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by element input parameters are listed in the table. 2. The target types are inconsistent. For details, see differences in the getArray() API. 3. Converting a higher-precision type to Double may result in precision loss.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Element Database Type	Java Types Supported by Element Input Parameters (Oracle Database OJDBC8)	Java Types Supported by Element Input Parameters (GaussDB)	Difference
BINARY_INTEGER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, AtomicInteger, AtomicLong, DoubleAccumulator, DoubleAdder, LongAccumulator, LongAdder, Striped64, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by element input parameters are listed in the table. 2. The input parameters of the OJDBC11 element support the following types: Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER. 3. The target types are inconsistent. For details, see differences in the getArray() API. 4. When the input value exceeds the Integer range, GaussDB reports an error. Oracle Database OJDBC8 performs data truncation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Element Database Type	Java Types Supported by Element Input Parameters (Oracle Database OJDBC8)	Java Types Supported by Element Input Parameters (GaussDB)	Difference
BOOLEAN	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, AtomicInteger, AtomicLong, DoubleAccumulator, DoubleAdder, LongAccumulator, LongAdder, Striped64, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by element input parameters are listed in the table. 2. The input parameters of the OJDBC8 element support the following types: Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER. 3. The target types are inconsistent. For details, see differences in the getArray() API. 4. In GaussDB, the target data type is Boolean, which only supports inputs of **1**, **0**, **"true"**, and **"false"**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Element Database Type	Java Types Supported by Element Input Parameters (Oracle Database OJDBC8)	Java Types Supported by Element Input Parameters (GaussDB)	Difference
TIMESTAMP	byte[], java.sql.Date, Calendar, java.util.Date, LocalDate, LocalDateTime, LocalTime, OffsetDateTime, OffsetTime, String, java.sql.Time, java.sql.Timestamp, oracle.sql.DATE, oracle.sql.TIMESTAMP, oracle.sql.TIMESTAMPTZ, oracle.sql.TIMESTAMPLTZ, and ZonedDateTime	java.util.Date, java.sql.Date, java.sql.Time, java.sql.Timestamp, LocalDateTime, and String	The different types supported by element input parameters are listed in the table.
TIMESTAMP WITH TIME ZONE	java.sql.Date, Calendar, java.util.Date, LocalDate, LocalDateTime, LocalTime, OffsetDateTime, OffsetTime, String, java.sql.Time, java.sql.Timestamp, oracle.sql.DATE, oracle.sql.TIMESTAMP, oracle.sql.TIMESTAMPTZ, oracle.sql.TIMESTAMPLTZ, and ZonedDateTime	java.util.Date, java.sql.Date, java.sql.Time, java.sql.Timestamp, LocalDateTime, and String	1. The different types supported by element input parameters are listed in the table. 2. The target types are inconsistent. For details, see differences in the getArray() API.
BLOB	oracle.sql.BLOB and oracle.jdbc.driver.OracleBlob	PGBlob	The different types supported by element input parameters are listed in the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Element Database Type	Java Types Supported by Element Input Parameters (Oracle Database OJDBC8)	Java Types Supported by Element Input Parameters (GaussDB)	Difference
CLOB	oracle.sql.CLOB, oracle.jdbc.driver.OracleClob, InputStream, and Reader	PGClob	The different types supported by element input parameters are listed in the table.
Set/Array	Array and Object	GaussArray and Object	1. The different types supported by element input parameters are listed in the table. 2. Oracle Database: No error is reported when the input parameter is of the Array type even if the Array type differs from the element type. GaussDB: An error is reported when the input parameter is of the GaussArray type that differs from the element type. 3. When the input parameter of an element is of the Object type, refer to the differences in array construction APIs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Element Database Type	Java Types Supported by Element Input Parameters (Oracle Database OJDBC8)	Java Types Supported by Element Input Parameters (GaussDB)	Difference
RECORD	Struct and Object[]	GaussStruct and Object[]	1. The different types supported by element input parameters are listed in the table. 2. Oracle Database: No error is reported when the input parameter is of the Struct type even if the Struct type differs from the element type. GaussDB: An error is reported when the input parameter is of the GaussStruct type that differs from the element type. 3. When the input parameter of an element is of the Object[] type, refer to the differences in struct construction APIs.

NOTE

1. When an array is constructed, if the Java type of the input element does not match the target type, an implicit conversion operation is performed. For details on the Java types of input elements supported by various database element types, refer to the preceding table.
2. GaussDB currently does not support the types not listed in the aforementioned table.
3. The constructor needs to provide an element array. The preceding table describes the differences between elements in the array.

3.2.7.1.2 Struct

This section describes the differences between Oracle Database and GaussDB when the JDBC driver of the java.sql.Struct type is used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-41 Constructor reference

Constructor	Oracle Database	GaussDB	API Difference
1. Use the static constructor of StructDescriptor to construct a StructDescriptor object. 2. Use StructDescriptor to construct a struct object.	String typeName = "XXX"; Connection conn = getConnection(); Object[] attributes = null; StructDescriptor desc = StructDescriptor.createDescriptor(typeName, conn); Struct struct = new STRUCT(desc, conn, attributes);	String typeName = "xxx"; Connection conn = getConnection(); Object[] elements = null; StructDescriptor desc = StructDescriptor.getDescriptor(typeName, conn); Struct struct = new GaussStruct(desc, attributes);	• Different names of the static constructor for StructDescriptor: In Oracle Database, it is createDescriptor, while in GaussDB it is getDescriptor. • Different names of the Struct constructor: In Oracle Database, it is STRUCT(StructDescriptor, Connection, Object[]), while in GaussDB it is GaussStruct(StructDescriptor, Object[]). • Variable description: *typeName* indicates the type name and is case-sensitive: uppercase in Oracle Database, while lowercase in GaussDB. *conn* indicates the connection object for the corresponding database. *attributes* indicates the element data array.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Constructor	Oracle Database	GaussDB	API Difference
Use the createStruct standard API of Connection to construct a struct object.	String typeName = "XXX"; Connection conn = getConnection(); Object[] attributes = null; Struct struct = conn.createStruct(typeName, attributes);	String typeName = "XXX"; Connection conn = getConnection(); Object[] attributes = null; Struct struct = conn.createStruct(typeName, attributes);	• Variable description: *typeName* indicates the type name and is case-sensitive: uppercase in Oracle Database, while lowercase in GaussDB. *conn* indicates the connection object for the corresponding database. *attributes* indicates the element data array.

NOTE

1. GaussDB currently does not support the constructors not listed in the aforementioned table.
2. If **attribute** is of the character type and the length of the construction input string exceeds that defined by the element type, Oracle Database reports an error during input parameter binding.

   GaussDB does not verify type modifiers when constructing or binding input parameters. When receiving struct objects and executing SQL statements, a database decides whether to report an error.
3. If the number of array elements exceeds the actual number of columns of the corresponding type, an error is reported during creation.

   When the number of array elements is less than the actual number of columns, the creation of Oracle Database is successful, but an error is reported during parameter input for execution; GaussDB reports an error during creation.

Table 3-42 API reference

Method	Return Type	Throws	GaussDB
getSQLTypeName()	String	SQLException	Supported.
getAttributes()	Object[]	SQLException	Supported.
getAttributes(java.util.Map<String,Class<?>> map)	Object[]	SQLException	Not supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

The differences in the getSQLTypeName API are as follows:

1. For the package type, the struct constructed in the *packageName.typeName* format, the differences in the getSQLTypeName API are as follows:
   - OJDBC11 returns *packageName.typeName*.
   - OJDBC8 generally returns *packageName.typeName*, or returns *"packageName"."typeName"* when *packageName* and *typeName* meet the following condition:

     Either *packageName* or *typeName* violates the rule of starting with a letter and containing only letters, digits, and underscores (_).
   - GaussDB generally returns *schemaName.packageName.typeName*, or returns *"schemaName"."packageName"."typeName"* when *schemaName*, *packageName*, and *typeName* meet the following condition:

     Any of *schemaName*, *packageName*, or *typeName* violates the rule of starting with a letter or underscore (_) and containing only letters, digits, and underscores.
2. For the package type, the differences in the getSQLTypeName API in other scenarios are as follows:
   - OJDBC11 returns *schemaName.packageName.typeName*.
   - OJDBC8 generally returns *schemaName.packageName.typeName*, or returns *"schemaName"."packageName.typeName"* when *schemaName*, *packageName*, and *typeName* meet the following condition:

     Any of *schemaName*, *packageName*, or *typeName* violates the rule of starting with a letter and containing only letters, digits, and underscores (_).
   - GaussDB generally returns *schemaName.packageName.typeName*, or returns *"schemaName"."packageName"."typeName"* when *schemaName*, *packageName*, and *typeName* meet the following condition:

     Any of *schemaName*, *packageName*, or *typeName* violates the rule of starting with a letter or underscore (_) and containing only letters, digits, and underscores.
3. For the non-package type, the differences in the getSQLTypeName API are as follows:
   - OJDBC11 returns *schemaName.typeName*.
   - OJDBC8 generally returns *schemaName.typeName*, or returns *"schemaName"."typeName"* when *schemaName* and *typeName* meet the following condition:

     Either *schemaName* or *typeName* violates the rule of starting with a letter and containing only letters, digits, and underscores (_).
   - GaussDB generally returns *schemaName.typeName*, or returns *"schemaName"."typeName"* when *schemaName* and *typeName* meet the following condition:

     Either *schemaName* or *typeName* violates the rule of starting with a letter or underscore (_) and containing only letters, digits, and underscores.

Table 3-43 Differences in the getAttributes() API

Database Attribute Type	Java Type of the Corresponding Element in the Return Value (Oracle Database OJDBC8)	Java Type of the Corresponding Element in the Return Value (Oracle Database OJDBC11)	Java Type of the Corresponding Element in the Return Value (GaussDB)
CHAR	String	String	String

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Database Attribute Type	Java Type of the Corresponding Element in the Return Value (Oracle Database OJDBC8)	Java Type of the Corresponding Element in the Return Value (Oracle Database OJDBC11)	Java Type of the Corresponding Element in the Return Value (GaussDB)
VARCHAR/ VARCHAR2	String	String	String
NCHAR	String	String	String
NVARCHAR2	String	String	String
NUMBER	BigDecimal	BigDecimal	BigDecimal
NUMERIC	BigDecimal	BigDecimal	BigDecimal
DECIMAL	BigDecimal	BigDecimal	BigDecimal
INTEGER	BigDecimal	BigDecimal	Integer
SMALLINT	BigDecimal	BigDecimal	Short
DOUBLE PRECISION	BigDecimal	BigDecimal	Double
FLOAT	BigDecimal	BigDecimal	Double
REAL	BigDecimal	BigDecimal	Float
BINARY_DOUBLE	Double	Double	Double
BINARY_INTEGER	BigDecimal	Integer	Integer
BOOLEAN	BigDecimal	Integer	Boolean
TIMESTAMP	Timestamp	Timestamp	Timestamp
TIMESTAMP WITH TIME ZONE	TIMESTAMPTZ	TIMESTAMPTZ	Timestamp
BLOB	BLOB	BLOB	PGBlob
CLOB	CLOB	CLOB	PGClob
Set/Array	ARRAY	ARRAY	GaussArray
RECORD	STRUCT	STRUCT	GaussStruct

NOTE

GaussDB currently does not support the types not listed in the aforementioned table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-44 Differences in struct construction APIs

Database Attribute Type	Java Types Supported by Attribute Input Parameters (Oracle Database)	Java Types Supported by Attribute Input Parameters (GaussDB)	Difference
CHAR	Any Java type	Byte, Short, Integer, Long, BigInteger, BigDecimal, Float, Double, Character, Boolean, String, java.sql.Date, java.sql.Time, java.sql.Timestamp, and PGClob	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. When the input parameter is of the String type, Oracle Database adds spaces to the end of the string until it meets the length defined by the type. GaussDB does not add spaces.
VARCHAR/ VARCHAR2	Any Java type	Byte, Short, Integer, Long, BigInteger, BigDecimal, Float, Double, Character, Boolean, String, java.sql.Date, java.sql.Time, java.sql.Timestamp, and PGClob	The different types supported by the input parameter **attribute** can be seen in the table.
NCHAR	Any Java type	Byte, Short, Integer, Long, BigInteger, BigDecimal, Float, Double, Character, Boolean, String, java.sql.Date, java.sql.Time, java.sql.Timestamp, and PGClob	The different types supported by the input parameter **attribute** can be seen in the table.
NVARCHAR2	Any Java type	Byte, Short, Integer, Long, BigInteger, BigDecimal, Float, Double, Character, Boolean, String, java.sql.Date, java.sql.Time, java.sql.Timestamp, and PGClob	The different types supported by the input parameter **attribute** can be seen in the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Database Attribute Type	Java Types Supported by Attribute Input Parameters (Oracle Database)	Java Types Supported by Attribute Input Parameters (GaussDB)	Difference
NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. When the input parameter is of the Float, Double, BigDecimal, or String type and the decimal part is 0, Oracle Database truncates the decimal part, while GaussDB retains it.
NUMERIC	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. When the input parameter is of the Float, Double, BigDecimal, or String type and the decimal part is 0, Oracle Database truncates the decimal part, while GaussDB retains it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Database Attribute Type	Java Types Supported by Attribute Input Parameters (Oracle Database)	Java Types Supported by Attribute Input Parameters (GaussDB)	Difference
DECIMAL	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. When the input parameter is of the Float, Double, BigDecimal, or String type and the decimal part is 0, Oracle Database truncates the decimal part, while GaussDB retains it.
INTEGER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. The target types are inconsistent. For details, see differences in the getAttributes() API. 3. When the input parameter value exceeds the integer range, GaussDB reports an error.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Database Attribute Type	Java Types Supported by Attribute Input Parameters (Oracle Database)	Java Types Supported by Attribute Input Parameters (GaussDB)	Difference
SMALLINT	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. The target types are inconsistent. For details, see differences in the getAttributes() API. 3. When the input parameter exceeds the Short range, GaussDB reports an error.
DOUBLE PRECISION	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. The target types are inconsistent. For details, see differences in the getAttributes() API. 3. Converting a higher-precision type to Double may result in precision loss.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Database Attribute Type	Java Types Supported by Attribute Input Parameters (Oracle Database)	Java Types Supported by Attribute Input Parameters (GaussDB)	Difference
FLOAT	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. The target types are inconsistent. For details, see differences in the getAttributes() API. 3. Converting a higher-precision type to Double may result in precision loss.
REAL	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. The target types are inconsistent. For details, see differences in the getAttributes() API. 3. Converting a higher-precision type to Float may result in precision loss.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Database Attribute Type	Java Types Supported by Attribute Input Parameters (Oracle Database)	Java Types Supported by Attribute Input Parameters (GaussDB)	Difference
BINARY_DOUBLE	byte[], Double, and oracle.sql.BINARY_DOUBLE	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. The target types are inconsistent. For details, see differences in the getAttributes() API. 3. Converting a higher-precision type to Double may result in precision loss.
BINARY_INTEGER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, AtomicInteger, AtomicLong, DoubleAccumulator, DoubleAdder, LongAccumulator, LongAdder, Striped64, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. If the input value exceeds the integer range, GaussDB reports an error, whereas Oracle Database truncates the value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Database Attribute Type	Java Types Supported by Attribute Input Parameters (Oracle Database)	Java Types Supported by Attribute Input Parameters (GaussDB)	Difference
BOOLEAN	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, AtomicInteger, AtomicLong, DoubleAccumulator, DoubleAdder, LongAccumulator, LongAdder, Striped64, String, and oracle.sql.NUMBER	Byte, Short, Integer, Long, Float, Double, Boolean, BigDecimal, BigInteger, and String	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. The target types are inconsistent. For details, see differences in the getAttributes() API. 3. In GaussDB, the target data type is Boolean, which only supports inputs of **1**, **0**, **"true"**, and **"false"**.
TIMESTAMP	byte[], java.sql.Date, String, java.sql.Time, java.sql.Timestamp, oracle.sql.TIMESTAMP, and oracle.sql.DATE	java.util.Date, java.sql.Date, java.sql.Time, java.sql.Timestamp, LocalDateTime, and String	The different types supported by the input parameter **attribute** can be seen in the table.
TIMESTAMP WITH TIME ZONE	java.sql.Date, Calendar, java.util.Date, LocalDate, LocalDateTime, LocalTime, OffsetDateTime, OffsetTime, String, java.sql.Time, java.sql.Timestamp, oracle.sql.TIMESTAMP, oracle.sql.TIMESTAMPTZ, oracle.sql.TIMESTAMPLTZ, and ZonedDateTime	java.util.Date, java.sql.Date, java.sql.Time, java.sql.Timestamp, LocalDateTime, and String	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. The target types are inconsistent. For details, see differences in the getAttributes() API.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Database Attribute Type	Java Types Supported by Attribute Input Parameters (Oracle Database)	Java Types Supported by Attribute Input Parameters (GaussDB)	Difference
BLOB	oracle.sql.BLOB and oracle.jdbc.driver.OracleBlob	PGBlob	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. The target types are inconsistent. For details, see differences in the getAttributes() API.
CLOB	oracle.sql.CLOB and oracle.jdbc.driver.OracleClob	PGClob	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. The target types are inconsistent. For details, see differences in the getAttributes() API.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Database Attribute Type	Java Types Supported by Attribute Input Parameters (Oracle Database)	Java Types Supported by Attribute Input Parameters (GaussDB)	Difference
Set/Array	Array and Object	GaussArray and Object	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. Oracle Database does not report an error when the input parameter **attribute** is of the ARRAY type, even if the actual type required by **attribute** is not ARRAY. GaussDB reports an error when the input parameter **attribute** is of the GaussArray type instead of the actual type required by **attribute**. 3. When the input parameter **attribute** is of the Object type, refer to the differences in array construction APIs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Database Attribute Type	Java Types Supported by Attribute Input Parameters (Oracle Database)	Java Types Supported by Attribute Input Parameters (GaussDB)	Difference
RECORD	Struct and Object[]	GaussStruct and Object[]	1. The different types supported by the input parameter **attribute** can be seen in the table. 2. Oracle Database does not report an error when the input parameter **attribute** is of the STRUCT type, even if the actual type required by **attribute** is not STRUCT. GaussDB reports an error when the input parameter **attribute** is of the GaussStruct type instead of the actual type required by **attribute**. 3. When the input parameter **attribute** is of the Object[] type, refer to the differences in struct construction APIs.

NOTE

1. When a struct is constructed, if the Java type of the input element does not match the target type, an implicit conversion operation is performed. For details on the Java types of input elements supported by various database element types, refer to the preceding table.
2. GaussDB currently does not support the types not listed in the aforementioned table.
3. The constructor needs to provide an attribute array. The preceding table describes the differences of each attribute in the array.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.2.8 Common SQL DDL Clauses

This chapter describes common compatible SQL DDL clauses, including allocate_extent_clause, constraint, deallocate_unused_clause, file_specification, logging_clause, parallel_clause, physical_attributes_clause, size_clause, storage_clause, aggregate function nesting, sequence creation, and sequence altering, as shown in **Table 3-45**.

Table 3-45 Common SQL DDL clauses

Oracle Database	GaussDB	Difference

allocate_extent_clause
Syntax:
`ALLOCATE EXTENT [ ( { SIZE size_clause | DATAFILE 'filename' | INSTANCE integer } ... ) ]`
For example, after the **employees** table is created, change the allocated extent **size** of the table to **10M**.
`SQL> CREATE TABLE employees(EMPLOYEE_ID NUMBER(38), JOB_ID NUMBER(38), SALARY NUMBER(38), LAST_NAME VARCHAR2(16));`
 	
`Table created.`
 	
`SQL> ALTER TABLE employees ALLOCATE EXTENT (SIZE 10M);`
 	
`Table altered.`
	Not supported.	-

constraint
Syntax:
`{ inline_constraint | out_of_line_constraint | inline_ref_constraint | out_of_line_ref_constraint }`
For example, when you create the **staff** table, the **ID** and **NAME** columns specified in the constraint clause cannot be empty.
`SQL> CREATE TABLE staff(ID INT NOT NULL, NAME char(8) NOT NULL, AGE INT, ADDRESS CHAR(50), SALARY REAL);`
 	
`Table created.`
	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
deallocate_unused_clause Syntax: DEALLOCATE UNUSED [KEEP size_clause] For example, after creating the **employees** table and performing some INSERT and DELETE operations, you want to use the deallocate_unused_clause to release the unused space of the **employees** table. SQL> CREATE TABLE employees(EMPLOYEE_ID NUMBER(38), JOB_ID NUMBER(38), SALARY NUMBER(38), LAST_NAME VARCHAR2(16)); Table created. - Perform some INSERT and DELETE operations. SQL> ALTER TABLE employees DEALLOCATE UNUSED; Table altered.	Not supported.	-
file_specification Syntax: {['filename' | 'ASM_filename'] [SIZE size_clause] [REUSE] [autoextend_clause]} | {['filename | ASM_filename' | ('filename | ASM_filename' [, 'filename | ASM_filename']...)] [SIZE size_clause] [BLOCKSIZE size_clause [REUSE]} For example, create a temporary tablespace **tbs_temp_01**. In this clause, create a temporary database file **temp01.dbf** in the tablespace, enable AUTOEXTEND, and allocate the tablespace to the tablespace group **tbs_grp_01**. SQL> CREATE TEMPORARY TABLESPACE tbs_temp_01 TEMPFILE 'temp01.dbf' AUTOEXTEND ON TABLESPACE GROUP tbs_grp_01; Tablespace created.	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference

logging_clause
Syntax:
{ LOGGING | NOLOGGING | FILESYSTEM_LIKE_LOGGING }
	Partially supported, with differences.	• GaussDB does not support the LOGGING and FILESYSTEM_LIKE_LOGGING constraint clauses.
		Example:
		When a table is created in GaussDB with the LOGGING constraint clause, a syntax error is reported.
		gaussdb=# CREATE LOGGING TABLE my_tab(id int, name char(16));
		ERROR: syntax error at or near "LOGGING"
		LINE 1: CREATE LOGGING TABLE my_tab(id int, name char(16));
		^
		When a table is created in GaussDB with the FILESYSTEM_LIKE_LOGGING constraint clause, a syntax error is reported.
		gaussdb=# CREATE FILESYSTEM_LIKE_LOGGING TABLE my_tab(id int, name char(16));
		ERROR: syntax error at or near "FILESYSTEM_LIKE_LOGGING"
		LINE 1: CREATE FILESYSTEM_LIKE_LOGGING TABLE my_tab(id int, name cha...
		^
		• GaussDB supports only table-level UNLOGGED constraints and does not support column-level UNLOGGED constraints.
		For example, when a table is created in GaussDB with the column-level UNLOGGED constraint clause, a syntax error is reported.
		gaussdb=# CREATE UNLOGGED TABLE my_tab(id int UNLOGGED, name char(16));
		ERROR: syntax error at or near "UNLOGGED"
		LINE 1: CREATE UNLOGGED TABLE my_tab(id int UNLOGGED, name char(16))...
		^
		• GaussDB uses logging clauses only in the CREATE TABLE, CREATE TABLE AS, and SELECT INTO statements.
		For example, when a TABLESPACE statement with the UNLOGGED constraint clause is created in GaussDB, a syntax error is reported.
		gaussdb=# CREATE UNLOGGED TABLESPACE tbs1 RELATIVE LOCATION 'tablespace1/tablespace_1';
		ERROR: syntax error at or near "TABLESPACE"
		LINE 1: CREATE UNLOGGED TABLESPACE tbs1 RELATIVE LOCATION 'tablespac...
		^

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
parallel_clause Syntax: { NOPARALLEL | PARALLEL [integer] } For example, if you create table **t1** and specify **PARALLEL 4** in the parallel_clause, a maximum of four parallel processes can be used to query and update table **t1**. SQL> CREATE TABLE t1 (id NUMBER, name VARCHAR2(50)) PARALLEL 4; Table created.	Not supported.	-

physical_attributes_clause
Syntax:
[{ PCTFREE integer | PCTUSED integer | INITRANS integer | storage_clause }...]
	Partially supported, with differences.	• GaussDB does not support PCTUSED. For example, if you run an SQL statement to create the **tbl1_ind** index in the **tbl1** table and set the space usage **PCTUSED** of the index to **20%** in the physical_attributes_clause of the statement, an error is reported when the SQL statement is executed in GaussDB.
		gaussdb=# CREATE INDEX tbl1_ind ON tbl1 (name) PCTUSED 20;
		ERROR: syntax error at or near "PCTUSED"
		LINE 1: CREATE INDEX tbl1_ind ON tbl1 (name) PCTUSED 20;
		^
		• GaussDB uses physical_attributes_clause only in the CREATE TABLE and CREATE INDEX statements.
		For example, if you run an SQL statement to obtain data from the **tbl1** table, create the materialized view **tbl1_mv**, and set the number of initial transactions of the view to **30** in the physical_attributes_clause, an error is reported when GaussDB executes the statement.
		gaussdb=# CREATE MATERIALIZED VIEW tbl1_mv INITRANS 30 as select * from tbl1;
		ERROR: syntax error at or near "INITRANS"
		LINE 1: CREATE MATERIALIZED VIEW tbl1_mv INITRANS 30 as select * fro...
		^

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
size_clause Syntax: integer [K | M | G | T | P | E] For example, create a temporary tablespace **tbs_temp_01** and create a temporary database file **temp01.dbf** in the tablespace. In this clause, specify the initial size as 5 MB, enable AUTOEXTEND, and allocate the tablespace to tablespace group **tbs_grp_01**. SQL> CREATE TEMPORARY TABLESPACE tbs_temp_01 TEMPFILE 'temp01.dbf' SIZE 5M AUTOEXTEND ON TABLESPACE GROUP tbs_grp_01; Tablespace created.	Not supported.	-

Oracle Database	GaussDB	Difference
storage_clause Syntax: STORAGE ({ INITIAL size_clause | NEXT size_clause | MINEXTENTS integer | MAXEXTENTS { integer | UNLIMITED } | maxsize_clause | PCTINCREASE integer | FREELISTS integer | FREELIST GROUPS integer | OPTIMAL [size_clause | NULL] | BUFFER_POOL { KEEP | RECYCLE | DEFAULT } | FLASH_CACHE { KEEP | NONE | DEFAULT } | (CELL_FLASH_CACHE (KEEP | NONE | DEFAULT)) | ENCRYPT } ...)	Partially supported, with differences.	• In Oracle Database, storage parameters are specified by the STORAGE clause. In GaussDB, storage parameters are specified by the WITH clause. Example: To create the **my_tab1** table in Oracle Database, set the initial size of the table to **10M** in the storage_clause, and add 5 MB each time when more space is required, run the following SQL statement: SQL> CREATE TABLE my_tab1 (id NUMBER(10) PRIMARY KEY, name VARCHAR2(50)) STORAGE (INITIAL 10M NEXT 5M); Table created. To create the **my_tab2** table in GaussDB and set the storage engine type to **ustore** in the storage_clause, run the following SQL statement: gaussdb=# CREATE TABLE my_tab2 (id NUMBER(10) PRIMARY KEY, name VARCHAR2(50)) with (storage_type=ustore); NOTICE: CREATE TABLE / PRIMARY KEY will create implicit index "my_tab2_pkey" for table "my_tab2" CREATE TABLE • Optional storage parameters in GaussDB are greatly different from those in Oracle Database. For details, see the parameter description in **CREATE TABLE** in GaussDB. The "WITH ({storage_parameter = value} [,...])" section describes the storage parameters supported by the CREATE TABLE statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
Aggregate function nesting For example, create the **revenue** table generated by nesting the aggregate functions MIN() and SUM() in the **sales_amount** column of the **sales** table. SQL> CREATE TABLE sales(ID INT, SALES_AMOUNT INT); Table created. SQL> INSERT INTO sales VALUES(1, 100); 1 row created. SQL> INSERT INTO sales VALUES (3, 200); 1 row created. SQL> CREATE TABLE revenue as SELECT SUM(MIN(sales_amount)) as total from sales group by sales_amount; Table created.	Supported.	-
Dropping a system schema Syntax: DROP USER schema_name CASCADE; For example, drop the SYS schema as the **SYS** user. SQL> DROP USER SYS; DROP USER SYS * ERROR at line 1: ORA-28050: specified user or role cannot be dropped	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference

Dropping a PRIMARY KEY constraint from a table
Syntax:
DROP PRIMARY KEY [CASCADE] [(KEEP | DROP) INDEX] [ONLINE]
	Partially supported, with differences.	• GaussDB supports only the DROP PRIMARY KEY syntax and does not support other options.
		• When this operation is performed, Oracle Database automatically drops the NOT NULL constraint added to the primary key column when the primary key is created, while GaussDB does not drop the NOT NULL constraint but retains it.
		-- GaussDB:
		gaussdb=# CREATE TABLE t1 (id int PRIMARY KEY);
		NOTICE: CREATE TABLE / PRIMARY KEY will create implicit index "t1_pkey" for table "t1"
		CREATE TABLE
		gaussdb=# ALTER TABLE t1 DROP PRIMARY KEY;
		ALTER TABLE
		gaussdb=# INSERT INTO t1 VALUES(null);
		ERROR: The null value in column "id" violates the not-null constraint.
		DETAIL: Failing row contains (null).
		gaussdb=# DROP TABLE t1;
		
		-- Oracle:
		SQL> CREATE TABLE t1 (id int PRIMARY KEY);
		SQL> ALTER TABLE t1 DROP PRIMARY KEY;
		SQL> INSERT INTO t1 VALUES(null);
		-- The operation can be successfully performed, that is, the NOT NULL constraint of the primary key column is also dropped.
		SQL> DROP TABLE t1;

Creating a sequence Syntax: CREATE SEQUENCE [IF NOT EXISTS] [schema.] sequence [SHARING = { METADATA | DATA | NONE }] [{ INCREMENT BY | START WITH } integer | { MAXVALUE integer | NOMAXVALUE } | { MINVALUE integer | NOMINVALUE } | { CYCLE | NOCYCLE } | { CACHE integer | NOCACHE } | { ORDER | NOORDER } | { KEEP | NOKEEP } | { SCALE {EXTEND | NOEXTEND} | NOSCALE } | { SHARD {EXTEND | NOEXTEND} | NOSHARD } | { SESSION | GLOBAL }]...	Partially supported, with differences.	• GaussDB does not support the **SHARING**, **CACHE**, **NOCACHE**, **ORDER**, **NOORDER**, **KEEP**, **NOKEEP**, **SCALE**, **NOSCALE**, **SHARD**, **NOSHARD**, **SESSION**, and **GLOBAL** options. • GaussDB does not check whether the absolute value of the increment is less than the difference between the maximum value and the minimum value. -- GaussDB: gaussdb=# CREATE SEQUENCE seq_test MINVALUE 1 MAXVALUE 10 INCREMENT BY 10; CREATE SEQUENCE -- Oracle: SQL> CREATE SEQUENCE seq_test MINVALUE 1 MAXVALUE 10 INCREMENT BY 10; CREATE SEQUENCE seq_test MINVALUE 1 MAXVALUE 10 INCREMENT BY 10 * ERROR at line 1: ORA-04005: INCREMENT must be less than MAXVALUE minus MINVALUE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference

Altering a sequence
Syntax:
ALTER SEQUENCE [schema.] sequence { { (INCREMENT BY) | (START WITH) } integer | { MAXVALUE integer | NOMAXVALUE } | { MINVALUE integer | NOMINVALUE } | RESTART | { CYCLE | NOCYCLE } | { CACHE integer | NOCACHE } | { ORDER | NOORDER } | { KEEP | NOKEEP } | { SCALE {EXTEND | NOEXTEND} | NOSCALE } | { SHARD {EXTEND | NOEXTEND} | NOSHARD } | { SESSION | GLOBAL } } ...
	Partially supported, with differences.	• GaussDB does not support the **INCREMENT BY**, **MINVALUE**, **NOMINVALUE**, **RESTART**, **CYCLE**, **NOCYLE**, **CACHE**, **NOCACHE**, **ORDER**, **NOORDER**, **KEEP**, **NOKEEP**, **SCALE**, **NOSCALE**, **SHARD**, **NOSHARD**, **SESSION**, and **GLOBAL** options.
		• When the maximum value is changed in GaussDB, the system does not check whether the absolute value of the increment is less than the difference between the maximum value and the minimum value.
		-- GaussDB:
		gaussdb=# CREATE SEQUENCE seq_test MINVALUE 1 MAXVALUE 100 INCREMENT BY 10;
		CREATE SEQUENCE
		gaussdb=# ALTER SEQUENCE seq_test MAXVALUE 10;
		ALTER SEQUENCE
		
		-- Oracle:
		SQL> CREATE SEQUENCE seq_test MINVALUE 1 MAXVALUE 100 INCREMENT BY 10;
		
		Sequence created.
		
		SQL> ALTER SEQUENCE seq_test MAXVALUE 10;
		ALTER SEQUENCE seq_test MAXVALUE 10
		*
		ERROR at line 1:
		ORA-04005: INCREMENT must be less than MAXVALUE minus MINVALUE
		• When the maximum value is changed in GaussDB, the system checks whether the start value of the sequence is within the range of the maximum value and minimum value.
		-- GaussDB:
		gaussdb=# CREATE SEQUENCE seq_test MINVALUE 1 MAXVALUE 10 START WITH 10 INCREMENT BY -1;
		CREATE SEQUENCE
		gaussdb=# SELECT seq_test.nextval FROM DUAL;
		nextval
		--------- 	
		10
		(1 row)
		
		gaussdb=# SELECT seq_test.nextval FROM DUAL;
		nextval
		--------- 	
		9
		(1 row)
		
		gaussdb=# SELECT seq_test.nextval FROM DUAL;

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
		nextval --------- 8 (1 row) gaussdb=# ALTER SEQUENCE seq_test MAXVALUE 9; ERROR: START value (10) cannot be greater than MAXVALUE (9). -- Oracle: SQL> CREATE SEQUENCE seq_test MINVALUE 1 MAXVALUE 10 START WITH 10 INCREMENT BY -1; Sequence created. SQL> SELECT seq_test.nextval FROM DUAL; NEXTVAL ---------- 10 SQL> SELECT seq_test.nextval FROM DUAL; NEXTVAL ---------- 9 SQL> SELECT seq_test.nextval FROM DUAL; NEXTVAL ---------- 8 SQL> ALTER SEQUENCE seq_test MAXVALUE 9; Sequence altered. • When nextval is executed for the first time in GaussDB, the internal position is the start value. When the maximum value is changed, the system checks whether the start value of the sequence is within the range of the maximum value and minimum value. -- GaussDB: gaussdb=# CREATE SEQUENCE seq_test MINVALUE 1 MAXVALUE 10 START WITH 6 INCREMENT BY -5; CREATE SEQUENCE gaussdb=# ALTER SEQUENCE seq_test MAXVALUE 9; ALTER SEQUENCE -- Oracle: SQL> CREATE SEQUENCE seq_test MINVALUE 1 MAXVALUE 10 START WITH 6 INCREMENT BY -5; Sequence created. SQL> ALTER SEQUENCE seq_test MAXVALUE 9;

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
		ALTER SEQUENCE seq_test MAXVALUE 9 * ERROR at line 1: ORA-04009: MAXVALUE cannot be made to be less than the current value

3.2.9 SQL Queries and Subqueries

GaussDB is compatible with SQL queries and subqueries except hierarchical queries.

Table 3-46 SQL queries and subqueries

Oracle Database	GaussDB	Difference
Creating simple queries	Supported.	-
Hierarchical queries	Not supported.	-
UNION [ALL], INTERSECT, MINUS	Supported.	-
Sorting query results	Supported, with differences.	If the GaussDB query does not contain groups and the target column contains both an aggregate function and a set returning function, the sorting of the set returning function column is not ignored. If the query column involves type conversion and the sorting key is the original column name, GaussDB sorts the data based on the converted type because the default alias of the conversion result is the original column name, while Oracle Database sorts the data based on the original column type because the default alias of the conversion result is the conversion expression. The sorting semantics are different, which may cause the result set sequence to be inconsistent.
Joins	Supported, with differences.	GaussDB supports only the same join types as Oracle Database, such as left/right, self, natural, and full outer join. Join optimization methods such as IN-MEMORY JOIN GROUPS are not supported.
Using subqueries	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
Unnesting of nested subqueries	Supported, with differences.	HASH_AJ or MERGE_AJ cannot be explicitly specified in GaussDB.
Distributed queries	Supported, with differences.	GaussDB requires explicit database link query.
Aggregate function nesting	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference

Hierarchical Queries	Supported, but with differences.	• When the PRIOR keyword is used to modify a column in a non-CONNECT BY clause in GaussDB, the returned column name is different from that in Oracle Database. For example, for column **a**, GaussDB returns **__prior_a__**, and Oracle Database returns **PRIORA**.
		• When an expression with the **PRIOR** column is used in the GROUP BY clause of SELECT, the behavior of GaussDB is different from that of Oracle Database. GaussDB supports the use of an expression with the **PRIOR** column in the GROUP BY clause, while Oracle Database reports an error.
		-- GaussDB:
		gaussdb=# CREATE TABLE test (c1 INT, c2 INT, c3 INT);
		NOTICE: The 'DISTRIBUTE BY' clause is not specified. Using 'c1' as the distribution column by default.
		HINT: Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
		CREATE TABLE
		gaussdb=# INSERT INTO test VALUES(1,2,3);
		INSERT 0 1
		gaussdb=# INSERT INTO test VALUES(2,3,4);
		INSERT 0 1
		gaussdb=# SELECT PRIOR c1 + 1 AS result FROM test CONNECT BY PRIOR c2 = c1 GROUP BY PRIOR c1 + 1;
		result
		-------- 	
		
		2
		(2 rows)
		
		gaussdb=# DROP TABLE test;
		DROP TABLE
		
		-- Oracle:
		SQL> CREATE TABLE test (c1 INT, c2 INT, c3 INT);
		SQL> INSERT INTO test VALUES(1,2,3);
		SQL> INSERT INTO test VALUES(2,3,4);
		-- This operation will cause the error "not a GROUP BY expression."
		SQL> SELECT PRIOR c1 + 1 AS result FROM test CONNECT BY PRIOR c2 = c1 GROUP BY PRIOR c1 + 1;
		SQL> DROP TABLE test;
		• In GaussDB, ORDER SIBLINGS BY and ORDER BY can be used together. In this case, the sorting result of the outer ORDER BY clause is used. Oracle Database does not support this usage.

Currently, xmltable_clause in GaussDB are different from that in the Oracle Database and PostgreSQL databases, as described in **Table 3-47**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-47 Comparison between GaussDB, Oracle Database, and PG

GaussDB	PostgreSQL Database	Oracle Database
XPath 1.0 expressions are used at row_expression and **PATH** in columns_clause.	Same as GaussDB.	XQuery 1.0 expressions are used at row_expression and **PATH** in columns_clause.
The default namespace function is not supported.	Same as GaussDB.	The default namespace function is supported.
A single data record can be passed through passing_clause, but the alias cannot be used.	Same as that in GaussDB. (Note that the input data is of the XML type.)	Multiple data records can be passed through the corresponding clause and aliases can be used.
The passing_clause can be omitted.	The passing_clause cannot be omitted.	The corresponding clause can be omitted.
The RETURNING SEQUENCE BY REF clause cannot be used after passing_clause.	Same as GaussDB.	The RETURNING SEQUENCE BY REF clause can be used after the corresponding clause.
In columns_clause, you cannot use (SEQUENCE) BY REF to modify the returned data of the XMLType type.	Same as GaussDB.	You can use (SEQUENCE) BY REF in the corresponding clause to modify the returned data of the XMLType type.
If the **PATH** part in columns_clause is omitted and the column name is not enclosed using double quotation marks (""), the node can be correctly found for subsequent operations when the node name in the data passed to XMLTABLE is in lowercase.	Same as GaussDB.	If the corresponding part is omitted and the column name is not enclosed using double quotation marks (""), the node can be correctly found for subsequent operations when the node name in the data passed to XMLTABLE is in uppercase.
The declaration of the column type in columns_clause cannot be omitted.	Same as GaussDB.	The declaration of the column type in the corresponding clause can be omitted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GaussDB	PostgreSQL Database	Oracle Database
In ORA-compatible mode, the columns_clause can be omitted. In PG-compatible mode, the columns_clause cannot be omitted.	The corresponding clause cannot be omitted.	The corresponding clause can be omitted.
In ORA-compatible mode, if the returned data length exceeds the typmod of the type, the data is truncated. In PG-compatible mode, if the returned data length exceeds the typmod of the type, an error is reported.	An error is reported when the length of the returned data exceeds the typmod of the type.	If the length of the returned data exceeds the typmod of the type, the data is truncated.

Example:

```
gaussdb=# CREATE DATABASE test DBCOMPATIBILITY = 'ORA';
gaussdb=# \c test
test=# SELECT * FROM XMLTABLE(
test(# XMLNAMESPACES('nspace1' AS "ns1", 'nspace2' AS "ns2"), -- Declare two XML namespaces 'nspace1' and 'nspace2' and the corresponding aliases "ns1" and "ns2".
test(# '/ns1:root/*:child' -- Select the root node whose namespace is 'nspace1' from the data passed through row_expression, select all child nodes under the root node, and ignore the child namespaces. ns1 is the alias of 'nspace1'.
test(# PASSING xmltype(
test(# '<root xmlns="nspace1">
test'#     <child>
test'#         <name>peter</name>
test'#         <age>11</age>
test'#     </child>
test'#     <child xmlns="nspace1">
test'#         <name>qiqi</name>
test'#         <age>12</age>
test'#     </child>
test'#     <child xmlns="nspace2">
test'#         <name>hacker</name>
test'#         <age>15</age>
test'#     </child>
test'# </root>')
test(# COLUMNS
test(# columns FOR ORDINALITY, -- Row number column
test(# name varchar(10) path 'ns1:name', -- Select the name node whose namespace is 'nspace1' from each child node obtained by row_expression, convert the value in the node to varchar(10), and return the value. ns1 is the alias of nspace1.
test(# age int); -- Select the age node from each child node obtained by row_expression, convert the value in the node to int, and return the value. The first child node does not explicitly specify the namespace. Therefore, the namespace of the parent node root is followed. Therefore, no value in this column is returned.
 column | name  | age
--------+-------+-----
      1 | peter |
      2 | qiqi  |
      3 |       |
(3 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Switch to the default database and drop the test database. Change the database name as required.
gaussdb=# \c postgres
gaussdb=# DROP DATABASE test;
```

3.2.10 PL/SQL Language

GaussDB is compatible with PL/SQL operators, expressions, control statements, collections, and records, but does not support predefined PL/SQL constants, types, and subtypes.

3.2.10.1 Basic PL/SQL Syntax

Table 3-48 PL/SQL operators

Oracle Database	GaussDB
+	Supported.
:=	Supported.
=>	Supported.
%	Supported.
'	Supported.
.	Supported.
||	Supported.
/	Supported.
**	Not supported.
(	Supported.
)	Supported.
:	Supported.
,	Supported.
<<	Supported.
>>	Supported.
/*	Supported.
*/	Supported.
*	Supported.
"	Supported.
..	Supported.
=	Supported.
<>	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
!=	Supported.
~=	Supported.
^=	Supported.
<	Supported.
>	Supported.
<=	Supported.
>=	Supported.
@	Supported.
--	Supported.
;	Supported.
-	Supported.

Table 3-49 Logical operators

Oracle Database	GaussDB
NOT	Supported.
AND	Supported.
OR	Supported.

Table 3-50 Comparison expressions

Oracle Database	GaussDB
IS [NOT] NULL	Supported.
LIKE	Supported.
BETWEEN	Supported.
IN	Supported.

Table 3-51 CASE expressions

Oracle Database	GaussDB
simple CASE	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
searched CASE	Supported.

Table 3-52 Parameters related to variable declaration

Oracle Database	GaussDB	Difference
%TYPE	Supported, with differences.	• GaussDB does not support *record%type*. • GaussDB does not support **pkg.***record%type* and **schema.pkg.***record%type* as the input and output parameter types. • In GaussDB, *Table/View*.**column.***column%type* or **schema.***Table/View*.**column.***column%type* cannot be nested with one or more layers as variable types or input/output parameter type. • In GaussDB, *record*.**column.***column%type* and **pkg.***record*.**column.***column%TYPE* cannot be nested with a column type of records at one or more layers as the variable type or input/output parameter type.
%ROWTYPE	Supported, with differences.	• When GaussDB has multiple CNs, the **%ROWTYPE** and **%TYPE** attributes of the temporary table cannot be declared in a stored procedure. The temporary table is valid only in the current session. During compilation, other CNs cannot view the temporary table of the current CN. Therefore, if there are multiple CNs, the system displays a message indicating that the temporary table does not exist. • GaussDB does not support *view%rowtype* and **schema.***view%rowtype* as the input and output parameter types. • GaussDB does not support the **package.***cursor%rowtype* as the input and output parameter types.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.2.10.2 Data Type Compatibility

Table 3-53 Other PL/SQL data types

Oracle Database	GaussDB	Difference
CHARACTER	Supported, with differences.	• GaussDB: The length ranges from 1 to 10485760 bytes. • Oracle Database: The length ranges from 1 to 32767 bytes.
VARCHAR	Supported, with differences.	• GaussDB: The length ranges from 1 to 10485760 bytes. • Oracle Database: The length ranges from 1 to 32767 bytes.
STRING	Not supported.	-
PLS_INTEGER	Not supported.	In GaussDB, you can use the INT type instead.
BINARY_INTEGER	Supported.	-

3.2.10.3 Control Statements

Table 3-54 Conditional statements

Oracle Database	GaussDB
IF THEN	Supported.
IF THEN ELSE	Supported.
IF THEN ELSIF	Supported.
simple CASE: CASE selector WHEN selector_value_1 THEN statements_1 WHEN selector_value_2 THEN statements_2 ... WHEN selector_value_n THEN statements_n [ELSE else_statements END CASE;]	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
searched CASE: CASE WHEN condition_1 THEN statements_1 WHEN condition_2 THEN statements_2 ... WHEN condition_n THEN statements_n [ELSE else_statements END CASE;]	Supported.

Table 3-55 LOOP statements

Oracle Database	GaussDB
[label] LOOP statements END LOOP [label];	Supported.
EXIT;	Supported.
EXIT WHEN;	Supported.
CONTINUE;	Supported.
CONTINUE WHEN;	Supported.

Table 3-56 FOR LOOP statements

Oracle Database	GaussDB	Difference
[label] FOR index IN [REVERSE] lower_bound..upper_bound LOOP statements END LOOP [label];	Supported, with differences.	When the keyword REVERSE is used in GaussDB, the lower bound must be greater than or equal to the upper bound; otherwise, the loop body is not executed.
EXIT WHEN;	Supported.	-
CONTINUE WHEN;	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-57 WHILE LOOP statement

Oracle Database	GaussDB
[label] WHILE condition LOOP statements END LOOP [label];	Supported.

Table 3-58 GOTO statement

Oracle Database	GaussDB
GOTO	Supported.

Table 3-59 NULL statement

Oracle Database	GaussDB
NULL	Supported.

3.2.10.4 Collections and Records

Table 3-60 Types

Oracle Database	GaussDB
Associative array (or index-by table)	Supported.
VARRAY (variable-size array)	Supported.
Nested table	Supported.
record	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-61 Syntax

Oracle Database	GaussDB	Difference
Associative array (or index-by table) syntax: TABLE OF datatype [NOT NULL] INDEX BY { PLS_INTEGER | BINARY_INTEGER | VARCHAR2 (v_size) | data_type }	Supported, but with differences.	• GaussDB does not support the PLS_INTEGER type. In GaussDB, the value of **data_type** can be a base data type or a record type, collection type, or array type defined in a stored procedure. The ref cursor type is not supported. • In GaussDB, NOT NULL is syntactically supported but functionally not effective. That is, the system does not check whether an element is NULL. • For details, see **Collections**.
VARRAY (variable-size array) syntax: { VARRAY | [VARYING] ARRAY } (size_limit) OF datatype [NOT NULL]	Supported, but with differences.	• GaussDB does not support the NOT NULL syntax. • In GaussDB, **datatype** cannot be set to **varray** (varray cannot be nested). • To make the size_limit function take effect, enable the **varray_compat** parameter in the GUC parameter **behavior_compat_options**. • For details, see **Arrays**.
Nested table syntax: TABLE OF datatype [NOT NULL]	Supported, but with differences.	• In GaussDB, NOT NULL is syntactically supported but functionally not effective. • For details, see **Collections**.
record syntax: TYPE record_type IS RECORD (field_definition [, field_definition]...) ;	Supported.	• Record columns can be defined as NOT NULL or a default value can be specified. If the record type is nested in other types, the default value and NOT NULL of the record type do not take effect. If a record variable is created using the package.record_type access type, the default value and NOT NULL of the record variable do not take effect. • For details, see **Records**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-62 Constructor

Oracle Database	GaussDB
collection_type ([value [, value]...])	Supported.

Table 3-63 Variable assignment

Oracle Database	GaussDB	Difference
Associative array (or index-by table)	Supported.	-
VARRAY (variable-size array)	Supported, but with differences.	• Values of different VARRAY data in GaussDB can be assigned to each other, depending on whether elements in the data can be implicitly converted to each other. • For details, see **Arrays**.
Nested table	Supported.	-
record	Supported, but with differences.	• Values of different record data in GaussDB can be assigned to each other, depending on whether columns can be implicitly converted. • For details, see **Records**.

Table 3-64 Collection operators

Oracle Database	GaussDB	Difference
=	Supported, but with differences.	• Oracle Database: The sequence of collection members is ignored during comparison. • GaussDB: The comparison is performed strictly based on the sequence of collection members.
<>	Supported, but with differences.	• Oracle Database: The sequence of collection members is ignored during comparison. • GaussDB: The comparison is performed strictly based on the sequence of collection members.
IS [NOT] NULL	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
^=	Supported, but with differences.	• Oracle Database: The sequence of collection members is ignored during comparison. • GaussDB: The comparison is performed strictly based on the sequence of collection members.
~=	Not supported.	-
IS [NOT] A SET	Not supported.	-
IS [NOT] EMPTY	Not supported.	-
expr [NOT] MEMBER [OF] nested_table	Not supported.	-
nested_table1 [NOT] SUBMULTISET [OF] nested_table2	Not supported.	-
[NOT] IN	Supported.	• Oracle Database: The sequence of collection members is ignored during comparison. • GaussDB: The comparison is performed strictly based on the sequence of collection members.

Table 3-65 MULTISET functions

Oracle Database	GaussDB
MULTISET UNION [ALL | DISTINCT]	Supported.
MULTISET EXCEPT [ALL | DISTINCT]	Supported.
MULTISET INTERSECT [ALL | DISTINCT]	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-66 Collection type functions

Oracle Database	GaussDB	Difference
exists(idx)	Supported.	-
extend[(count[, idx])]	Supported, but with differences.	GaussDB supports only nested tables.
delete[(idx1[, idx2])]	Supported.	-
trim[(n)]	Supported, but with differences.	GaussDB supports only nested tables.
count	Supported.	-
first	Supported.	-
last	Supported.	-
prior(idx)	Supported.	-
next(idx)	Supported.	-
limit	Supported, but with differences.	GaussDB supports only nested tables.

Table 3-67 Record variable operations

Oracle Database	GaussDB
Constructors	Supported.
%ROWTYPE to declare a variable	Supported.
Defining constants	Not supported.

Table 3-68 Collection-related functions

Oracle Database	GaussDB	Difference
unnest_table(anynesttable)	Supported.	-
unnest_table(anyindexbytable)	Supported.	-
table(anyarray)	Not supported.	GaussDB uses the unnest (anyarray) function for equivalent rewriting.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.2.10.5 Static SQL Statements

Table 3-69 Static query SQL statements

Oracle Database	GaussDB	Difference
SELECT	Supported, but with differences.	GaussDB and Oracle Database are different in some scenarios. In GaussDB, FOR SHARE adds a shared lock to the retrieved rows. The shared locks of different transactions do not block each other. If data is locked by FOR SHARE in one transaction and SELECT FOR SHARE SKIP LOCKED is used in another transaction, SKIP LOCKED does not skip the lock.

Table 3-70 Static DML SQL statements

Oracle Database	GaussDB	Difference
INSERT	Supported, but with differences.	• Oracle Database allows the number of columns in the target table to be greater than the number of columns in the subquery result. However, you must explicitly specify the names of columns to be inserted to ensure alignment with the number of columns. • GaussDB eliminates the need to explicitly specify the names of columns to be inserted. In this case, the value of the first column in the subquery result is inserted into the first column of the target table, and so on. If the target table contains extra columns, they will receive either **NULL** (if permitted) or the default value (if any).
UPDATE	Supported.	-
DELETE	Supported.	-
MERGE	Supported.	-
LOCK TABLE	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
INSERT ALL	Supported, but with differences.	• Oracle Database does not support alias setting for the tables of into_clause, but GaussDB supports. • When into_clause specifies the sequence: – Oracle Database: If nextval is referenced for the first time, the next number of the current value is generated. Otherwise, the same number will always be returned. – GaussDB: The number generated by referencing nextval can increment automatically. • The plan_hint statement can take effect in Oracle Database but does not take effect in GaussDB. • Oracle Database allows the number of columns in the target table to be greater than the number of columns in the subquery result. However, you must explicitly specify the names of columns to be inserted to ensure alignment with the number of columns. GaussDB eliminates the need to explicitly specify the names of columns to be inserted. In this case, the value of the first column in the subquery result is inserted into the first column of the target table, and so on. If the target table contains extra columns, they will receive either **NULL** (if permitted) or the default value (if any).

Table 3-71 Static TCL SQL statements

Oracle Database	GaussDB	Difference
COMMIT	Supported.	-
ROLLBACK	Supported.	-
SAVEPOINT	Supported.	-
SET TRANSACTION	Supported, but with differences.	GaussDB does not support the NAME string and USE ROLLBACK SEGMENT rollback_segment syntax.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-72 Pseudocolumns

Oracle Database	GaussDB	Difference
CURRVAL and NEXTVAL	Supported.	-
LEVEL	Not supported.	-
OBJECT_VALUE	Not supported.	-
ROWID	Supported.	-
ROWNUM	Supported, but with differences.	It is not recommended that the ROWNUM condition be used in the JOIN ON clause. In GaussDB, when the ROWNUM condition is used in the JOIN ON clause, the behavior in the LEFT JOIN, RIGHT JOIN, FULL JOIN, and MERGE INTO scenarios is different from that in other databases, causing risks in service migration.

Table 3-73 Implicit cursor attributes

Oracle Database	GaussDB	Difference
SQL%FOUND	Supported, but with differences.	GaussDB does not update the implicit cursor result after COMMIT or ROLLBACK. Oracle Database updates the implicit cursor result after COMMIT or ROLLBACK.
SQL %NOTFOUND	Supported, but with differences.	
SQL %ROWCOUNT	Supported, but with differences.	
SQL%ISOPEN	Supported, but with differences.	
SQL %BULK_ROWCOUNT	Not supported.	
SQL %BULK_EXCEPTIONS	Not supported.	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-74 Explicit cursor syntax and keywords

Oracle Database	GaussDB	Difference
CURSOR cursor_name [parameter_list] RETURN return_type;	Supported.	-
CURSOR cursor_name [parameter_list] [RETURN return_type] IS select_statement;	Supported.	-
OPEN	Supported.	-
CLOSE	Supported, but with differences.	GaussDB is automatically closed in the exception, but Oracle Database is not automatically closed in the exception.
FETCH	Supported.	-
CURRENT OF CURSOR	Supported.	-

Table 3-75 Explicit cursor attributes

Oracle Database	GaussDB
SQL%FOUND	Supported.
SQL%NOTFOUND	Supported.
SQL%ROWCOUNT	Supported.
SQL%ISOPEN	Supported.

Table 3-76 Cursor loop

Oracle Database	GaussDB
FOR LOOP	Supported, but with differences. In the FORALL+BULK COLLECT INTO scenario, the *INTO* variable returns only the execution result of a single DML statement in GaussDB, while in Oracle Database, it returns the accumulated execution results of DML statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-77 Scenarios supported by autonomous transactions

Oracle Database	GaussDB
Stored procedures	Supported.
Anonymous blocks	Supported.
Functions	Supported.
Packages	Supported.

3.2.10.6 Dynamic SQL Statements

Table 3-78 Dynamic SQL statement execution modes

Oracle Database	GaussDB	Difference
EXECUTE IMMEDIATE	Supported, but with differences.	• GaussDB uses the **dynamic_sql_compat** parameter to determine whether variables with the same name read the same parameter and checks whether the input and output parameter types of the bound parameters are the same as those of the statement parameters when a stored procedure is called. • GaussDB does not support the scenario where some bound parameters in an anonymous block are called, for example, dynamic statements are nested in an anonymous block and parameters are bound using expressions. For details, see **Dynamically Calling Anonymous Blocks**. • GaussDB does not support RETURNING or RETURN INTO.
OPEN FOR, FETCH, CLOSE	Supported.	GaussDB uses the **dynamic_sql_compat** parameter to determine whether a variable reads parameters with the same name. When a stored procedure is called, GaussDB checks whether the input and output parameter types are consistent with the statement parameter types.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.2.10.7 Triggers

Table 3-79 Trigger types

Oracle Database	GaussDB	Difference
DML TRIGGER	Supported, but with differences.	GaussDB: Compound DML triggers are not supported.
SYSTEM TRIGGER	Not supported.	-

Table 3-80 CREATE triggers

Oracle Database	GaussDB	Difference
CREATE syntaxes: CREATE [OR REPLACE] [EDITIONABLE | NONEDITIONABLE] TRIGGER plsql_trigger_source	Supported, but with differences.	GaussDB does not support OR REPLACE or EDITIONABLE | NONEDITIONABLE, but supports some behaviors of plsql_trigger_source.
plsql_trigger_source ::= syntax: [schema.] trigger_name [sharing_clause] [default_collation_clause] { simple_dml_trigger | instead_of_dml_trigger | compound_dml_trigger | system_trigger }	Supported, but with differences.	GaussDB: The schema, sharing_clause, and default_collation_clause are not supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
simple_dml_trigger ::= syntax: { BEFORE | AFTER } dml_event_clause [referencing_clause] [FOR EACH ROW] [trigger_edition_clause] [trigger_ordering_clause] [ENABLE | DISABLE] [WHEN (condition)] trigger_body	Supported, but with differences.	GaussDB does not support referencing_clause (instead, from referencing_table is used), trigger_edition_clause, trigger_ordering_clause, and ENABLE | DISABLE. It supports some behaviors of trigger_body. No error is reported in GaussDB when a statement-level BEFORE/ AFTER TRIGGER is created in a view without INSTEAD OF TRIGGER. An error is reported when DML is executed.
dml_event_clause ::= syntax: { DELETE | INSERT | UPDATE [OF column [, column]...] } [OR { DELETE | INSERT | UPDATE [OF column [, column]...] }... ON [schema.] { table | view }	Not supported.	-
trigger_body ::= syntax: { plsql_block | CALL routine_clause }	Supported, but with differences.	GaussDB: The plsql_block is not supported. A function can be executed only in the EXECUTE PROCEDURE function_name (arguments); format. In addition, the function needs to be defined by users and declared that it does not contain parameters and the return type is trigger. It is executed when a trigger is triggered.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
instead_of_dml_trigger ::= syntax: INSTEAD OF { DELETE | INSERT | UPDATE } [OR { DELETE | INSERT | UPDATE }]... ON [NESTED TABLE nested_table_column OF] [schema.] noneditioning_view [referencing_clause] [FOR EACH ROW] [trigger_edition_clause] [trigger_ordering_clause] [ENABLE | DISABLE] trigger_body	Supported, but with differences.	GaussDB: The NESTED TABLE nested_table_column OF, referencing_clause, trigger_edition_clause, trigger_ordering_clause, and ENABLE | DISABLE are not supported.
compound_dml_trigger ::= syntax: CREATE trigger FOR dml_event_clause ON view COMPOUND TRIGGER INSTEAD OF EACH ROW IS BEGIN statement; END INSTEAD OF EACH ROW;	Not supported.	-
system_trigger ::= syntax: { BEFORE | AFTER | INSTEAD OF } { ddl_event [OR ddl_event]... | database_event [OR database_event]... } ON { [schema.] SCHEMA | [PLUGGABLE] DATABASE } [trigger_ordering_clause] [ENABLE | DISABLE] trigger_body	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-81 ALTER trigger

Oracle Database	GaussDB	Difference
ALTER TRIGGER [schema.] trigger_name { trigger_compile_clause | { ENABLE | DISABLE } | RENAME TO new_name | { EDITIONABLE | NONEDITIONABLE } } ;	Supported, but with differences.	GaussDB: The schema, trigger_compile_clause, { ENABLE | DISABLE }, and { EDITIONABLE | NONEDITIONABLE } are not supported.

Table 3-82 drop trigger

Oracle Database	GaussDB	Difference
DROP TRIGGER [schema.] trigger ;	Supported, but with differences.	GaussDB does not support schemas. You need to add **ON table_name** to the end of **trigger_name**.

The views whose names are *_TRIGGERS in Oracle Database collect statistics about triggers, but such views in GaussDB are different. For details about such views in GaussDB, see **DB_TRIGGERS**, **ADM_TRIGGERS**, and **MY_TRIGGERS**.

Table 3-83 Compatibilities of nested, package, and standalone subprograms

Oracle Database	GaussDB	Difference
Nested subprograms (subblocks)	Supported, but with differences.	GaussDB does not support overloading. GaussDB does not support the definition of autonomous transactions. GaussDB does not support SETOF. Only one qualifier can reference nested subprograms or variables of nested subprograms.
Package subprograms	Supported.	-
Standalone subprograms (including functions and procedures)	Supported.	-
Anonymous blocks	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-84 RETURN statements

Oracle Database	GaussDB
Functions	Supported.
Procedures	Supported.
Anonymous blocks	Supported.

Table 3-85 Function-related parameters

Oracle Database	GaussDB	Difference
DETERMINISTIC	Supported, but with differences.	In GaussDB, it is IMMUTABLE.
PARALLEL_ENABLE	Not supported.	-
PIPELINED	Not supported.	-
RESULT_CACHE	Not supported.	-

Table 3-86 Parameter formats

Oracle Database	GaussDB
IN	Supported.
OUT	Supported.
IN OUT	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-87 CREATE statements

Oracle Database	GaussDB	Difference
CREATE FUNCTION	Supported, but with differences.	GaussDB does not support the IF NOT EXISTS syntax, sharing_clause, or keywords [EDITIONABLE | NONEDITIONABLE]. Only some clauses that specify the **function** attribute are supported (only the invoker_rights_clause clause is supported). For details about the syntax in GaussDB, see **CREATE FUNCTION**.
CREATE LIBRARY	Not supported.	-
CREATE PACKAGE	Supported, but with differences.	GaussDB does not support the IF NOT EXISTS syntax, sharing_clause, or keywords [EDITIONABLE | NONEDITIONABLE]. Only some clauses that specify the **package** attribute are supported (only the invoker_rights_clause clause is supported). For details about the syntax in GaussDB, see **CREATE PACKAGE**.
CREATE PACKAGE BODY	Supported, but with differences.	GaussDB does not support the IF NOT EXISTS syntax, sharing_clause, or keywords [EDITIONABLE | NONEDITIONABLE]. For details about the syntax in GaussDB, see **CREATE PACKAGE**.
CREATE PROCEDURE	Supported, but with differences.	GaussDB does not support the IF NOT EXISTS syntax, sharing_clause, or keywords [EDITIONABLE | NONEDITIONABLE]. For details about the syntax in GaussDB, see **CREATE PROCEDURE**.
CREATE TRIGGER	Supported, but with differences.	For details about the syntax in GaussDB, see **CREATE TRIGGER**.
CREATE TYPE	Supported, but with differences.	GaussDB does not support the varray, object type, and UNDER syntax. For details about the syntax in GaussDB, see **CREATE TYPE**.
CREATE TYPE BODY	Not supported.	-
CREATE OR REPLACE TYPE	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-88 ALTER statements

Oracle Database	GaussDB	Difference
ALTER FUNCTION	Supported, but with differences.	GaussDB does not support keywords [EDITIONABLE | NONEDITIONABLE], REUSE, SETTINGS, and DEBUG. For details about the syntax in GaussDB, see **ALTER FUNCTION**.
ALTER LIBRARY	Not supported.	-
ALTER PACKAGE	Supported, but with differences.	GaussDB does not support keywords [EDITIONABLE | NONEDITIONABLE], REUSE, SETTINGS, and DEBUG. For details about the syntax in GaussDB, see **ALTER PACKAGE**.
ALTER PROCEDURE	Supported, but with differences.	GaussDB does not support keywords [EDITIONABLE | NONEDITIONABLE], REUSE, SETTINGS, and DEBUG.
ALTER TRIGGER	Supported, but with differences.	In GaussDB, only the trigger name can be modified. For details about the syntax in GaussDB, see **ALTER TRIGGER**.
ALTER TYPE	Supported, but with differences.	GaussDB supports only some statements. For details about the syntax in GaussDB, see **ALTER TYPE**.

Table 3-89 DROP statements

Oracle Database	GaussDB	Difference
DROP FUNCTION	Supported.	-
DROP LIBRARY	Not supported.	-
DROP PACKAGE	Supported.	-
DROP PROCEDURE	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DROP TRIGGER	Supported, but with differences.	The syntax of GaussDB is different from that of Oracle Database. For details about the syntax in GaussDB, see **DROP TRIGGER**.
DROP TYPE	Supported, but with differences.	GaussDB does not support keywords FORCE and VALIDATE. For details about the syntax in GaussDB, see **DROP TYPE**.
DROP TYPE BODY	Not supported.	-

Table 3-90 Keywords related to functions, procedures, and anonymous blocks

Oracle Database	GaussDB	Difference
ACCESSIBLE BY	Not supported.	-
AGGREGATE	Supported, but with differences.	• GaussDB does not support Oracle Database's aggregate using [schema.] implementation_type. • For details about the usage of GaussDB, see **CREATE AGGREGATE**. The syntax is different, but the implementation functions are the same.
DETERMINISTIC	Supported, but with differences.	GaussDB supports the keyword DETERMINISTIC only in syntax, but does not support this function.
PIPE ROW	Not supported.	-
PIPELINED	Not supported.	-
SQL_MACRO	Not supported.	-
RESTRICT_REFERENCES	Not supported.	-
INLINE	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-91 Keywords related to exception handling

Oracle Database	GaussDB	Difference
EXCEPTION_INIT	Supported, but with differences.	Binding with system error codes is not supported in GaussDB. In the same stored procedure or function, if EXCEPTION_INIT is used to set the same error code, the Oracle database does not support capturing exceptions separately as conditions. GaussDB supports this behavior. For details, see **Examples**.
Exception	Supported.	-
Exception Handler	Supported.	-
SQLCODE	Supported.	-
SQLERRM	Supported.	-

Table 3-92 Other PL/SQL keywords

Oracle Database	GaussDB
COVERAGE	Not supported.
COLLATION	Supported.
DEPRECATE	Not supported.
FORALL	Supported.
NOCOPY	Not supported.
RETURNING	Supported.
SERIALLY_REUSABLE	Not supported.
SHARING	Not supported.
BULK COLLECT	Supported.

3.2.11 System Functions

Compatible functions are classified into single-row functions, user-defined functions, aggregate functions, analytic functions, object reference functions, model functions, and OLAP functions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.2.11.1 Single-Row Functions

Oracle Database	GaussDB
Table 3-93	Supported, but with differences.
Table 3-94	Supported, but with differences.
Table 3-95	Supported, but with differences.
Character set functions	Not supported.
Collation functions	Not supported.
Table 3-96	Supported, but with differences.
Table 3-97	Supported, but with differences.
Table 3-98	Supported, but with differences.
Table 3-99	Supported, but with differences.
Collection functions	Not supported.
Table 3-100	Supported.
Data mining functions	Not supported.
Table 3-101	Supported, but with differences.
Table 3-102	Supported, but with differences.
Table 3-103	Supported, but with differences.
Table 3-104	Supported.
Table 3-105	Supported, but with differences.

Table 3-93 Numeric functions

Oracle Database	GaussDB	Difference
ABS	Supported.	-
ACOS	Supported, with differences.	If the parameter is of the BINARY_DOUBLE or BINARY_FLOAT type and the value is not in the range of –1 to 1, GaussDB reports an error, while Oracle Database returns NaN.
ASIN	Supported, with differences.	If the parameter is of the BINARY_DOUBLE or BINARY_FLOAT type and the value is not in the range of –1 to 1, GaussDB reports an error, while Oracle Database returns NaN.
ATAN	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
ATAN2	Supported, with differences.	When both parameters are **0**, GaussDB returns **0**, while Oracle Database reports an error.
BITAND	Supported, with differences.	• The maximum value of the input parameter in GaussDB is of the BIGINT type, while that in Oracle Database is of the NUMBER type. • If a parameter contains a decimal, GaussDB rounds the value before calculation, while Oracle Database truncates the value before calculation.
CEIL	Supported.	-
COS	Supported, with differences.	• The data types and precision of the return values are different. GaussDB: The return value is of the float8 type, and the result precision retains 15 valid digits. Oracle Database: The return type is number. • If the parameter value is **Infinity** or **-Infinity**, GaussDB reports an error, while Oracle Database returns NaN.
COSH	Supported, with differences.	The data types and precision of the return values are different. GaussDB: If the input is of the numeric type, the return value is of the numeric type. If the input is of another type, the return value is of the float8 type, and the result precision retains 15 valid digits. Oracle Database: The return type is number.
EXP	Supported.	-
FLOOR	Supported.	The return types are different. GaussDB: • If one of the inputs is of the numeric type, the return value is of the numeric type. • For other data types, the return value is of the float8 type. Oracle Database: • The return type is numeric.
LN	Supported.	The return values are different: • Oracle Database returns a value with 37 decimal places, and GaussDB returns a value with 15 decimal places.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
LOG	Supported.	The input parameter types are different: ● GaussDB checks invalid input parameters, such as **NaN**, **0.0**, **-Infinity**, **0**, and **Infinity**.
MOD	Supported, but with differences.	● The return types are different. In Oracle Database, the return types include BINARY_DOUBLE, BINARY_FLOAT, and NUMBER. In GaussDB, the return types include int2, int4, int8, and numeric. ● The return values are different. When the input parameter is a string containing decimals (for example, '2.5'), Oracle Database implicitly converts the string to the NUMBER type and retains the decimal part for calculation, while GaussDB implicitly converts the string to the integer type and rounds the decimal part to the nearest integer for calculation. As a result, the result is different from that of Oracle Database. ● If the first input parameter is of the numeric type, the second parameter must be of the int, numeric, or a type that can be converted to numeric. If **a_format_version** is set to **10c**, **a_format_dev_version** is set to **s6**, and the first parameter is of the text type that can be converted to numeric, the second parameter must be of the numeric type that does not exceed int4 or of the character type that contains only digits.
NANVL	Supported, but with differences.	GaussDB: NaN cannot be obtained by directly declaring or dividing a floating-point number by 0.
POWER	Supported, but with differences.	The return types are different. GaussDB: ● If one of the inputs is of the numeric type, the return value is of the numeric type. If the numeric result is less than 10^16, **0** is returned. ● For other data types, the return value is of the float8 type. Oracle Database: The return type is number.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
REMAINDER	Supported, but with differences.	The return types are different. GaussDB: ● If one input value is of the float4 type and the other is of the numeric type, values of the float4 type are returned. ● If both input values are of the float4 type, values of the float4 type are returned. ● If both input values are of the float8 type, values of the float8 type are returned. ● If both inputs are of the character type, the return value is of the float8 type. ● For other data types, values of the numeric type are returned. Oracle Database: The return type is number. The return results are inconsistent: ● Oracle Database uses ROUND to obtain the quotient, while GaussDB uses FLOOR to obtain the quotient. When the decimal part of n2/n1 is 0.5, the two databases use different rounding methods. As a result, the remainder results are inconsistent.
ROUND	Supported, but with differences.	● For the first parameter **n** of the float type, GaussDB has precision loss, and the precision is lower than that of Oracle Database. ● The returned types are different. For round(n, integer), Oracle Database returns the NUMBER type, and GaussDB returns the numeric type. For round(n), Oracle Database returns the data type of **n**, and GaussDB returns only the float8 and numeric types. The float4 return type is missing. ● The logic for the GaussDB to determine that the input parameter is null and the execution framework to return null is different from that in Oracle Database. `SELECT round(NULL,'q');` Oracle Database reports null, and GaussDB reports the error: invalid input syntax for integer: "q".
SIGN	Supported, with differences.	The return types are different. GaussDB: If the input is of the numeric type, the return value is of the numeric type. Otherwise, the return value is of the float8 type. Oracle Database: The return type is number.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
SIN	Supported, with differences.	• The data types and precision of the return values are different. GaussDB: The return value is of the float8 type, and the result precision retains 15 valid digits. Oracle Database: The return type is number. • If the parameter value is **Infinity** or **-Infinity**, GaussDB reports an error, while Oracle Database returns NaN.
SINH	Supported, but with differences.	The data types and precision of the return values are different. GaussDB: If the input is of the numeric type, the return value is of the numeric type. If the input is of another type, the return value is of the float8 type, and the result precision retains 15 valid digits. Oracle Database: The return type is number.
SQRT	Supported.	-
TAN	Supported, with differences.	• The data types and precision of the return values are different. GaussDB: The return value is of the float8 type, and the result precision retains 15 valid digits. Oracle Database: The return type is number. • If the parameter value is **Infinity** or **-Infinity**, GaussDB reports an error, while Oracle Database returns NaN.
TANH	Supported, with differences.	The data types and precision of the return values are different. GaussDB: If the input is of the numeric type, the return value is of the numeric type. If the input is of another type, the return value is of the float8 type, and the result precision retains 15 valid digits. Oracle Database: The return type is number.
TRUNC	Supported.	The input parameter types are different: • In GaussDB, if the second input parameter is of the text type, the first input parameter must be of the time type. The return types are different. • In GaussDB, if the number of input parameters is 1 and the parameter is not of the numeric type, the return value is of the float8 type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
WIDTH_BUCKET	Supported.	The input parameter types are different: GaussDB: • The input parameter cannot be of a time type. • The last input parameter of the function cannot be in scientific notation.

Table 3-94 Character functions returning character values

Oracle Database	GaussDB	Difference
CHR	Supported, but with differences.	• If the entered number does not comply with the existing character set, GaussDB reports an error in JDBC and Oracle Database returns garbled characters. • If you enter **0** or **256**, Oracle Database returns characters whose ASCII code is 0, and GaussDB truncates the characters at **'\0;**. • When the input parameter is a decimal number in the range of (–1,1), the parameter is rounded to 0 in GaussDB, and the function output is null; however, in Oracle Database, an error is reported, indicating that the parameter is out of range.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
CONCAT	Supported, with differences.	• The return type of the function is different. In GaussDB, the return type is text, while in Oracle Database, the return type is dynamically determined based on the type of the input parameter. • For the concatenation of invalid characters, GaussDB does not check the validity of characters and directly concatenates all characters; however, if Oracle Database detects invalid characters, it ignores them. The || operator has similar differences. Example: -- GaussDB gaussdb=# SET a_format_version='10c'; SET gaussdb=# SET a_format_dev_version='s2'; SET gaussdb=# SELECT dump(CHR(239) || 'test') FROM dual; dump ---------- Typ=25 Len=5: 239,116,101,115,116 (1 row) -- Oracle SQL> SELECT dump(CHR(239) || 'test') FROM dual; dump(CHR(239)||'TEST') ---------- Typ=1 Len=4: 116,101,115,116
INITCAP	Supported, but with differences.	The return value is restricted by the database character set. As a result, the return result is different from that in Oracle Database. When the input parameter is EMPTY_CLOB(), GaussDB returns an empty string, while Oracle Database returns null.
LOWER	Supported, but with differences.	• The return types are different. The data types of Oracle Database are the same as the input types. • The time format is implicitly converted. When the time type is entered, the time type is implicitly converted to a character string and then the lower operation is performed. SELECT LOWER(TO_DATE('2012-12-10','YYYY-MM-DD')); Oracle Database returns 10-DEC-12, and GaussDB returns 2012-12-10 00:00:00. • The return value is restricted by the database character set. As a result, the return result is different from that in Oracle Database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
LPAD	Supported, with differences.	• GaussDB returns a text type. If the second parameter exceeds the int range, an error is reported. Oracle Database returns a varchar2 type. If the second parameter exceeds the int range, the value is truncated to 4000 characters based on the maximum length of varchar2, and the result is output normally. • For invalid characters, Oracle Database directly fills the invalid characters, while GaussDB checks the validity of the characters first. If the characters are invalid, an error is reported.
LTRIM	Supported, but with differences.	• The return types are different. If the input is of the character data type, Oracle Database returns the VARCHAR2 type. If the input is of the national character set specified during database creation, Oracle Database returns the NVARCHAR2 type. If the input is of the LOB type, Oracle Database returns the LOB type, and GaussDB returns the TEXT type. • When the source string is EMPTY_CLOB(), Oracle Database returns an empty character, while GaussDB returns null. • Oracle Database supports only the deletion of a single character, while GaussDB supports the deletion of a string. • When the character to be deleted is EMPTY_CLOB(), Oracle Database directly returns null, while GaussDB considers that the empty character is deleted and returns the source string.
NCHR	Supported, but with differences.	• The byte length of the return value is different from that of Oracle Database. • The return value is restricted by the database character set. As a result, the return result is different from that in Oracle Database. • When the byte array corresponding to the input parameter is returned, a question mark (?) is returned if a single byte ranges from 0x80 to 0xFF. In Oracle Database, a question mark (?) is returned, no output is returned, or an error is reported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
NLS_LOWER	Supported, but with differences.	• The return types are different. If the input is of the character data type, Oracle Database returns the VARCHAR2 type. If the input is of the LOB type, Oracle Database returns the LOB type. GaussDB returns the TEXT type. • In Oracle Database, the **nlsparam** parameter can be of a type except nls_sort, and no error is reported. GaussDB supports only nls_sort. • The return value is restricted by the database character set. As a result, the return result is different from that in Oracle Database.
NLS_UPPER	Supported, but with differences.	• The return types are different. If the input is of the character data type, Oracle Database returns the VARCHAR2 type. If the input is of the LOB type, Oracle Database returns the LOB type. GaussDB returns the TEXT type. • In Oracle Database, the **nlsparam** parameter can be of a type except nls_sort, and no error is reported. GaussDB supports only nls_sort. • The return value is restricted by the database character set. As a result, the return result is different from that in Oracle Database.
NLSSORT	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference

REGEXP_REPLACE	Supported, with differences.	• GaussDB input parameter **source_char** does not support the NCLOB, BLOB, or XMLTYPE type.
		• The meaning of the **'n'** option in the **match_param** input parameter is different. In GaussDB, the **'n'** option has the same meaning as the **'m'** option, indicating that the multi-row matching mode is used. In Oracle Database, it indicates that the dot (.) can match the '\n' character. If this option is not specified, the '\n' character cannot be matched by default. In GaussDB, the dot (.) matches '\n' by default. You do not need to specify the option.
		• The matching results of some regular expressions may be different.
		`SELECT REGEXP_REPLACE('abc01234xyz', '(.*?)(\d+)(.*)', '#', 'g') FROM DUAL;`
		Oracle Database reports an error, and GaussDB returns **#####xyz**.
		• The matching results may be different when Chinese characters are entered in the UTF-8 character set. Oracle Database needs to implement regular expression matching for Chinese character strings in the GBK character set.
		• The matching results of regular expressions that contain some escape characters may be different.
		`SELECT REGEXP_REPLACE('abcabc', '\abc', '#', 'g') FROM DUAL;`
		Oracle Database reports an error, and GaussDB returns **abcabc**.
		• The matching rules are affected by the **aformat_regexp_match** parameter. For details about the affected specifications, see the description of the REGEXP_REPLACE function in **Character Processing Functions and Operators**.
		• If the character class [] in a regular expression contains escape characters (such as \t, \n, \r, \s, \d, and \w), GaussDB escapes them to tab characters, while Oracle Database regards them as common characters and does not escape them.
		• For an incomplete regular expression (for example, '{2,', GaussDB strictly verifies the correctness of the regular expression and reports an error, while Oracle Database regards it as a common character.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
		• The rounding mode of the fourth and fifth parameters is different. GaussDB rounds the value to the nearest integer, while Oracle Database truncates the value to an integer. • GaussDB does not support the regular expression [A-Z]?? with more than three question marks, while Oracle Database supports it. • GaussDB supports regular expressions whose length exceeds 512 characters, while Oracle Database does not support it. • When four parameters are entered, the fourth parameter in GaussDB cannot be a number in string format, while Oracle Database supports it. • GaussDB supports the non-capturing group syntax (?:...), while Oracle Database does not support it.
REGEXP_SUBSTR	Supported, but with differences.	• The matching rules are affected by the **aformat_regexp_match** parameter. For details about the affected specifications, see the description of the REGEXP_SUBSTR function in **Character Processing Functions and Operators**. • GaussDB limits the maximum value of the regular expression A{1,xxx} to 255. • In GaussDB, the CHR(9) tab character or CHR(10) newline character in the regular expression 'x' is regarded as a blank and ignored, while in Oracle Database, the CHR(9) tab character or CHR(10) newline character is regarded as a common matching character. • GaussDB does not support six input parameters. • GaussDB supports the third parameter as a decimal number ranging from 0 to 1, while Oracle Database does not support this.
REPLACE	Supported, with differences.	• GaussDB does not support the input of the XMLTYPE type, while Oracle Database supports it. • GaussDB does not support the input of the BLOB type, while Oracle Database supports it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
RPAD	Supported, with differences.	• GaussDB returns a text type. If the second parameter exceeds the int range, an error is reported. Oracle Database returns a varchar2 type. If the second parameter exceeds the int range, the value is truncated to 4000 characters based on the maximum length of varchar2, and the result is output normally. • For invalid characters, Oracle Database directly fills the invalid characters, while GaussDB checks the validity of the characters first. If the characters are invalid, an error is reported.
RTRIM	Supported, with differences.	• When the source string is EMPTY_CLOB(), Oracle Database returns an empty character, while GaussDB returns null. • Oracle Database supports only the deletion of a single character, while GaussDB supports the deletion of a string. • When the character to be deleted is EMPTY_CLOB(), Oracle Database directly returns null, while GaussDB considers that the empty character is deleted and returns the source string.
SUBSTR	Supported, with differences.	• When the source string is EMPTY_CLOB(), GaussDB returns null, while Oracle Database returns an empty CLOB. • When the second or third parameter is a number with decimal places, GaussDB rounds the number to the nearest integer, while Oracle Database rounds the number down to the nearest integer. • When the initial index of the second parameter exceeds the total length of the source string, GaussDB reports an error, while Oracle Database returns null. • GaussDB does not support the scientific notation of a string (such as '1E2') as the second or third parameter, while Oracle Database supports it.
TRANSLATE	Supported, with differences.	When the source string is EMPTY_CLOB(), GaussDB returns an empty string, while Oracle Database returns null.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
TRIM	Supported, with differences.	• When the source string is EMPTY_CLOB(), Oracle Database returns an empty character, while GaussDB returns null. • Oracle Database requires that the keyword specifying the truncation direction or the truncation character be specified before FROM, while GaussDB allows all parameters before FROM to be omitted. • Oracle Database supports only the deletion of a single character, while GaussDB supports the deletion of a string. • When the character to be deleted is EMPTY_CLOB(), Oracle Database directly returns null, while GaussDB considers that the empty character is deleted and returns the source string.
UPPER	Supported, but with differences.	• The return types are different. The data types of Oracle Database are the same as the input types. GaussDB returns the TEXT type. • The time format is implicitly converted. When the time type is entered, the time type is implicitly converted to a character string and then the upper operation is performed. `SELECT UPPER(TO_DATE('2012-12-10','YYYY-MM-DD'));` Oracle Database returns 10-DEC-12, and GaussDB returns 2012-12-10 00:00:00. • The return value is restricted by the database character set. As a result, the return result is different from that in Oracle Database.
REVERSE	Supported, with differences.	• Oracle Database does not support the input of the CLOB, NUMBER, DATE, TIMESTAMP, TIMESTAMPTZ, INTERVAL, FLOAT, or DOUBLE data type, while GaussDB supports it. • For multi-byte characters, Oracle Database performs reversal by byte. As a result, garbled characters are displayed after conversion, while GaussDB performs conversion by character.
INSTRB	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-95 Character functions returning number values

Oracle Database	GaussDB	Difference
ASCII	Supported, but with differences.	• The return types are different. Oracle Database returns the uint4 type, while GaussDB returns the int4 type. • For invalid characters, GaussDB checks whether the characters are invalid (for example, 0x80), while Oracle Database is relatively loose and returns the ASCII code of the invalid characters. • When the input parameter is EMPTY_CLOB(), GaussDB returns **0**, while Oracle Database returns null.
INSTR	Supported, with differences.	• The third parameter in GaussDB is of the int type, and that in Oracle Database is of the numeric type. • The rules for rounding off the third and fourth parameters are different. GaussDB uses the rounding-off rule, while Oracle Database uses the integer truncation rule. • The fourth parameter in Oracle Database is truncated to an integer. When the value is a negative number or a decimal ranging from 0 to 1, the truncation result is **0** or a negative number. When the value is an invalid number, an error is reported. In GaussDB, no error is reported and **0** is returned directly. • GaussDB does not support the JSON type as the source parameter.
LENGTH	Supported, with differences.	The return types are different. Oracle Database returns the numeric type, while GaussDB returns the int type.
LENGTHB	Supported, with differences.	GaussDB supports only the text string type as the input parameter and does not support the BLOB type, while Oracle Database supports the BLOB type as the input parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
REGEXP_COUNT	Supported, but with differences.	• GaussDB input parameter **source_char** does not support the NCLOB type. • The meaning of the **'n'** option in the **match_param** input parameter is different. In GaussDB, the **'n'** option has the same meaning as the **'m'** option, indicating that the multi-row matching mode is used. In Oracle Database, it indicates that the dot (.) can match the '\n' character. If this option is not specified, the '\n' character cannot be matched by default. In GaussDB, the dot (.) matches '\n' by default. You do not need to specify the option. • The matching results of some regular expressions may be different. • The matching results may be different when Chinese characters are entered in the UTF-8 character set. Oracle Database needs to implement regular expression matching for Chinese character strings in the GBK character set. • The matching results of regular expressions that contain some escape characters may be different. • The matching rules are affected by the **aformat_regexp_match** parameter. For details about the affected specifications, see the description of the REGEXP_COUNT function in **Character Processing Functions and Operators**. • GaussDB supports the non-capturing group syntax (?:...), while Oracle Database does not. • In x extended mode, GaussDB ignores the whitespace characters (spaces and tabs) and comments in the pattern, while Oracle Database does not support this function. • If the regular expression contains an undefined escape character, such as '\x', GaussDB reports an error, while Oracle Database degrades it to a common character for matching. • When the flag is **m**, if the source string contains concatenated CHR(10), the return result is different. Oracle Database does not count the last line. • [[:blank:]] can match CHR(9) in GaussDB, but cannot match CHR(9) in Oracle Database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
REGEXP_INSTR	Supported, but with differences.	• The matching rules are affected by the **aformat_regexp_match** parameter. For details about the affected specifications, see the description of the REGEXP_INSTR function in **Character Processing Functions and Operators**. • The behavior of \s in the character class [...] is different in the two databases. The GaussDB regular expression engine (based on POSIX ERE/Perl style) supports the use of \s as a whitespace character class in [], and \s = [\t\n \r\f\v] (all whitespace characters), while the Oracle Database regular expression engine does not support the use of \s as a whitespace character class in []. • Parameter type: – GaussDB does not support the BLOB type as the source string parameter, while Oracle Database supports it. – The third, fourth, and fifth parameters in GaussDB are of the int type, and an error is reported if the value of the parameters exceeds the int range. The third, fourth, and fifth parameters in Oracle Database are of the numeric type. • Rounding mode: The third, fourth, and fifth parameters in GaussDB are rounded off, while those in Oracle Database are truncated. • Number of parameters: GaussDB does not support seven input parameters, while Oracle Database supports them.
REGEXP_LIKE	Supported, with differences.	• GaussDB does not support parameters identified by 'n'. • For an incorrect regular expression, GaussDB reports an error, while Oracle Database returns **false**. • The regular expression in GaussDB does not support '^X{4000}$' greater than 255.
LENGTHC	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DUMP	Supported, with differences.	• If the second, third, or fourth parameter is null, GaussDB directly returns null, while Oracle Database uses the default value. The default value of the second parameter is **10**, the third parameter is **1**, and the fourth parameter is **all** (that is, all bytes of the expression are read). • Parameter type: – GaussDB does not support the XMLType and name types as the first parameter, while Oracle Database supports these types. – The second, third, and fourth parameters in GaussDB are of the int type, while the second, third, and fourth parameters in Oracle Database are of the numeric type. • Rounding mode: The second, third, and fourth parameters in GaussDB are rounded off, while Oracle Database uses the rounding mode. • If the source input is null, GaussDB returns null, while Oracle Database returns the string 'NULL'.

Table 3-96 Datetime functions

Oracle Database	GaussDB	Difference
ADD_MONTHS	Supported, but with differences.	• From A.D. to B.C., the difference between GaussDB and Oracle Database is one year. • The earliest year can be –4714 in GaussDB and –4713 in Oracle Database. • When a character type literal is entered, GaussDB prefers the overloaded function of timestamp, whereas Oracle Database only has functions with input parameters of the date type.
CURRENT_DATE	Supported, but with differences.	• GaussDB: The **nls_date_format** parameter cannot be used to set the time display format. • When **mapping_date_to_datea** is enabled, the return value of **current_date** is of the datea type. When **mapping_date_to_datea** is disabled, the return value of **current_date** is of the date type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
CURRENT_TIMESTAMP	Supported, but with differences.	The value ranges from 0 to 9 in Oracle Database. The value ranges from 0 to 6 in GaussDB. The trailing zeros in microseconds are not displayed.
DBTIMEZONE	Supported, but with differences.	GaussDB: The timestamp API with the built-in tz cannot be called.
EXTRACT	Supported.	-
LAST_DAY	Supported, but with differences.	GaussDB has two overloaded functions with input parameters of the datea and timestamp types. When a literal of the character type is entered, GaussDB preferentially selects the overloaded function of the timestamp type. However, Oracle Database has only functions with input parameters of the date type.
LOCALTIMESTAMP	Supported, but with differences.	The value ranges from 0 to 9 in Oracle Database. The value ranges from 0 to 6 in GaussDB. The trailing zeros in microseconds are not displayed.
MONTHS_BETWEEN	Supported, but with differences.	GaussDB has two overloaded functions with input parameters of the datea and timestamp types. When a literal of the character type is entered, GaussDB preferentially selects the overloaded function of the timestamp type. However, Oracle Database has only functions with input parameters of the date type.
NEW_TIME	Supported, but with differences.	GaussDB has two overloaded functions with input parameters of the datea and timestamp types. When a literal of the character type is entered, GaussDB preferentially selects the overloaded function of the timestamp type. However, Oracle Database has only functions with input parameters of the date type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
NEXT_DAY	Supported.	GaussDB has two overloaded functions with input parameters of the datea and timestamp types. When a literal of the character type is entered, GaussDB preferentially selects the overloaded function of the timestamp type. However, Oracle Database has only functions with input parameters of the date type.
NUMTODSINTERVAL	Supported, but with differences.	GaussDB: The dsinterval type is not supported. Currently, interval is used to be compatible with the dsinterval type.
NUMTOYMINTERVAL	Supported, but with differences.	• GaussDB: The yminterval type is not supported. Currently, interval is used to be compatible with the yminterval type. • GaussDB does not support the BLOB type as an input parameter. • The rounding of floating-point numbers is different. As a result, when the input parameter is a floating-point number, the return results are different. -- Oracle SQL> SELECT NUMTOYMINTERVAL(0.01, 'YEAR') FROM DUAL; NUMTOYMINTERVAL(0.01,'YEAR') ---------- +000000000-01 -- GaussDB gaussdb=# SET a_format_version='10c'; SET gaussdb=# SET a_format_dev_version='s2'; SET gaussdb=# SELECT NUMTOYMINTERVAL(0.01, 'YEAR') FROM DUAL; numtoyminterval ---------- 00:00:00 (1 row)
SESSIONTIMEZONE	Supported, but with differences.	• The assignment syntax is different. In GaussDB, the SET SESSION TIME ZONE 8 syntax is used. In Oracle Database, alter session set time_zone= '+08:00' is used. • The default values are different. GaussDB: The time zone name is displayed, for example, **PRC**. The offset is used in Oracle Database, for example, **+08:00**.

Oracle Database	GaussDB	Difference
SYS_EXTRACT_UTC	Supported, with differences.	If the input time value is in the DST gap, Oracle Database reports the ORA-01878 error, while GaussDB does not report any error. The UTC time is calculated based on the time zone offset before the time shift. For example, the DST starts at 02:00:00 on October 6, 2024 in the Australia/Sydney time zone (the time changes from 02:00:00 AEST to 03:00:00 AEDT). In this case, the SYS_EXTRACT_UTC(TIMESTAMP '2024-10-06 02:00:00 Australia/Sydney') function returns an error in Oracle Database, but returns 2024-10-05 16:00:00 (calculated based on the offset UTC+10 before the time shift) in GaussDB.
SYSDATE	Supported, but with differences.	The return types are different. In GaussDB, when **mapping_date_to_datea** is disabled, the return type is timestamp without time zone; when it is enabled, the return type is datea. The return type in Oracle Database is date.
SYSTIMESTAMP	Supported, but with differences.	• GaussDB supports only six digits for millisecond calculation, while Oracle Database supports nine digits. • In GaussDB, the SYSTIMESTAMP function obtains the current time of the OS each time it is called in the same SQL statement. Therefore, different time values may be returned when the function is called multiple times in the same statement. In Oracle Database, the SYSTIMESTAMP function returns the same statement-level snapshot time in the same SQL statement.
TO_DSINTERVAL	Supported, but with differences.	• GaussDB: The dsinterval type is not supported. Currently, interval is used to be compatible with the dsinterval type. • GaussDB does not support the **nlsparam** parameter. If the **nlsparam** parameter is passed, an error indicating that the function does not exist is reported. Oracle Database supports the **nlsparam** parameter, but this parameter is reserved only for backward compatibility and has no actual function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
TO_TIMESTAMP	Supported, but with differences.	• GaussDB supports only six digits for millisecond calculation, while Oracle Database supports nine digits. If the format string specifies FF7 to FF9, Oracle Database can store and return the decimal seconds with the corresponding precision. For example, if FF7 is specified, Oracle Database returns the decimal seconds with seven digits. However, the maximum physical precision of the TIMESTAMP type in GaussDB is six digits and the decimal seconds with more than six digits are directly truncated (rather than rounded off). A maximum of six decimal digits are returned. **NOTE** This truncation behavior is different from the rounding rules of TIMESTAMP literals. For details about the rounding rules of TIMESTAMP literals, see **Data Types**. • When the input year is specified as 0000 in the *YYYY* format, the year is changed to the current year in GaussDB, and an error is reported in Oracle Database. • If the input parameter contains the text delimiter "text", GaussDB does not verify the text delimiter in the input, while Oracle Database verifies the text delimiter in the input. • If the input parameter contains the day format such as DAY/DY, GaussDB may return an unknown result. • If the decimal part of the value is **0**, the decimal places are not displayed in GaussDB, while nine decimal places are displayed in Oracle Database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
TO_TIMESTAMP_TZ	Supported, but with differences.	• The timestamptz type in GaussDB is equivalent to the timestampwithlocaltimezone type in Oracle Database. There is no exact type mapping between the two databases. The value of **nls_date_language** can only be **ENGLISH** or **AMERICAN**. • GaussDB uses the IANA time zone database (tzdata) at the bottom layer. For some countries or regions (such as PRC), the IANA database uses the local mean time (LMT) to define the local time offset of the region when the time value is far earlier than the date when the standard time was introduced (for example, before 1900). The fixed standard time zone offset is not used. Oracle Database uses the fixed standard time zone offset for dates far earlier than the date when the standard time was introduced, and does not use the LMT. Therefore, for the timestamp with time zone of a date far earlier than the date when the standard time was introduced, the time zone offset in GaussDB may be different from that in Oracle Database. -- Oracle SQL> SELECT DBTIMEZONE AS "Database", SESSIONTIMEZONE AS "Session" FROM DUAL; Dat Session --- --- PRC PRC SQL> SELECT TO_TIMESTAMP_TZ('0001-01-01 00:00:00 +00:00', 'YYYY-MM-DD HH24:MI:SS TZH:TZM') FROM DUAL; TO_TIMESTAMP_TZ('0001-01-0100:00:00+00:00','YYYY-MM-DDHH24:MI:SSTZH:TZM') --- 0001-01-01 00:00:00.000000000 +00:00 -- GaussDB gaussdb=# SET a_format_version='10c'; SET gaussdb=# SET a_format_dev_version='s1'; SET gaussdb=> SET TIMEZONE = 'Asia/Beijing'; SET gaussdb=> SELECT TO_TIMESTAMP_TZ('0001-01-01 00:00:00 +00:00', 'YYYY-MM-DD HH24:MI:SS TZH:TZM') FROM DUAL; to_timestamp_tz ---

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
		0001-01-01 08:00:00+08 (1 row) gaussdb=> SET TIMEZONE = 'PRC'; SET gaussdb=# SELECT TO_TIMESTAMP_TZ('0001-01-01 00:00:00 +00:00', 'YYYY-MM-DD HH24:MI:SS TZH:TZM') FROM DUAL; to_timestamp_tz ---------- 0001-01-01 08:05:43+08:05:43 (1 row)
TO_YMINTERVAL	Supported, but with differences.	GaussDB: The yminterval type is not supported. Currently, interval is used to be compatible with the yminterval type.
TRUNC	Supported, but with differences.	The return type in GaussDB is the same as the type of the first input parameter. In Oracle Database, the return type is always date. The supported formats are different. For details, see **Date and Time Processing Functions and Operators**.
TZ_OFFSET	Supported, with differences.	• When a time zone name is received as an input parameter, the number of time zone names supported by GaussDB is less than that supported by Oracle Database. For example, GaussDB does not support time zone abbreviations (such as PST, CST, and EST). You can use the IANA time zone region name (such as US/Pacific) instead. • When the input parameter is in the time zone offset format (such as '+hh:mi' or '-hh:mi') and the offset is zero (such as '-00:00'), Oracle Database normalizes the output to +00:00, while GaussDB retains the original sign and returns **-00:00**. • If the input parameter contains only blank characters (such as spaces), GaussDB returns an empty value, while Oracle Database reports an error.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
ROUND	Supported, but with differences.	The return types are different. In GaussDB, when **mapping_date_to_datea** is disabled, the return type is timestamp without time zone; when it is enabled, the return type is datea. The return type in Oracle Database is date. In GaussDB, the input parameter can be of the character type, whereas Oracle Database does not support an input parameter of the character type for the ROUND function. When the second input parameter has a specified unit as a year defined in the ISO 8601 standard that includes the calendar week, such as **'iyyy'** or **'iy'**, there are differences in the returned results between Oracle Database and GaussDB for certain years.

Table 3-97 General comparison functions

Oracle Database	GaussDB	Difference
GREATEST	Supported, but with differences.	• GaussDB: The comparison mode specified by the **NLS_SORT** parameter is not supported. Only binary comparison is supported. • GaussDB: Expressions in multiple languages are not supported.
LEAST	Supported, but with differences.	• GaussDB: The comparison mode specified by the **NLS_SORT** parameter is not supported. Only binary comparison is supported. • GaussDB: Expressions in multiple languages are not supported.

Table 3-98 Conversion functions

Oracle Database	GaussDB	Difference
ASCIISTR	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
CAST	Supported, but with differences.	• GaussDB: The MULTISET clause is not supported. • GaussDB: The **nlsparam** parameter is not supported. • When the CAST(expr AS NUMBER, fmt) operation is performed, GaussDB requires that the number of commas in the input data be greater than or equal to the number of commas in *fmt*. Otherwise, an error is reported. In Oracle Database, the extra commas in *fmt* are ignored. -- Oracle SELECT CAST('10' AS NUMBER, '99,99,99') FROM DUAL; -- 10 -- GaussDB gaussdb=# SET a_format_version='10c'; SET gaussdb=# SET a_format_dev_version='s1'; SET gaussdb=# SELECT CAST('10' AS NUMBER, '99,99,99') FROM DUAL; ERROR: surplus fmt. CONTEXT: referenced column: numeric
HEXTORAW	Supported.	-

RAWTOHEX	Supported, with differences.	When the input parameter of a function is of the numeric type (including integer and floating point) or time type (such as DATEA and TIMESATMP), the return results of the function are inconsistent.
		-- Oracle
		SQL> SELECT RAWTOHEX(1), RAWTOHEX(1.12) FROM DUAL;
		RAWT RAWTOH
		---- ------
		C102 C1020D
		
		SQL> SELECT RAWTOHEX(TO_DATE('1970-01-01', 'YYYY-MM-DD')) FROM DUAL;
		RAWTOHEX(TO_DATE
		---------------- 	
		B207010100000000
		
		-- GaussDB
		gaussdb=# SELECT RAWTOHEX(1), RAWTOHEX(1.12) FROM DUAL;
		rawtohex | rawtohex
		----------+----------
		31 | 312e3132
		(1 row)
		
		gaussdb=# SELECT RAWTOHEX(TO_DATE('1970-01-01', 'YYYY-MM-DD')) FROM DUAL;
		rawtohex
		-------------------- 	
		30312d4a414e2d3730
		(1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference

TO_BINARY_DOUBLE	Supported, but with differences.	• GaussDB: The **nlsparam** parameter is not supported.
		• In GaussDB, literals are parsed according to the IEEE 754 floating-point standard, and values within the target type range can be correctly returned. In Oracle Database, numeric literals (without the f/d suffix) are first converted to the NUMBER type, which is restricted by the NUMBER type range (1E-130 to 1E+126) and the maximum length of a string that can be parsed. If the value exceeds the upper limit, an error is reported. If the value is less than the lower limit, **0** is returned.
		-- Oracle
		SELECT TO_BINARY_DOUBLE(1e+400) FROM DUAL; -- The literal exceeds the NUMBER range.
		-- numeric overflow
		SELECT TO_BINARY_DOUBLE(LPAD('9', 4000, '9')) FROM DUAL; -- The string length exceeds the upper limit.
		invalid number
		SELECT TO_BINARY_DOUBLE(-2.2250738585072014E-308) FROM DUAL;
		-- 0
		
		-- GaussDB
		gaussdb=# SET a_format_version='10c';
		SET
		gaussdb=# SET a_format_dev_version='s2';
		SET
		gaussdb=# SELECT TO_BINARY_DOUBLE(1e+400) FROM DUAL;
		to_binary_double
		------------------ 	
		Infinity
		(1 row)
		
		gaussdb=# SELECT TO_BINARY_DOUBLE(LPAD('9', 4000, '9')) FROM DUAL;
		to_binary_double
		------------------ 	
		Infinity
		(1 row)
		
		gaussdb=# SELECT TO_BINARY_DOUBLE(-2.2250738585072014E-308) FROM DUAL;
		to_binary_double
		---------------------- 	
		-2.2250738585072e-308
		(1 row)
		• The differences between the **fmt** options in GaussDB and Oracle Database are as follows:
		1. GaussDB does not support the C format.
		2. GaussDB does not support the EEEE format.
		3. When the format 9 parses non-digit characters, GaussDB skips the characters, while Oracle Database reports an error.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference

		• If the **fmt** parameter is not specified, GaussDB requires that the string be in strict compliance with the floating-point literal format. Otherwise, an error is reported. Oracle Database truncates the valid prefix of a non-standard string containing characters related to the numeric format (such as hyphens (-) and plus signs (+)) and returns the valid prefix instead of reporting an error. For a string containing characters that are not in the numeric format, Oracle Database reports an error.
		`-- Oracle`
		`SELECT TO_BINARY_DOUBLE('1-23') FROM DUAL;`
		`-- 1`
		
		`-- GaussDB`
		`gaussdb=# SET a_format_version='10c';`
		`SET`
		`gaussdb=# SET a_format_dev_version='s2';`
		`SET`
		`gaussdb=# SELECT TO_BINARY_DOUBLE('1-23') FROM DUAL;`
		`ERROR:  invalid input syntax for type double precision: "1-23"`
		`CONTEXT:  referenced column: to_binary_double`
		• If the input parameter contains a null value, GaussDB directly returns null, while Oracle Database executes the function as normal.
		• The result of TO_BINARY_DOUBLE('NAN') is NAN. In the numeric operation with 0, the result in GaussDB is different from that in Oracle Database.
		– In Oracle Database, the result of the calculation between NAN and 0 is always NAN.
		– In GaussDB, the calculation between NAN and 0 complies with the normal arithmetic logic of the calculation with 0.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference

TO_BINARY_FLOAT	Supported, but with differences.	• GaussDB: The **nlsparam** parameter is not supported.
		• In GaussDB, literals are parsed according to the IEEE 754 floating-point standard, and values within the target type range can be correctly returned. In Oracle Database, numeric literals (without the f/d suffix) are first converted to the NUMBER type, which is restricted by the NUMBER type range (1E-130 to 1E+126) and the maximum length of a string that can be parsed. If the value exceeds the upper limit, an error is reported. If the value is less than the lower limit, **0** is returned.
		-- Oracle
		SELECT TO_BINARY_FLOAT(1e+400) FROM DUAL; -- The literal value exceeds the NUMBER range.
		-- numeric overflow
		SELECT TO_BINARY_FLOAT(LPAD('9', 4000, '9')) FROM DUAL; -- The string length exceeds the upper limit.
		-- invalid number
		
		-- GaussDB
		gaussdb=# SET a_format_version='10c';
		SET
		gaussdb=# SET a_format_dev_version='s2';
		SET
		gaussdb=# SELECT TO_BINARY_FLOAT(1e+400) FROM DUAL;
		to_binary_float
		---------- 	
		Infinity
		(1 row)
		
		gaussdb=# SELECT TO_BINARY_FLOAT(LPAD('9', 4000, '9')) FROM DUAL;
		to_binary_float
		---------- 	
		Infinity
		(1 row)
		• The differences between the **fmt** options in GaussDB and Oracle Database are as follows:
		1. GaussDB does not support the C format.
		2. GaussDB does not support the EEEE format.
		3. When the format 9 parses non-digit characters, GaussDB skips the characters, while Oracle Database reports an error.
		• If the **fmt** parameter is not specified, GaussDB requires that the string be in strict compliance with the floating-point literal format. Otherwise, an error is reported. Oracle Database truncates the valid prefix of a non-standard string containing characters related to the numeric format (such as hyphens (-) and plus signs (+)) and returns the valid prefix

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
		instead of reporting an error. For a string containing characters that are not in the numeric format, Oracle Database reports an error. -- Oracle SELECT TO_BINARY_FLOAT('1-23') FROM DUAL; -- 1 -- GaussDB gaussdb=# SET a_format_version='10c'; SET gaussdb=# SET a_format_dev_version='s2'; SET gaussdb=# SELECT TO_BINARY_FLOAT('1-23') FROM DUAL; ERROR: invalid input syntax for type real: "1-23" CONTEXT: referenced column: to_binary_float ● If the input parameter contains a null value, GaussDB directly returns null, while Oracle Database executes the function as normal. ● The result of TO_BINARY_FLOAT('NAN') is NAN. In the numeric operation with 0, the result in GaussDB is different from that in Oracle Database. – In Oracle Database, the result of the calculation between NAN and 0 is always NAN. – In GaussDB, the calculation between NAN and 0 complies with the normal arithmetic logic of the calculation with 0.
TO_BLOB	Supported, but with differences.	● GaussDB: The long raw type is not supported. ● GaussDB: The bfile and mime_type types are not supported.
TO_CLOB	Supported, with differences.	GaussDB does not support the BLOB type as an input parameter, while Oracle Database supports it.

Oracle Database	GaussDB	Difference

TO_DATE	Supported, but with differences.	• Multi-language parameters are not supported.
		• GaussDB does not support the BLOB type as an input parameter, while Oracle Database supports it.
		• The returned types are different. In GaussDB, when **mapping_date_to_datea** is enabled, the return type is datea; when it is disabled, the return type is timestamp.
		• In GaussDB, when **mapping_date_to_datea** is enabled, it is affected by the control parameter **nls_date_format**. Otherwise, it is not affected.
		• Some formats are not supported.
		• fmt = 'j'. The output before October 15, 1582 in Oracle Database is inconsistent with that in GaussDB.
		• If there is no separator, the value may be different from that in Oracle Database. Take to_date('220725','yymmdd') as an example. If yy/rr is parsed based on the fixed length 4, the year is parsed as 2207 and the month is parsed as 25. The month 25 is invalid, and an error will be reported.
		• If the input parameter contains a null value, GaussDB directly returns null, while Oracle Database executes the function as normal.
		• In some scenarios, Oracle Database can ignore the spaces in *fmt*, but GaussDB strictly verifies the spaces. Therefore, it is recommended that the input parameter string strictly match *fmt*.
		`-- Oracle`
		`SQL> SELECT TO_DATE('15-JANUARY -2024', 'DD-MONTH-YYYY') FROM DUAL;`
		`TO_DATE`
		`-------------------`
		`2024-01-15 00:00:00`
		
		`-- GaussDB (with a space after the month JANUARY)`
		`gaussdb=# SELECT TO_DATE('15-JANUARY -2024', 'DD-MONTH-YYYY') FROM DUAL;`
		`ERROR:  Invalid data exists in a string for matching date.`
		`CONTEXT:  referenced column: to_date`

TO_MULTI_BYTE	Supported.	-
TO_NCHAR	Supported, but with differences.	• GaussDB: The input parameter type is converted to text. • Oracle Database: The input parameter type is converted to the national character set. For other differences, see the TO_CHAR function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
TO_NUMBER	Supported, but with differences.	GaussDB does not support the **NLS_PARAM** parameter. GaussDB does not support the CLOB type as an input parameter, while Oracle Database supports it. The differences between the format options of GaussDB and Oracle Database are as follows: 1. $ GaussDB does not support this format. 2. , (comma) GaussDB: Commas (,) can appear at any position of a format. Oracle Database: ● In the format, commas can only appear in the integer part and cannot appear at the beginning of a number. In the original data, commas can appear at the beginning of a number. ● The number and position of commas in the format can be different from those in the original data, but the position of the last comma must be the same. ● Consecutive commas in the original data and the format are equivalent to no comma. ● If the original data does not contain commas, the number of digits after the last comma in the format must be the same as that in the original data. 3. B GaussDB does not support this function. 4. C GaussDB does not support the **NLS** parameter. 5. G GaussDB does not support the **NLS** parameter. 6. L GaussDB does not support the **NLS** parameter. 7. U GaussDB does not support the **NLS** parameter. 8. D GaussDB does not support the **NLS** parameter. 9. PR GaussDB: It is equivalent to S. A negative number is returned. Oracle Database:

Oracle Database	GaussDB	Difference
		• Returns the negative value in the angle brackets (< >). • Returns a positive value with leading and trailing spaces. • Limitation: PR elements can only appear at the last position of the digital format model. 10.RN | rn GaussDB does not support this function. 11.TM| TM9 | TMe GaussDB does not support this function. 12.V GaussDB does not support this function. 13.FM • In GaussDB, when there is FM, the comma in the format can be more than that in the original data. In other words, the number of commas is not necessarily the same. • In Oracle Database, spaces before and after the return value are retained. 14.EEEE GaussDB does not support this function. 15.XXXX The processing of negative input parameters is different between GaussDB and Oracle Database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference

TO_CHAR	Supported, with differences.	• GaussDB: The **nlsparam** parameter is not supported.
		• The fmt '5' is not included in Oracle Database documents and is not adapted.
		• When the input parameter is of the character type, Oracle Database returns the same type as the input parameter, while GaussDB returns the text type.
		• When the input parameter is of the INTERVAL type and **fmt** is not explicitly specified, the format of the return result in GaussDB is different from that in Oracle Database.
		`-- GaussDB`
		`gaussdb=# SELECT TO_CHAR(INTERVAL '2 12:30:45' DAY TO SECOND) FROM DUAL;`
		`to_char`
		`------------------`
		`2.52135416666667`
		`(1 row)`
		
		`-- Oracle`
		`SQL> SELECT TO_CHAR(INTERVAL '2 12:30:45' DAY TO SECOND) FROM DUAL;`
		`TO_CHAR(INTERVAL'21`
		`-------------------`
		`+02 12:30:45.000000`
		• The processing logic of zero-value placeholders is different. In Oracle Database, the specified number of zeros are retained according to the format template. In GaussDB, only one zero is returned.
		`-- GaussDB`
		`gaussdb=# SELECT TO_CHAR(0, '999V99') FROM DUAL;`
		`to_char`
		`---------`
		`0`
		`(1 row)`
		
		`-- Oracle`
		`SQL> SELECT TO_CHAR(0, '999V99') FROM DUAL;`
		`TO_CHA`
		`------`
		`00`
		• If the input parameter contains a null value, GaussDB directly returns null, while Oracle Database executes the function as normal.
		• GaussDB does not support the TZH and TZM formats.
		`-- GaussDB`
		`gaussdb=# set a_format_version = '10c';`
		`SET`
		`gaussdb=# set a_format_dev_version = 's7';`
		`SET`
		`gaussdb=# SELECT TO_CHAR(TIMESTAMP '2024-06-15 14:30:00 +08:00', 'TZH:TZM') FROM DUAL;`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference

		ERROR: invalid date format model
		CONTEXT: referenced column: to_char
		
		-- Oracle
		SELECT TO_CHAR(TIMESTAMP '2024-06-15 14:30:00 +08:00', 'TZH:TZM') FROM DUAL;
		-- +08:00
		• If the integer part of a value exceeds the number of digits that can be represented by **fmt**, the number sign (#) is used to fill the format width. The return results of GaussDB and Oracle Database are different.
		-- GaussDB
		gaussdb=# SET a_format_version='10c';
		SET
		gaussdb=# SET a_format_dev_version='s2';
		SET
		gaussdb=# SELECT TO_NCHAR(1000, '99.99MI') FROM DUAL;
		to_nchar
		---------- 	
		#####
		(1 row)
		
		-- Oracle
		SELECT TO_CHAR(-1000, '$99.99MI') FROM DUAL;
		-- #######
		• If the value is **0**, the return results of GaussDB and Oracle Database are different when **fmt** is used together with B, S, D, and (dot).
		-- GaussDB
		gaussdb=# SET a_format_version='10c';
		SET
		gaussdb=# SET a_format_dev_version='s1';
		SET
		gaussdb=# SELECT TO_CHAR(0, 'B9999.99S') FROM DUAL;
		to_char
		----------- 	
		. +
		(1 row)
		
		-- Oracle
		SELECT TO_CHAR(0, 'B9999.99S') FROM DUAL;
		-- Replace all with spaces.
		• In the format, the SP suffix is not supported. That is, the suffix cannot be converted into English words. If the suffix is specified, it does not take effect and the value is still output in number format.

TO_SINGLE_BYTE	Supported.	-
TREAT	Supported, but with differences.	In GaussDB, the period (.) operator cannot be used to obtain values, and the values cannot be converted to the object type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
UNISTR	Supported, but with differences.	GaussDB supports only UTF-8 encoding. Oracle Database supports UTF-8 and UTF-16 encodings.

Table 3-99 Large object functions

Oracle Database	GaussDB	Difference
EMPTY_BLOB	Supported.	-
EMPTY_CLOB	Supported, but with differences.	GaussDB: The CLOB type does not support the locator concept in Oracle Database.

Table 3-100 Hierarchical functions

Oracle Database	GaussDB	Difference
SYS_CONNECT_BY_PATH	Supported, but with differences.	In GaussDB, the column specified by the first input parameter can only be of the CHAR, VARCHAR, NVARCHAR2, TEXT, INT1, INT2, INT4, INT8, FLOAT4, FLOAT8, or NUMERIC type. Currently, the **col** input of the function cannot be an expression. If the column content is the same as the separator, no error is reported.
CONNECT_BY_ROOT	Supported, but with differences.	In GaussDB, the CONNECT_BY_ROOT operation has the following syntax restrictions: If the specified column name is a non-reserved keyword, the column name must be enclosed in parentheses, for example, CONNECT_BY_ROOT(*column_name*), to assign an alias to the expression. If the column name is not enclosed in parentheses, a syntax error occurs. For example, CONNECT_BY_ROOT *unreserved_keyword_column_name* AS *alias_name* needs to be changed to CONNECT_BY_ROOT (*unreserved_keyword_column_name*) AS *alias_name*.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-101 XML functions

Oracle Database	GaussDB	Difference
EXISTSNODE	Supported, but with differences.	If the input parameter has a namespace, aliases must be defined for both the XPath and namespace.
EXTRACTVALUE	Supported, but with differences.	Currently, only XPath 1.0 is supported.
SYS_XMLAGG	Supported, but with differences.	This is an alias of xmlagg and can be replaced with xmlagg.
XMLAGG	Supported, but with differences.	For details about the differences, see the note below.
XMLCOMMENT	Supported, but with differences.	
XMLCONCAT	Supported, but with differences.	
XMLELEMENT	Supported, but with differences.	For xmlelement and xmlattributes, when the value of **name** is **NULL**, the database behavior is different from that in Oracle Database. • When the **name** column of **xmlelement** is set to **NULL**, the name information is empty and the attribute information is not displayed. • When the **name** column of **xmlattributes** is set to **NULL**, the attribute information is not displayed.
XMLEXISTS	Supported, but with differences.	GaussDB input parameter is of the XML type.
XMLFOREST	Supported, but with differences.	GaussDB return value is of the XML type. GaussDB does not support the EVALNAME syntax.
XMLPARSE	Supported, but with differences.	GaussDB return value is of the XML type. GaussDB does not support the WELLFORMED syntax.
XMLROOT	Supported, but with differences.	GaussDB return value is of the XML type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
XMLSERIALIZE	Supported, with differences.	The syntax rules are different. In GaussDB, the input parameter must be followed by AS type after value, which can be omitted in Oracle Database.
XMLTABLE	Supported, but with differences.	GaussDB: ● The XPath 1.0 expression is used to select data from XML text. ● The default namespace cannot be declared. ● Multiple groups of input and aliasing are not supported. ● The RETURNING SEQUENCE BY REF and (SEQUENCE) BY REF clauses are not supported. ● The XPath query and output functionalities are implemented based on the libxml2 library. This implementation inherits the character processing mechanism of libxml2, including the escape rules for XML special characters (such as < > & " '). ● When the input is a sequence consisting of numbers and XML text, the differences are as follows: – A standard XPath and XML text cannot be combined to form a sequence for input. – The XML text can carry version information. – In scientific notation, such as '1e10', spaces are allowed between 'e' or 'E' and the exponent. Oracle Database: XQuery expressions are used to select data from XML text.
XMLQUERY	Supported, but with differences.	GaussDB: ● Only the XPath 1.0 syntax is supported. ● The root node of the XML text is fixed as the context space where text search is started. Oracle Database: XQuery expressions are used to select data from XML text.
XMLCAST	Supported, but with differences.	Some data types supported by GaussDB are different from those supported by Oracle Database, causing the variance in output data format and precision.
GETSTRINGVAL	Supported, but with differences.	For details about the differences, see the note below.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
GETCLOBVAL	Supported, but with differences.	
XMLSEQUENCE	Supported, but with differences.	

NOTE

- The XML functions listed in **Table 3-101** are implemented based on libxml2. Therefore, the behavior of these functions in GaussDB is different from that in Oracle Database in some scenarios, including XML parsing, escape rules for special characters, and XPath query result formats.

  The following example shows the differences in the output of the same XML query in GaussDB and Oracle Database.

  GaussDB in ORA-compatible mode:

```
gaussdb=# SELECT * FROM xmltable('/' passing xmltype('<a>&lt; &gt; &quot; &amp; &apos;</a>'));
       column_value
----------------------------
 <a>&lt; &gt; " &amp; '</a>+

(1 row)
```

  Oracle Database:

```
SQL> SELECT * FROM xmltable('/' passing xmltype('<a>&lt; &gt; &quot; &amp; &apos;</a>'));
COLUMN_VALUE
--------------------------------------------------------------------------------
<a>&lt; &gt; &quot; &amp; &apos;</a>
```

- When the libxml2 library on which GaussDB depends is upgraded, incompatible changes may be introduced.

Table 3-102 JSON functions

Oracle Database	GaussDB	Difference
JSON_ARRAY	Supported, but with differences.	GaussDB: The input parameter is a variable parameter list **([VARIADIC "any"])**. Oracle Database: The input parameter is more complex, that is, **(expr [FORMAT JSON] [, expr [FORMAT JSON]]... [JSON_on_null_clause] [JSON_returning_clause] [STRICT])**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
JSON_OBJECT	Supported, with differences.	GaussDB: The JSON_OBJECT([key, val[, key1, val1]...]) function is supported. If the key is duplicate, the first key-value pair is retained. Oracle Database: The input parameter is more complex, that is, (key_name VARCHAR2, value_expr [, key_name VARCHAR2, value_expr]... [JSON_on_null_clause] [JSON_returning_clause] [STRICT]). Explicit key-value pairing, independent NULL value processing policies (such as ABSENT ON NULL or NULL ON NULL), return type control, and strict mode (**STRICT**, which prevents duplicate key names or type mismatches. If duplicate keys exist and are not specified, all keys are retained by default.) are supported.

Table 3-103 Encoding and decoding functions

Oracle Database	GaussDB	Difference
DECODE	Supported, with differences.	During the comparison of conditional expressions, if one side is of the fixed-length character type and the other side is of the variable-length character type, GaussDB ignores the spaces at the end of the value (the same as the behavior of the comparison operator), while Oracle Database does not ignore the spaces at the end of the value.
DUMP	Supported, with differences.	The return results of the numeric and time types in GaussDB are inconsistent with those in Oracle Database due to different storage formats. For example: ● In GaussDB, "SELECT dump(123);" returns "Typ=23 Len=4: 123,0,0,0" (you need to run **SET a_format_version='10c';** and **SET a_format_dev_version='s2';** first). ● In Oracle Database, **SELECT dump(123) FROM dual;** returns **Typ=2 Len=3: 194,2,24**.
ORA_HASH	Supported, but with differences.	GaussDB has the following behaviors: ● The input parameter of the time type is converted into the character string type and then hashed. ● The **maxbucket** parameter is not supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
VSIZE	Supported, with differences.	• The return results of the numeric and time types in GaussDB are inconsistent with those in Oracle Database due to different storage formats. For example: – In GaussDB, **SELECT vsize(999);** returns **4** (you need to run **SET a_format_version='10c';** and **SET a_format_dev_version='s2';** first). – In Oracle Database, **SELECT vsize(999) FROM dual;** returns **3**. • GaussDB does not support input parameters of the XMLType type.

Table 3-104 Null-related functions

Oracle Database	GaussDB	Difference
COALESCE	Supported.	-
LNNVL	Supported.	-
NULLIF	Supported, with differences.	During the comparison of expressions, if one side is of the fixed-length character type and the other side is of the variable-length character type, GaussDB ignores the spaces at the end of the value (the same as the behavior of the comparison operator), while Oracle Database does not ignore the spaces at the end of the value.
NVL	Supported, with differences.	If the first expression is NULL, GaussDB returns the text type, while Oracle Database returns the data type of the second expression.
NVL2	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-105 Environment and identifier functions

Oracle Database	GaussDB	Difference
SYS_CONTEXT	Supported, with differences.	• GaussDB supports SYS_CONTEXT(namespace, attribute) and does not support the function of specifying the length of the return value using SYS_CONTEXT(namespace, attribute, len). • GaussDB returns NULL for unsupported parameters. The following parameters are not supported: – 'action' – 'is_application_root' – 'is_application_pdb' – 'audited_cursorid' – 'authenticated_identity' – 'authentication_data' – 'authentication_method' – 'cdb_domain' – 'cdb_name' – 'client_identifier' – 'con_id' – 'con_name' – 'current_sql_length' – 'db_domain' – 'db_supplemental_log_level' – 'dblink_info' – 'drain_status' – 'entryid' – 'enterprise_identity' – 'fg_job_id' – 'global_uid' – 'identification_type' – 'instance' – 'is_dg_rolling_upgrade' – 'ldap_server_type' – 'module' – 'network_protocol' – 'nls_calendar' – 'nls_sort' – 'nls_territory' – 'oracle_home'

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
		– 'os_user' – 'platform_slash' – 'policy_invoker' – 'proxy_enterprise_identity' – 'proxy_user' – 'proxy_userid' – 'scheduler_job' – 'session_edition_id' – 'session_edition_name' – 'sessionid' – 'statementid' – 'terminal' – 'unified_audit_sessionid' – 'session_default_collation' – 'client_info' – 'bg_job_id' – 'client_program_name' – 'current_bind' – 'global_context_memory' – 'host' – 'current_sqln'
SYS_GUID	Supported.	-
USER	Supported, but with differences.	The return types are different.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
USERENV	Supported, but with differences.	GaussDB returns NULL for unsupported parameters. The following parameters are not supported: • 'action' • 'is_application_root' • 'is_application_pdb' • 'audited_cursorid' • 'authenticated_identity' • 'authentication_data' • 'authentication_method' • 'cdb_domain' • 'cdb_name' • 'client_identifier' • 'con_id' • 'con_name' • 'current_sql_length' • 'db_domain' • 'db_supplemental_log_level' • 'dblink_info' • 'drain_status' • 'entryid' • 'enterprise_identity' • 'fg_job_id' • 'global_uid' • 'identification_type' • 'is_dg_rolling_upgrade' • 'ldap_server_type' • 'module' • 'network_protocol' • 'nls_calendar' • 'nls_sort' • 'nls_territory' • 'oracle_home' • 'os_user' • 'platform_slash' • 'policy_invoker' • 'proxy_enterprise_identity' • 'proxy_user'

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
		• 'proxy_userid' • 'scheduler_job' • 'session_edition_id' • 'session_edition_name' • 'sessionid' • 'statementid' • 'terminal' • 'unified_audit_sessionid' • 'session_default_collation' • 'client_info' • 'bg_job_id' • 'client_program_name' • 'current_bind' • 'global_context_memory' • 'host' • 'current_sqln'

3.2.11.2 Other Functions

Oracle Database	GaussDB
Table 3-106	Supported.
Table 3-107	Supported.
Object reference functions	Not supported.
Models functions	Not supported.
OLAP functions	Not supported.
Data cartridge functions	Not supported.
User-defined functions	Supported.

Table 3-106 Aggregate functions

Oracle Database	GaussDB	Difference
AVG	Supported.	-
CORR	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
COUNT	Supported.	-
COVAR_POP	Supported.	-
COVAR_SAMP	Supported.	-
CUME_DIST	Supported.	-
DENSE_RANK	Supported.	-
FIRST	Supported.	The KEEP syntax used in GaussDB is compatible with Oracle Database.
GROUPING	Supported.	-
LAST	Supported.	The KEEP syntax used in GaussDB is compatible with Oracle Database.
LISTAGG	Supported.	-
MAX	Supported.	-
MEDIAN	Supported.	-
MIN	Supported.	-
PERCENT_RANK	Supported.	-
PERCENTILE_CONT	Supported.	-
RANK	Supported.	-
REGR_ (Linear Regression)	Supported.	-
STDDEV	Supported.	-
STDDEV_POP	Supported.	-
STDDEV_SAMP	Supported.	-
SUM	Supported.	-
VAR_POP	Supported.	-
VAR_SAMP	Supported.	-
VARIANCE	Supported.	-
WM_CONCAT	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-107 Analytic functions

Oracle Database	GaussDB	Difference
FIRST_VALUE	Supported.	-
LAG	Supported.	-
LAST_VALUE	Supported.	-
LEAD	Supported.	-
NTH_VALUE	Supported, with differences.	• Oracle Database: The FROM FIRST|LAST syntax format is supported. • GaussDB: The FROM FIRST|LAST syntax format is not supported.
NTILE	Supported.	-
ROW_NUMBER	Supported.	-
RATIO_TO_REPORT	Supported.	-

3.2.12 System Views

GaussDB is compatible with some Oracle Database system views, as shown in the following table.

For details about columns in the system view, see **System Views**.

Table 3-108 Supported system views

Oracle Database	GaussDB
ALL_ALL_TABLES	DB_ALL_TABLES
ALL_COL_PRIVS	DB_COL_PRIVS
ALL_COLL_TYPES	DB_COLL_TYPES
ALL_IND_COLUMNS	DB_IND_COLUMNS
ALL_COL_COMMENTS	DB_COL_COMMENTS
ALL_CONS_COLUMNS	DB_CONS_COLUMNS
ALL_CONSTRAINTS	DB_CONSTRAINTS
ALL_DEPENDENCIES	DB_DEPENDENCIES
ALL_DIRECTORIES	DB_DIRECTORIES
ALL_IND_EXPRESSIONS	DB_IND_EXPRESSIONS

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
ALL_IND_PARTITIONS	DB_IND_PARTITIONS
ALL_INDEXES	DB_INDEXES
ALL_IND_SUBPARTITIONS	DB_IND_SUBPARTITIONS
ALL_OBJECTS	DB_OBJECTS
ALL_PART_COL_STATISTICS	DB_PART_COL_STATISTICS
ALL_PART_KEY_COLUMNS	DB_PART_KEY_COLUMNS
ALL_PART_TABLES	DB_PART_TABLES
ALL_SCHEDULER_JOB_ARGS	DB_SCHEDULER_JOB_ARGS
ALL_SCHEDULER_PROGRAM_ARGS	DB_SCHEDULER_PROGRAM_ARGS
ALL_SEQUENCES	DB_SEQUENCES
ALL_SUBPART_KEY_COLUMNS	DB_SUBPART_KEY_COLUMNS
ALL_SYNONYMS	DB_SYNONYMS
ALL_TAB_COL_STATISTICS	DB_TAB_COL_STATISTICS
ALL_TAB_COMMENTS	DB_TAB_COMMENTS
ALL_TAB_HISTOGRAMS	DB_TAB_HISTOGRAMS
ALL_TAB_STATS_HISTORY	DB_TAB_STATS_HISTORY
ALL_TYPES	DB_TYPES
ALL_PROCEDURES	DB_PROCEDURES
ALL_SOURCE	DB_SOURCE
ALL_TAB_COLUMNS	DB_TAB_COLUMNS
ALL_TAB_PARTITIONS	DB_TAB_PARTITIONS
ALL_TAB_SUBPARTITIONS	DB_TAB_SUBPARTITIONS
ALL_TABLES	DB_TABLES
ALL_TRIGGERS	DB_TRIGGERS
ALL_USERS	DB_USERS
ALL_VIEWS	DB_VIEWS
DBA_AUDIT_OBJECT	ADM_AUDIT_OBJECT
DBA_AUDIT_SESSION	ADM_AUDIT_SESSION
DBA_AUDIT_STATEMENT	ADM_AUDIT_STATEMENT

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
DBA_AUDIT_TRAIL	ADM_AUDIT_TRAIL
DBA_COL_COMMENTS	ADM_COL_COMMENTS
DBA_COL_PRIVS	ADM_COL_PRIVS
DBA_COLL_TYPES	ADM_COLL_TYPES
DBA_ARGUMENTS	ADM_ARGUMENTS
DBA_CONSTRAINTS	ADM_CONSTRAINTS
DBA_DATA_FILES	ADM_DATA_FILES
DBA_CONS_COLUMNS	ADM_CONS_COLUMNS
DBA_DEPENDENCIES	ADM_DEPENDENCIES
DBA_DIRECTORIES	ADM_DIRECTORIES
DBA_PART_COL_STATISTICS	ADM_PART_COL_STATISTICS
DBA_PART_TABLES	ADM_PART_TABLES
DBA_ROLE_PRIVS	ADM_ROLE_PRIVS
DBA_ROLES	ADM_ROLES
DBA_SCHEDULER_JOB_ARGS	ADM_SCHEDULER_JOB_ARGS
DBA_SCHEDULER_PROGRAMS	ADM_SCHEDULER_PROGRAMS
DBA_SCHEDULER_PROGRAM_ARGS	ADM_SCHEDULER_PROGRAM_ARGS
DBA_HIST_SNAPSHOT	ADM_HIST_SNAPSHOT
DBA_HIST_SQL_PLAN	ADM_HIST_SQL_PLAN
DBA_HIST_SQLSTAT	ADM_HIST_SQLSTAT
DBA_HIST_SQLTEXT	ADM_HIST_SQLTEXT
DBA_ILMDATAMOVEMENT-POLICIES	GS_ADM_ILMDATAMOVEMENTPOLICIES
DBA_ILMEVALUATIONDE-TAILS	GS_ADM_ILMEVALUATIONDETAILS
DBA_ILMOBJECTS	GS_ADM_ILMOBJECTS
DBA_ILMPARAMETERS	GS_ADM_ILMPARAMETERS
DBA_ILMPOLICIES	GS_ADM_ILMPOLICIES
DBA_ILMRESULTS	GS_ADM_ILMRESULTS
DBA_ILMTASKS	GS_ADM_ILMTASKS

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
DBA_IND_COLUMNS	ADM_IND_COLUMNS
DBA_IND_EXPRESSIONS	ADM_IND_EXPRESSIONS
DBA_IND_PARTITIONS	ADM_IND_PARTITIONS
DBA_INDEXES	ADM_INDEXES
DBA_OBJECTS	ADM_OBJECTS
DBA_PART_INDEXES	ADM_PART_INDEXES
DBA_PROCEDURES	ADM_PROCEDURES
DBA_SCHEDULER_JOBS	ADM_SCHEDULER_JOBS
DBA_SCHEDULER_RUNNING_JOBS	ADM_SCHEDULER_RUNNING_JOBS
DBA_SEGMENTS	ADM_SEGMENTS
DBA_SEQUENCES	ADM_SEQUENCES
DBA_SOURCE	ADM_SOURCE
DBA_IND_SUBPARTITIONS	ADM_IND_SUBPARTITIONS
DBA_SUBPART_KEY_COLUMNS	ADM_SUBPART_KEY_COLUMNS
DBA_SYS_PRIVS	ADM_SYS_PRIVS
DBA_TAB_COL_STATISTICS	ADM_TAB_COL_STATISTICS
DBA_TAB_HISTOGRAMS	ADM_TAB_HISTOGRAMS
DBA_TAB_STATISTICS	ADM_TAB_STATISTICS
DBA_TAB_STATS_HISTORY	ADM_TAB_STATS_HISTORY
DBA_TABLESPACES	ADM_TABLESPACES
DBA_TYPES	ADM_TYPES
DBA_USERS	ADM_USERS
DBA_SYNONYMS	ADM_SYNONYMS
DBA_TAB_COLS	ADM_TAB_COLS
DBA_TAB_COLUMNS	ADM_TAB_COLUMNS
DBA_TAB_COMMENTS	ADM_TAB_COMMENTS
DBA_TABLES	ADM_TABLES
DBA_TAB_PARTITIONS	ADM_TAB_PARTITIONS
DBA_TAB_SUBPARTITIONS	ADM_TAB_SUBPARTITIONS

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
DBA_TRIGGERS	ADM_TRIGGERS
DBA_TYPE_ATTRS	ADM_TYPE_ATTRS
DBA_VIEWS	ADM_VIEWS
ROLE_ROLE_PRIVS	ROLE_ROLE_PRIVS
ROLE_SYS_PRIVS	ROLE_SYS_PRIVS
ROLE_TAB_PRIVS	ROLE_TAB_PRIVS
USER_COL_COMMENTS	MY_COL_COMMENTS
USER_COL_PRIVS	MY_COL_PRIVS
USER_COLL_TYPES	MY_COLL_TYPES
USER_CONSTRAINTS	MY_CONSTRAINTS
USER_DEPENDENCIES	MY_DEPENDENCIES
DICT	DICT
DICTIONARY	DICTIONARY
DUAL	DUAL
NLS_DATABASE_PARAMETERS	NLS_DATABASE_PARAMETERS
NLS_INSTANCE_PARAMETERS	NLS_INSTANCE_PARAMETERS
PLAN_TABLE	PLAN_TABLE
USER_ILMDATAMOVEMENT-POLICIES	GS_MY_ILMDATAMOVEMENTPOLICIES
USER_ILMEVALUATIONDE-TAILS	GS_MY_ILMEVALUATIONDETAILS
USER_ILMOBJECTS	GS_MY_ILMOBJECTS
USER_ILMPOLICIES	GS_MY_ILMPOLICIES
USER_ILMRESULTS	GS_MY_ILMRESULTS
USER_ILMTASKS	GS_MY_ILMTASKS
USER_IND_COLUMNS	MY_IND_COLUMNS
USER_IND_EXPRESSIONS	MY_IND_EXPRESSIONS
USER_IND_PARTITIONS	MY_IND_PARTITIONS
USER_IND_SUBPARTITIONS	MY_IND_SUBPARTITIONS
USER_INDEXES	MY_INDEXES

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
USER_JOBS	MY_JOBS
USER_OBJECTS	MY_OBJECTS
USER_PART_COL_STATISTICS	MY_PART_COL_STATISTICS
USER_PART_INDEXES	MY_PART_INDEXES
USER_PART_TABLES	MY_PART_TABLES
USER_PROCEDURES	MY_PROCEDURES
USER_SCHEDULER_JOB_ARGS	MY_SCHEDULER_JOB_ARGS
USER_SCHEDULER_PROGRAM_ARGS	MY_SCHEDULER_PROGRAM_ARGS
USER_SEQUENCES	MY_SEQUENCES
USER_SOURCE	MY_SOURCE
USER_SUBPART_KEY_COLUMNS	MY_SUBPART_KEY_COLUMNS
USER_SYNONYMS	MY_SYNONYMS
USER_SYS_PRIVS	MY_SYS_PRIVS
USER_TAB_COL_STATISTICS	MY_TAB_COL_STATISTICS
USER_TAB_COLUMNS	MY_TAB_COLUMNS
USER_TAB_COMMENTS	MY_TAB_COMMENTS
USER_TAB_HISTOGRAMS	MY_TAB_HISTOGRAMS
USER_TAB_PARTITIONS	MY_TAB_PARTITIONS
USER_TAB_STATISTICS	MY_TAB_STATISTICS
USER_TAB_STATS_HISTORY	MY_TAB_STATS_HISTORY
USER_TABLES	MY_TABLES
USER_TABLESPACES	MY_TABLESPACES
USER_TRIGGERS	MY_TRIGGERS
USER_TYPE_ATTRS	MY_TYPE_ATTRS
USER_TYPES	MY_TYPES
USER_VIEWS	MY_VIEWS
V$NLS_PARAMETERS	V$NLS_PARAMETERS
V$SESSION_WAIT	V$SESSION_WAIT
V$SYSSTAT	V$SYSSTAT

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
V$SYSTEM_EVENT	V$SYSTEM_EVENT
V$VERSION	V$VERSION
V$INSTANCE	V_INSTANCE
GV$INSTANCE	GV_INSTANCE
V$MYSTAT	V_MYSTAT
V$SESSION	V_SESSION
GV$SESSION	GV_SESSION
V$SESSION_LONGOPS	DV_SESSION_LONGOPS
V$SESSION	DV_SESSIONS
ALL_ARGUMENTS	DB_ARGUMENTS
USER_CONS_COLUMNS	MY_CONS_COLUMNS
USER_PART_KEY_COLUMNS	MY_PART_KEY_COLUMNS
USER_ROLE_PRIVS	MY_ROLE_PRIVS
DBA_TAB_PRIVS	ADM_TAB_PRIVS
USER_SCHEDULER_JOBS	MY_SCHEDULER_JOBS
V$LOCK	V$LOCK
V$DBLINK	V$DBLINK
V$GLOBAL_TRANSACTION	V$GLOBAL_TRANSACTION
V$OPEN_CURSOR	V$OPEN_CURSOR
V$GLOBAL_OPEN_CURSOR	V$GLOBAL_OPEN_CURSOR
ALL_TAB_PRIVS	DB_TAB_PRIVS
ALL_TAB_MODIFICATIONS	DB_TAB_MODIFICATIONS
USER_TAB_MODIFICATIONS	MY_TAB_MODIFICATIONS
USER_AUDIT_TRAIL	MY_AUDIT_TRAIL

3.2.13 Advanced Packages

GaussDB is compatible with some advanced packages in Oracle Database. For details, see the following table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-109 Supported advanced packages

Oracle Database	GaussDB	Difference
Table 3-110	DBE_LOB	For details about the usage in GaussDB, see **DBE_LOB**.
Table 3-111	DBE_RANDOM	For details about the usage in GaussDB, see **DBE_RANDOM**.
Table 3-112	DBE_OUTPUT	For details about the usage in GaussDB, see **DBE_OUTPUT**.
Table 3-113	DBE_RAW	For details about the usage in GaussDB, see **DBE_RAW**.
Table 3-114	DBE_SCHEDULER	For details about the usage in GaussDB, see **DBE_SCHEDULER**.
Table 3-115	DBE_UTILITY	For details about the usage in GaussDB, see **DBE_UTILITY**.
Table 3-116	DBE_SQL	For details about the usage in GaussDB, see **DBE_SQL**.
Table 3-118	DBE_FILE	For details about the usage in GaussDB, see **DBE_FILE**.
Table 3-119	DBE_SESSION	For details about the usage in GaussDB, see **DBE_SESSION**.
Table 3-120	DBE_MATCH	For details about the usage in GaussDB, see **DBE_MATCH**.
Table 3-121	DBE_APPLICATION_INFO	For details about the usage in GaussDB, see **DBE_APPLICATION_INFO**.
Table 3-122	DBE_XMLDOM	For details about the usage in GaussDB, see **DBE_XMLDOM**.
Table 3-123	DBE_XMLPARSER	For details about the usage in GaussDB, see **DBE_XMLPARSER**.
Table 3-124	DBE_ILM	For details about the usage in GaussDB, see **DBE_ILM**.
Table 3-125	DBE_ILM_ADMIN	For details about the usage in GaussDB, see **DBE_ILM_ADMIN**.
Table 3-126	DBE_COMPRESSION	For details about the usage in GaussDB, see **DBE_COMPRESSION**.
Table 3-127	DBE_HEAT_MAP	For details about the usage in GaussDB, see **DBE_HEAT_MAP**.
Table 3-128	DBE_DESCRIBE	For details about the usage in GaussDB, see **DBE_DESCRIBE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
Table 3-130	DBE_XMLGEN	For details about the usage in GaussDB, see DBE_XMLGEN.
Table 3-129	DBE_STATS	For details about the usage in GaussDB, see DBE_STATS.
Table 3-131	DBE_OBFUSCATION_TOOLKIT	For details about the usage in GaussDB, see DBE_OBFUSCATION_TOOLKIT.

Table 3-110 DBMS_LOB compatibility

Oracle Database	GaussDB	Difference
APPEND Procedures	APPEND Procedures	-
CLOB2FILE Procedure	Not supported.	-
CLOSE Procedure	BFILECLOSE Procedure	GaussDB: The parameter type is BFILE, and no function overloading exists. Oracle Database: There are three overloaded procedures. The procedures have three parameters **lob_loc**, **lob_loc**, and **file_loc**, which are of the BLOB, CLOB CHARACTER SET ANY_CS, and BFILE types, respectively.
COMPARE Functions	COMPARE Functions	GaussDB: There are three overloaded functions. The third parameter (**len**) is of the BIGINT type. Oracle Database: There are three overloaded functions. The third parameter (**amount**) is of the INTEGER type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
CONVERTTOBLOB Procedure	LOB_CONVERTTOBLOB Procedure	GaussDB: This procedure has five parameters, and the third, fourth, and fifth parameters are of the BIGINT type. Oracle Database: This procedure has eight parameters. In addition to all GaussDB parameters, the **blob_csid**, **lang_context**, and **warning** parameters are added, which are of the NUMBER, INTEGER, and INTEGER types, respectively. The third, fourth, and fifth parameters are of the INTEGER type.
CONVERTTOCLOB Procedure	LOB_CONVERTTOCLOB Procedure	GaussDB: This procedure has five parameters. The third, fourth, and fifth parameters are of the BIGINT type. Oracle Database: This procedure has eight parameters. The third, fourth, and fifth parameters are of the INTEGER type. In addition to all GaussDB parameters, this procedure in Oracle Database adds three parameters: **blob_csid**, **lang_context**, and **warning**, which are of NUMBER, INTEGER, and INTEGER types, respectively.
COPY Procedures	LOB_COPY Functions	-
COPY_DBFS_LINK Procedures	Not supported.	-
COPY_FROM_DBFS_LINK	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
CREATETEMPORARY Procedures	CREATE_TEMPORARY Procedures	GaussDB: There are two overloaded procedures. The first parameter (**lob_loc**) of the first overloaded procedure is of the BLOB type, and the first parameter (**lob_loc**) of the second overloaded procedure is of the CLOB type. The third parameter (**dur**) of the two overloaded procedures is of the INTEGER type, and the default value is **10**. Oracle Database: There are two overloaded procedures. The first parameter (**lob_loc**) of the first overloaded procedure is of the BLOB type. The first parameter (**lob_loc**) of the second overloaded procedure is of the CLOB type. The third parameter (**dur**) of the two overloaded procedures is of the PLS_INTEGER type. The default value of **dur** of the first overloaded procedure is **DBMS_LOB.SESSION**, and the default value of **dur** of the second overloaded procedure is **10**.
DBFS_LINK_GENERATE_PATH Functions	Not supported.	-
ERASE Procedures	LOB_ERASE Procedures	-
FILECLOSE Procedure	Not supported.	-
FILECLOSEALL Procedure	Not supported.	-
FILEEXISTS Function	Not supported.	-
FILEGETNAME Procedure	Not supported.	-
FILEISOPEN Function	Not supported.	-
FILEOPEN Procedure	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
FRAGMENT_DELETE Procedure	Not supported.	-
FRAGMENT_INSERT Procedures	Not supported.	-
FRAGMENT_MOVE Procedure	Not supported.	-
FRAGMENT_REPLACE Procedures	Not supported.	-
FREETEMPORARY Procedures	Not supported.	-
GET_DBFS_LINK Functions	Not supported.	-
GET_DBFS_LINK_STATE Procedures	Not supported.	-
GETCHUNKSIZE Functions	GETCHUNKSIZE Functions	-
GETCONTENTTYPE Functions	Not supported.	-
GETLENGTH Functions	Not supported.	-
GETOPTIONS Functions	Not supported.	-
GET_STORAGE_LIMIT Function	Not supported.	-
INSTR Functions	MATCH Functions	GaussDB: There are three overloaded functions. The third and fourth parameters of the three overloaded functions are of the BIGINT type. Oracle Database: There are three overloaded functions. The third and fourth parameters of the three overloaded functions are of the INTEGER type.
ISOPEN Functions	Not supported.	-
ISREMOTE Function	Not supported.	-
ISSECUREFILE Function	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
ISTEMPORARY Functions	Not supported.	-
LOADBLOBFROMFILE Procedure	LOADBLOBFROMFILE Procedure	-
LOADCLOBFROMFILE Procedure	LOADCLOBFROMFILE Procedure	-
LOADFROMFILE Procedure	LOADFROMFILE Procedure	-
MOVE_TO_DBFS_LINK Procedures	Not supported.	-
OPEN Procedures	BFILEOPEN Procedure	GaussDB: There is no overloaded procedure. The first parameter (**bfile**) is of the DBE_LOB.BFILE type, and the second parameter (**open_mode**) is of the TEXT type. Only the read mode is supported. Oracle Database: There are three overloaded procedures. In the first overloaded procedure, the first parameter (**lob_loc**) is of the NOCOPY BLOB type, and the second parameter (**openmode**) is of the BINARY_INTEGER type. In the second overloaded procedure, the first parameter (**lob_loc**) is of the NOCOPY CLOB CHARACTER SET ANY_CS type, and the second parameter (**openmode**) is of the BINARY_INTEGER type. In the third overloaded procedure, the first parameter (**file_loc**) is of the NOCOPY BFILE type, and the second parameter (**openmode**) is of the BINARY_INTEGER type, and the value can only be **file_readonly**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
READ Procedures	READ Procedures	GaussDB: There are two overloaded procedures. Oracle Database: There are three overloaded procedures. The first two overloaded procedures are the same as those in GaussDB. The third overloaded procedure includes four parameters: **file_loc**, **amount**, **offset**, and **buffer**, which are of the BFILE, NOCOPY INTEGER, INTEGER, and RAW types, respectively.
SET_DBFS_LINK Procedures	Not supported.	-
SETCONTENTTYPE Procedure	Not supported.	-
SETOPTIONS Procedures	Not supported.	-
SUBSTR Functions	LOB_SUBSTR Functions	-
TRIM Procedures	STRIP Functions	GaussDB: There are two overloaded procedures. The second parameter (**newlen**) of the two overloaded procedures is of the BIGINT type. Oracle Database: There are two overloaded procedures. The second parameter (**newlen**) of the two overloaded procedures is of the INTEGER type.
WRITE Procedures	WRITE Functions	-
WRITEAPPEND Procedures	WRITEAPPEND Functions	-

Table 3-111 DBMS_RANDOM compatibility

Oracle Database	GaussDB	Difference
INITIALIZE Procedure	Not supported.	-
NORMAL Function	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
RANDOM Function	Not supported.	-
SEED Procedures	DBE_RANDOM.SET_SEED Function	GaussDB: There is no overloaded function. The parameter is of the INTEGER type. Oracle Database: There are two overloaded procedures. The parameter types of the two overloaded procedures are VARCHAR2 and BINARY_INTEGER, respectively.
STRING Function	Not supported.	-
TERMINATE Procedure	Not supported.	-
VALUE Functions	DBE_RANDOM.GET_VALUE Function	GaussDB: There is no overloaded function. Oracle Database: The VALUE function without parameters is overloaded, and the return type is NUMBER.

Table 3-112 DBMS_OUTPUT compatibility

Oracle Database	GaussDB	Difference
DISABLE Procedure	DISABLE Function	-
ENABLE Procedure	ENABLE Function	-
GET_LINE Procedure	GET_LINE Function	GaussDB: There is no overloaded function. The first parameter (**lines**) is of the VARCHAR[] type. Oracle Database: There are two overloaded procedures. The first parameters (**lines**) of the two overloaded procedures are of the CHARARR and DBMSOUTPUT_LINESARRAY types, respectively.
GET_LINES Procedure	GET_LINES Function	-
NEW_LINE Procedure	NEW_LINE Function	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
PUT Procedure	PUT Function	GaussDB: If the character set of the database server (**server_encoding**) is not encoded in UTF-8 and the character encoding of the input parameter is valid UTF-8, this function converts character encoding based on the relationship "UTF-8 > server_encoding" and then outputs the result regardless of the data type of the input parameter. Oracle Database: If the character set of the database server (**server_encoding**) is not encoded in UTF-8, the character encoding of the input parameter is valid UTF-8, and the input parameter type is NVARCHAR2, this procedure coverts the character encoding based on the relationship "UTF-8 > server_encoding" and then outputs the result. If the input parameter is of other character types, the character encoding is regarded as an invalid character and output as a placeholder.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
PUT_LINE Procedure	PUT_LINE Function	GaussDB: If the character set of the database server (**server_encoding**) is not encoded in UTF-8 and the character encoding of the input parameter is valid UTF-8, this function converts character encoding based on the relationship "UTF-8 > server_encoding" and then outputs the result regardless of the data type of the input parameter. Oracle Database: If the character set of the database server (**server_encoding**) is not encoded in UTF-8, the character encoding of the input parameter is valid UTF-8, and the input parameter type is NVARCHAR2, this procedure coverts the character encoding based on the relationship "UTF-8 > server_encoding" and then outputs the result. If the input parameter is of other character types, the character encoding is regarded as an invalid character and output as a placeholder.

Table 3-113 UTL_RAW compatibility

Oracle Database	GaussDB	Difference
BIT_AND Function	BIT_AND Function	-
BIT_COMPLEMENT Function	BIT_COMPLEMENT Function	-
BIT_OR Function	BIT_OR Function	GaussDB: The two parameters are defined as the TEXT type and returned as the TEXT type. Oracle Database: The two parameters are of the RAW type and returned as the RAW type.
BIT_XOR Function	BIT_XOR Function	-
CAST_FROM_BINARY_DOUBLE Function	CAST_FROM_BINARY_DOUBLE_TO_RAW Function	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
CAST_FROM_BINARY_FLOAT Function	CAST_FROM_BINARY_FLOAT_TO_RAW Function	GaussDB: The **n** parameter is of the FLOAT4 type, and it is truncated based on its range and precision before conversion. Oracle Database: The **n** parameter is of the FLOAT type.
CAST_FROM_BINARY_INTEGER Function	CAST_FROM_BINARY_INTEGER_TO_RAW Function	GaussDB: The **value** parameter is of the BIGINT type. Oracle Database: The **value** parameter is of the INTEGER type.
CAST_FROM_NUMBER Function	CAST_FROM_NUMBER_TO_RAW Function	GaussDB: The **n** parameter is of the NUMERIC type. Oracle Database: The **n** parameter is of the NUMBER type.
CAST_TO_BINARY_DOUBLE Function	CAST_FROM_RAW_TO_BINARY_DOUBLE Function	-
CAST_TO_BINARY_FLOAT Function	CAST_FROM_RAW_TO_BINARY_FLOAT Function	GaussDB: The function returns the FLOAT4 type. Oracle Database: The function returns the FLOAT type.
CAST_TO_BINARY_INTEGER Function	CAST_FROM_RAW_TO_BINARY_INTEGER Function	GaussDB: The **endianess** parameter is of the INTEGER type, and the function returns the INTEGER type. Oracle Database: The **endianess** parameter is of the PLS_INTEGER type, and the function returns the BINARY_INTEGER type.
CAST_TO_NUMBER Function	CAST_FROM_RAW_TO_NUMBER Function	GaussDB: The function returns the NUMERIC type. Oracle Database: The function returns the NUMBER type.
CAST_TO_NVARCHAR2 Function	CAST_FROM_RAW_TO_NVARCHAR2 Function	-
CAST_TO_RAW Function	CAST_FROM_VARCHAR2_TO_RAW Function	-
CAST_TO_VARCHAR2 Function	CAST_TO_VARCHAR2 Function	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
COMPARE Function	COMPARE Function	GaussDB: The function returns the INTEGER type. Oracle Database: The function returns the NUMBER type.
CONCAT Function	CONCAT Function	-
CONVERT Function	CONVERT Function	-
COPIES Function	COPIES Function	GaussDB: The **n** parameter is of the NUMERIC type. Oracle Database: The **n** parameter is of the NUMBER type.
LENGTH Function	GET_LENGTH Function	-
OVERLAY Function	OVERLAY Function	-
REVERSE Function	REVERSE Function	-
SUBSTR Function	SUBSTR Function	GaussDB: The **lob_loc** parameter is of the BLOB type. The **off_set** parameter is of the INTEGER type and its default value is **1**. The **amount** parameter is of the INTEGER type and its default value is **32767**. Oracle Database: The **r** parameter is of the RAW type. The **pos** parameter is of the BINARY_INTEGER type and has no default value. The **len** parameter is of the BINARY_INTEGER type and its default value is **NULL**.
TRANSLATE Function	TRANSLATE Function	-
TRANSLITERATE Function	TRANSLITERATE Function	-
XRANGE Function	XRANGE Function	GaussDB: The **start_byte** and **end_byte** parameters do not have default values. Oracle Database: The default values of the **start_byte** and **end_byte** parameters are **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-114 DBMS_SCHEDULER compatibility

Oracle Database	GaussDB
ADD_EVENT_QUEUE_SUBSCRIBER Procedure	Not supported.
ADD_GROUP_MEMBER Procedure	Not supported.
ADD_JOB_EMAIL_NOTIFICATION Procedure	Not supported.
ADD_TO_INCOMPATIBILITY Procedure	Not supported.
ALTER_CHAIN Procedure	Not supported.
ALTER_RUNNING_CHAIN Procedure	Not supported.
CLOSE_WINDOW Procedure	Not supported.
COPY_JOB Procedure	Not supported.
CREATE_CHAIN Procedure	Not supported.
CREATE_CREDENTIAL Procedure	CREATE_CREDENTIAL Procedure
CREATE_DATABASE_DESTINATION Procedure	Not supported.
CREATE_EVENT_SCHEDULE Procedure	Not supported.
CREATE_FILE_WATCHER Procedure	Not supported.
CREATE_GROUP Procedure	Not supported.
CREATE_INCOMPATIBILITY Procedure	Not supported.
CREATE_JOB Procedure	CREATE_JOB Procedure
CREATE_JOB_CLASS Procedure	CREATE_JOB_CLASS Procedure
CREATE_JOBS Procedure	Not supported.
CREATE_PROGRAM Procedure	CREATE_PROGRAM Procedure
CREATE_RESOURCE Procedure	Not supported.
CREATE_SCHEDULE Procedure	CREATE_SCHEDULE Procedure
CREATE_WINDOW Procedure	Not supported.
DEFINE_ANYDATA_ARGUMENT Procedure	Not supported.
DEFINE_CHAIN_EVENT_STEP Procedure	Not supported.
DEFINE_CHAIN_RULE Procedure	Not supported.
DEFINE_CHAIN_STEP Procedure	Not supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
DEFINE_METADATA_ARGUMENT Procedure	Not supported.
DEFINE_PROGRAM_ARGUMENT Procedure	DEFINE_PROGRAM_ARGUMENT Procedure
DISABLE Procedure	DISABLE Procedure
DROP_AGENT_DESTINATION Procedure	Not supported.
DROP_CHAIN Procedure	Not supported.
DROP_CHAIN_RULE Procedure	Not supported.
DROP_CHAIN_STEP Procedure	Not supported.
DROP_CREDENTIAL Procedure	DROP_CREDENTIAL Procedure
DROP_DATABASE_DESTINATION Procedure	Not supported.
DROP_FILE_WATCHER Procedure	Not supported.
DROP_GROUP Procedure	Not supported.
DROP_INCOMPATIBILITY Procedure	Not supported.
DROP_JOB Procedure	DROP_JOB Procedure
DROP_JOB_CLASS Procedure	DROP_JOB_CLASS Procedure
DROP_PROGRAM Procedure	DROP_PROGRAM Procedure
DROP_PROGRAM_ARGUMENT Procedure	Not supported.
DROP_SCHEDULE Procedure	DROP_SCHEDULE Procedure
DROP_WINDOW Procedure	Not supported.
ENABLE Procedure	ENABLE Procedure
END_DETACHED_JOB_RUN Procedure	Not supported.
EVALUATE_CALENDAR_STRING Procedure	EVALUATE_CALENDAR_STRING Procedure
EVALUATE_RUNNING_CHAIN Procedure	Not supported.
GENERATE_JOB_NAME Function	GENERATE_JOB_NAME Function
GET_AGENT_INFO Function	Not supported.
GET_AGENT_VERSION Function	Not supported.
GET_ATTRIBUTE Procedure	Not supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB
GET_FILE Procedure	Not supported.
GET_SCHEDULER_ATTRIBUTE Procedure	Not supported.
OPEN_WINDOW Procedure	Not supported.
PURGE_LOG Procedure	Not supported.
PUT_FILE Procedure	Not supported.
REMOVE_EVENT_QUEUE_SUBSCRIBER Procedure	Not supported.
REMOVE_FROM_INCOMPATIBILITY Procedure	Not supported.
REMOVE_GROUP_MEMBER Procedure	Not supported.
REMOVE_JOB_EMAIL_NOTIFICATION Procedure	Not supported.
RESET_JOB_ARGUMENT_VALUE Procedure	Not supported.
RUN_CHAIN Procedure	Not supported.
RUN_JOB Procedure	RUN_JOB Procedure
SET_AGENT_REGISTRATION_PASS Procedure	Not supported.
SET_ATTRIBUTE Procedure	SET_ATTRIBUTE Procedure
SET_ATTRIBUTE_NULL Procedure	Not supported.
SET_JOB_ANYDATA_VALUE Procedure	Not supported.
SET_JOB_ARGUMENT_VALUE Procedure	SET_JOB_ARGUMENT_VALUE Procedure
SET_JOB_ATTRIBUTES Procedure	Not supported.
SET_RESOURCE_CONSTRAINT Procedure	Not supported.
SET_SCHEDULER_ATTRIBUTE Procedure	Not supported.
STOP_JOB Procedure	STOP_JOB Procedure

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-115 DBMS_UTILITY compatibility

Oracle Database	GaussDB	Difference
ACTIVE_INSTANCES Procedure	Not supported.	-
ANALYZE_DATABASE Procedure	Not supported.	-
ANALYZE_PART_OBJECT Procedure	Not supported.	-
ANALYZE_SCHEMA Procedure	Not supported.	-
CANONICALIZE Procedure	CANONICALIZE Procedure	GaussDB: The default size of the **canon_len** parameter is 1024 bytes. Oracle Database: The **canon_len** parameter has no default value.
COMMA_TO_TABLE Procedures	COMMA_TO_TABLE Procedures	GaussDB: The **tab** parameter is a VARCHAR2 array. Oracle Database: There are two overloaded procedures. The **tab** parameter can be of the uncl_array or lname_array type.
COMPILE_SCHEMA Procedure	Not supported.	-
CREATE_ALTER_TYPE_ERROR_TABLE Procedure	Not supported.	-
CURRENT_INSTANCE Function	Not supported.	-
DATA_BLOCK_ADDRESS_BLOCK Function	Not supported.	-
DATA_BLOCK_ADDRESS_FILE Function	Not supported.	-
DB_VERSION Procedure	DB_VERSION Procedure	GaussDB: There is only the **version** parameter, which is of the VARCHAR2 type. Oracle Database: There are the **version** and **compatibility** parameters, which are of the VARCHAR2 type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
EXEC_DDL_STATEMENT Procedure	EXEC_DDL_STATEMENT Function	GaussDB: The **parse_string** parameter is of the TEXT type. Oracle Database: The **parse_string** parameter is of the VARCHAR2 type.
EXPAND_SQL_TEXT Procedure	EXPAND_SQL_TEXT Function	GaussDB: The **output_sql_text** parameter is of the CLOB type. Oracle Database: The **output_sql_text** parameter is of the NOCOPY CLOB type. The **OUT** parameter is passed by reference.
FORMAT_CALL_STACK Function	FORMAT_CALL_STACK Function	GaussDB: The function returns the TEXT type. Oracle Database: The function returns the VARCHAR2 type.
FORMAT_ERROR_BACKTRACE Function	FORMAT_ERROR_BACKTRACE Function	GaussDB: The function returns the TEXT type. Oracle Database: The function returns the VARCHAR2 type.
FORMAT_ERROR_STACK Function	FORMAT_ERROR_STACK Function	GaussDB: The function returns the TEXT type. Oracle Database: The function returns the VARCHAR2 type.
GET_CPU_TIME Function	GET_CPU_TIME Function	GaussDB: The function returns the BIGINT type. Oracle Database: The function returns the NUMBER type.
GET_DEPENDENCY Procedure	Not supported.	-
GET_ENDIANNESS Function	GET_ENDIANNESS Function	GaussDB: The function returns the INTEGER type. Oracle Database: The function returns the NUMBER type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
GET_HASH_VALUE Function	GET_HASH_VALUE Function	GaussDB: The **base** and **hash_size** parameter values and the return value are all of the INTEGER type. Oracle Database: The **base** and **hash_size** values and the return value are all of the NUMBER type.
GET_PARAMETER_VALUE Function	Not supported.	-
GET_SQL_HASH Function	GET_SQL_HASH Function	GaussDB: The **last4bytes** parameter of the BIGINT type specifies the last four bytes of an MD5 hash value and is displayed as an unsigned integer. The function returns the BIGINT type. Oracle Database: The **pre10ihash** parameter of the NUMBER type is used to store the 4-byte hash value among the 16 bytes calculated by MD5.
GET_TIME Function	GET_TIME Function	GaussDB: The function returns the BIGINT type. Oracle Database: The function returns the NUMBER type.
GET_TZ_TRANSITIONS Procedure	Not supported.	-
INVALIDATE Procedure	Not supported.	-
IS_BIT_SET Function	IS_BIT_SET Function	GaussDB: The **n** parameter and return type are both INTEGER. Oracle Database: The **n** parameter and return type are both NUMBER.
IS_CLUSTER_DATABASE Function	IS_CLUSTER_DATABASE Function	-
MAKE_DATA_BLOCK_ADDRESS Function	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
NAME_RESOLVE Procedure	NAME_RESOLVE Procedure	GaussDB: The **context** and **part1_type** parameters are of the INTEGER type, and the **object_number** parameter is of the OID type. GaussDB does not support implicit conversion from NUMBER to OID. Oracle Database: The **context**, **part1_type**, and **object_number** parameters are of the NUMBER type.
NAME_TOKENIZE Procedure	NAME_TOKENIZE Procedure	GaussDB: The **nextpos** parameter is of the INTEGER type. Oracle Database: The **nextpos** parameter is of the BINARY_INTEGER type.
OLD_CURRENT_SCHEMA Function	OLD_CURRENT_SCHEMA Function	GaussDB: The function returns the VARCHAR type. Oracle Database: The function returns the VARCHAR2 type.
OLD_CURRENT_USER Function	OLD_CURRENT_USER Function	GaussDB: The function returns the TEXT type. Oracle Database: The function returns the VARCHAR2 type.
PORT_STRING Function	Not supported.	-
SQLID_TO_SQLHASH Function	Not supported.	-
TABLE_TO_COMMA Procedures	TABLE_TO_COMMA Procedures	GaussDB: The **tab** parameter is a VARCHAR2 array. Oracle Database: There are two overloaded stored procedures. The **tab** parameter can be of the uncl_array or lname_array type.
VALIDATE Procedure	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
WAIT_ON_PENDING_DML Function	Not supported.	-

Table 3-116 DBMS_SQL compatibility

Oracle Database	GaussDB	Difference
BIND_ARRAY Procedures	SQL_BIND_ARRAY Function	-
BIND_VARIABLE Procedures	SQL_BIND_VARIABLE Function	-
BIND_VARIABLE_PKG Procedure	Not supported.	-
CLOSE_CURSOR Procedure	SQL_UNREGISTER_CONTEXT Function	-
COLUMN_VALUE Procedure	GET_RESULT Procedure	-
COLUMN_VALUE_LONG Procedure	Not supported.	-
DEFINE_ARRAY Procedure	SET_RESULTS_TYPE Procedure	-
DEFINE_COLUMN Procedures	SET_RESULT_TYPE Procedure	-
DEFINE_COLUMN_CHAR Procedure	Not supported.	-
DEFINE_COLUMN_LONG Procedure	Not supported.	-
DEFINE_COLUMN_RAW Procedure	Not supported.	-
DEFINE_COLUMN_ROWID Procedure	Not supported.	-
DESCRIBE_COLUMNS Procedure	DESCRIBE_COLUMNS Procedure	-
DESCRIBE_COLUMNS2 Procedure	Not supported.	-
DESCRIBE_COLUMNS3 Procedure	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
EXECUTE Function	SQL_RUN Function	• GaussDB: The return value is a constant **1**. Currently, the comparison between unknown types in the statement cannot return correct results. • Oracle Database: The return value is the number of affected rows for INSERT, UPDATE, and DELETE statements and is meaningless for other statements.
EXECUTE_AND_FETCH Function	RUN_AND_NEXT Function	-
FETCH_ROWS Function	NEXT_ROW Function	-
GET_NEXT_RESULT Procedures	Not supported.	-
IS_OPEN Function	IS_ACTIVE Function	-
LAST_ERROR_POSITIO N Function	Not supported.	-
LAST_ROW_COUNT Function	LAST_ROW_COUNT Function	-
LAST_ROW_ID Function	Not supported.	-
LAST_SQL_FUNCTION_ CODE Function	Not supported.	-
OPEN_CURSOR Functions	REGISTER_CONTEXT Function	-
PARSE Procedures	Supported, but with differences.	In GaussDB, the SQL_SET_SQL function does not support overloading.
RETURN_RESULT Procedures	Not supported.	-
TO_CURSOR_NUMBER Function	Not supported.	-
TO_REFCURSOR Function	Not supported.	-
VARIABLE_VALUE Procedures	GET_VARIABLE_RESUL T Procedures	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
VARIABLE_VALUE_PKG Procedure	Not supported.	-

Table 3-117 DBMS_SQL data type compatibility

Oracle Database	GaussDB
DBMS_SQL DESC_REC	DBE_SQL.DESC_REC
DBMS_SQL DATE_TABLE	DBE_SQL.DATE_TABLE
DBMS_SQL NUMBER_TABLE	DBE_SQL.NUMBER_TABLE
DBMS_SQL VARCHAR2_TABLE	DBE_SQL.VARCHAR2_TABLE
DBMS_SQL BLOB_TABLE	DBE_SQL.BLOB_TABLE

Table 3-118 UTL_FILE compatibility

Oracle Database	GaussDB	Difference
FCLOSE Procedure	CLOSE Procedure	-
FCLOSE_ALL Procedure	CLOSE_ALL Procedure	-
FCOPY Procedure	COPY Procedure	-
FFLUSH Procedure	FLUSH Procedure	-
FGETATTR Procedure	GET_ATTR Procedure	-
FGETPOS Function	GET_POS Function	-
FOPEN Function	FOPEN Function	-
FOPEN_NCHAR Function	FOPEN_NCHAR Function	-
FREMOVE Procedure	REMOVE Procedure	-
FRENAME Procedure	RENAME Procedure	-
FSEEK Procedure	SEEK Procedure	-
GET_LINE Procedure	READ_LINE Procedure	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
GET_LINE_NCHAR Procedure	READ_LINE_NCHAR Procedure	-
GET_RAW Procedure	GET_RAW Procedure	-
IS_OPEN Function	IS_OPEN Function	-
NEW_LINE Procedure	Supported, but with differences in the NEW_LINE function.	GaussDB defines the API as a function.
PUT Procedure	Supported, but with differences in the WRITE function.	GaussDB defines the API as a function.
PUT_LINE Procedure	Supported, but with differences in the WRITE_LINE function.	GaussDB defines the API as a function.
PUT_LINE_NCHAR Procedure	Supported, but with differences in the WRITE_LINE_NCHAR function.	GaussDB defines the API as a function.
PUT_NCHAR Procedure	Supported, but with differences in the WRITE_NCHAR function.	GaussDB defines the API as a function.
PUTF Procedure	Supported, but with differences in the FORMAT_WRITE function.	GaussDB defines the API as a function.
PUTF_NCHAR Procedure	Supported, but with differences in the FORMAT_WRITE_NCHAR function.	GaussDB defines the API as a function.
PUT_RAW Procedure	Supported, but with differences in the PUT_RAW function.	GaussDB defines the API as a function.

Table 3-119 DBMS_SESSION compatibility

Oracle Database	GaussDB	Difference
CLEAR_ALL_CONTEXT Procedure	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
CLEAR_CONTEXT Procedure	CLEAR_CONTEXT Function	-
CLEAR_IDENTIFIER Procedure	Not supported.	-
CLOSE_DATABASE_LINK Procedure	Not supported.	-
CURRENT_IS_ROLE_ENABLED Function	Not supported.	-
FREE_UNUSED_USER_MEMORY Procedure	Not supported.	-
GET_PACKAGE_MEMORY_UTILIZATION Procedure	Not supported.	-
IS_ROLE_ENABLED Function	Not supported.	-
IS_SESSION_ALIVE Function	Not supported.	-
LIST_CONTEXT Procedures	Not supported.	-
MODIFY_PACKAGE_STATE Procedure	MODIFY_PACKAGE_STATE Procedure	GaussDB: The scenario where **flags** is set to **1** is supported. Oracle Database: The scenario where **flags** is set to **1** or **2** is supported.
RESET_PACKAGE Procedure	Not supported.	-
SESSION_IS_ROLE_ENABLED Function	Not supported.	-
SESSION_TRACE_DISABLE Procedure	Not supported.	-
SESSION_TRACE_ENABLE Procedure	Not supported.	-
SET_CONTEXT Procedure	SET_CONTEXT Function	GaussDB: There are the **namespace**, **attribute**, and **value** parameters of the TEXT type. Oracle Database: There are the **namespace**, **attribute**, **value**, **username**, and **client_id** parameters of the VARCHAR2 type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
SET_EDITION_DEFERRED Procedure	Not supported.	-
SET_IDENTIFIER Procedure	Not supported.	-
SET_NLS Procedure	Not supported.	-
SET_ROLE Procedure	Not supported.	-
SET_SQL_TRACE Procedure	Not supported.	-
SLEEP Procedure	Not supported.	-
SWITCH_CURRENT_CONSUMER_GROUP Procedure	Not supported.	-
UNIQUE_SESSION_ID Function	Not supported.	-

Table 3-120 UTL_MATCH compatibility

Oracle Database	GaussDB	Difference
EDIT_DISTANCE Function	Not supported.	-
EDIT_DISTANCE_SIMILARITY Function	EDIT_DISTANCE_SIMILARITY Function	GaussDB: The **str1** and **str2** parameters are of the TEXT type, and the function returns the INTEGER type. Oracle Database: The **s1** and **s2** parameters are of the VARCHAR2 type, and the function returns the PLS_INTEGER type.
JARO_WINKLER Function	Not supported.	-
JARO_WINKLER_SIMILARITY Function	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-121 DBMS_APPLICATION_INFO compatibility

Oracle Database	GaussDB	Difference
READ_CLIENT_INFO Function	READ_CLIENT_INFO Procedure	GaussDB: The **client_info** parameter is of the TEXT type. Oracle Database: The **client_info** parameter is of the VARCHAR2 type.
READ_MODULE Procedure	READ_MODULE Procedure	GaussDB: The **module_name** and **action_name** parameters are of the TEXT type. Oracle Database: The **module_name** and **action_name** parameters are of the VARCHAR2 type.
SET_ACTION Procedure	SET_ACTION Procedure	GaussDB: The **action_name** parameter is of the TEXT type. Oracle Database: The **action_name** parameter is of the VARCHAR2 type.
SET_CLIENT_INFO Function	SET_CLIENT_INFO Procedure	GaussDB: The **str** parameter is of the TEXT type, and the return type is VOID. Oracle Database: The **client_info** parameter is of the VARCHAR2 type and no value is returned. Both of them are written to the client. The maximum length is 64 bytes. If the length exceeds 64 bytes, it will be truncated.
SET_MODULE Procedure	SET_MODULE Procedure	GaussDB: The **module_name** and **action_name** parameters are of the TEXT type. Oracle Database: The **module_name** and **action_name** parameters are of the VARCHAR2 type.
SET_SESSION_LONGOPS Procedure	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-122 DBMS_XMLDOM compatibility

Oracle Database	GaussDB	Difference
DBMS_XMLDOM.APPENDCHILD	DBE_XMLDOM.APPENDCHILD	• GaussDB: The error message "operation not support" is displayed for the APPEND ATTR node under the DOCUMENT node. Oracle Database: No error is reported in this scenario, but the mounting fails. • GaussDB: The error message "operation not support" is displayed for the APPEND ATTR node under the ATTR node. Oracle Database: No error is reported in this scenario, but the mounting fails. • GaussDB: When multiple child nodes of the ATTR type are added to a parent node, the child nodes with the same key value cannot exist under the same parent node. Oracle Database: Child nodes with the same key value can exist under the same parent node.
DBMS_XMLDOM.CREATEELEMENT	DBE_XMLDOM.CREATEELEMENT	-
DBMS_XMLDOM.CREATETEXTNODE	DBE_XMLDOM.CREATETEXTNODE	-
DBMS_XMLDOM.FREEDOCUMENT	Supported, but with differences in DBE_XMLDOM.FREEDOCUMENT.	GaussDB: Objects are not released immediately. They are released after a certain number of objects are accumulated. All nodes in the document are invalid. Oracle Database: The object is released immediately.
DBMS_XMLDOM.FREEELEMENT	DBE_XMLDOM.FREEELEMENT	-
DBMS_XMLDOM.FREENODE	DBE_XMLDOM.FREENODE	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_XMLDOM.FREENODELIST	Supported, but with differences in DBE_XMLDOM.FREENODELIST.	GaussDB: The nodelist will be released. Oracle Database: After the nodelist is released, it can still be queried in the original document.
DBMS_XMLDOM.GETATTRIBUTE	DBE_XMLDOM.GETATTRIBUTE	-
DBMS_XMLDOM.GETATTRIBUTES	DBE_XMLDOM.GETATTRIBUTES	-
DBMS_XMLDOM.GETCHILDNODES	DBE_XMLDOM.GETCHILDNODES	GaussDB: When the document node is used, DTD is included. Oracle Database: DTD is not included.
DBMS_XMLDOM.GETCHILDRENBYTAGNAME	DBE_XMLDOM.GETCHILDRENBYTAGNAME	GaussDB: The **ns** parameter of the DBE_XMLDOM.GETCHILDRENBYTAGNAME API does not support the asterisk (*) parameter. To obtain all attributes of a node, use the DBE_XMLDOM.GETCHILDNODES API. Oracle Database: The input parameter ***** is supported.
DBMS_XMLDOM.GETDOCUMENTELEMENT	DBE.XMLDOM.GETDOCUMENTELEMENT	-
DBMS_XMLDOM.GETFIRSTCHILD	DBE_XMLDOM.GETFIRSTCHILD	-
DBMS_XMLDOM.GETLASTCHILD	DBE_XMLDOM.GETLASTCHILD	-
DBMS_XMLDOM.GETLENGTH	DBE_XMLDOM.GETLENGTH	-
DBMS_XMLDOM.GETLOCALNAME	DBE_XMLDOM.GETLOCALNAME	-
DBMS_XMLDOM.GETNAMEDITEM	DBE_XMLDOM.GETNAMEDITEM	-
DBMS_XMLDOM.GETNEXTSIBLING	DBE_XMLDOM.GETNEXTSIBLING	-
DBMS_XMLDOM.GETNODENAME	DBE_XMLDOM.GETNODENAME	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_XMLDOM.GETNODETYPE	DBE_XMLDOM.GETNODETYPE	-
DBMS_XMLDOM.GETTAGNAME	DBE_XMLDOM.GETTAGNAME	-
DBMS_XMLDOM.IMPORTNODE	DBE_XMLDOM.IMPORTNODE	-
DBMS_XMLDOM.ISNULL	DBE_XMLDOM.ISNULL	GaussDB: When the input parameter is of the DOMNODELIST type, an error is reported if the object does not exist in the hash table. Oracle Database: No error is reported.
DBMS_XMLDOM.ITEM	DBE_XMLDOM.ITEM	-
DBMS_XMLDOM.MAKENODE	DBE_XMLDOM.MAKENODE	GaussDB: This function cannot be directly returned as the function return value. Oracle Database: It is directly returned as the function return value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_XMLDOM.NEWDOMDOCUMENT	DBE_XMLDOM.NEWDOMDOCUMENT	• GaussDB: The size of the input parameter must be less than 1 GB. Oracle Database: The size is the same as that of the CLOB type. • Currently, external DTD parsing is not supported in GaussDB. Oracle Database: External DTD can be parsed. • GaussDB: The default character set of doc created by newdomdocument is UTF-8. Oracle Database: It is generated based on the character set of the server. • GaussDB: Each doc parsed from the same xmltype instance is independent, and the modification of the doc does not affect the xmltype. Oracle Database: Each doc parsed from the same xmltype instance is not independent but associated. • GaussDB: The **version** column supports only **1.0**. If 1.0 to 1.9 are parsed, a warning is reported but the execution is normal. For versions later than 1.9, an error is reported. Oracle Database: No error is reported. • DTD validation difference: GaussDB reports an error for **! ATTLIST to type (CHECK|check|Check) "Ch..."** because the default value **"Ch..."** is not an enumerated value in the brackets. However, Oracle Database does not report this error. An error will be reported for **<!ENTITY baidu "www.baidu.com">..... &Baidu;&writer** because the letters are case-sensitive and **Baidu** cannot correspond to **baidu**. Oracle Database: No error is reported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
		• Namespace verification difference between GaussDB and Oracle Database: Undeclared namespace tags are parsed in GaussDB. Oracle Database: An error is reported.
DBMS_XMLDOM.SETATTRIBUTE	DBE_XMLDOM.SETATTRIBUTE	GaussDB: The attribute **key** cannot be null or an empty string. Oracle Database: The attribute **key** can be null or an empty string.
DBMS_XMLDOM.SETCHARSET	DBE_XMLDOM.SETCHARSET	Currently, the following character sets are supported in GaussDB: UTF-8, UCS-4, UCS-2, ISO-8859-1, ISO-8859-2, ISO-8859-3, ISO-8859-4, ISO-8859-5, ISO-8859-6, ISO-8859-7, ISO-8859-8, ISO-8859-9, ISO-2022-JP, Shift_JIS, EUC-JP, and ASCII. If you enter other character sets, an error is reported or garbled characters may be displayed.
DBMS_XMLDOM.SETDOCTYPE	DBE_XMLDOM.SETDOCTYPE	GaussDB: The total length of name, sysid, and pubid cannot exceed 32,500 bytes. Oracle Database: The maximum size is 32,767 bytes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_XMLDOM.WRITETOBUFFER	Supported, but with differences in DBE_XMLDOM.WRITETOBUFFER.	• GaussDB: The writetobuffer output buffer is limited to less than 1 GB. Oracle Database: The maximum size is 32,767 bytes. • GaussDB: The output doc will contain the XML declaration version and encoding. Oracle Database: It is not contained unless being specified by users. • GaussDB: If the input parameter is of the domnode type and the node is converted from a doc, the output node contains the XML declaration version and encoding. Oracle Database: It is not contained unless being specified by users. • GaussDB: By default, XML files are output in the UTF-8 character set. Oracle Database: It is generated based on the database character set.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_XMLDOM.WRITETOCLOB	DBE_XMLDOM.WRITETOCLOB	• GaussDB: The writetoclob size cannot exceed 1 GB. Oracle Database: The supported size depends on the CLOB size. • GaussDB: The output doc will contain the XML declaration version and encoding. Oracle Database: It is not contained unless being specified by users. • GaussDB: If the input parameter is of the domnode type and the node is converted from a doc, the output node contains the XML declaration version and encoding. Oracle Database: It is not contained unless being specified by users. • GaussDB: By default, XML files are output in the UTF-8 character set. Oracle Database: It is generated based on the database character set.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_XMLDOM.WRITETOFILE	DBE_XMLDOM.WRITETOFILE	• GaussDB: For an input parameter of the document type, the length of **filename** cannot exceed 255 bytes. For details about **charset**, see the dbe_xmldom.setcharset API. Oracle Database: The length of **filename** is affected by the OS and is greater than 255 bytes. • GaussDB: For an input parameter of the domnode type, the length of **filename** cannot exceed 255 bytes. For details about **charset**, see the dbe_xmldom.setcharset API. Oracle Database: The length of **filename** is affected by the OS and is greater than 255 bytes. • GaussDB: This function adds content such as indentation to format the output. The output doc will contain the XML declaration version and encoding. If the input parameter is of the domnode type and the node is converted from a doc, the output node contains the XML declaration version and encoding. Oracle Database: It is not contained unless being specified by users. • GaussDB: If newdomdocument() is used to create a doc without parameters, no error is reported when **charset** is not specified. The UTF-8 character set is used by default. Oracle Database: An error is reported. • GaussDB: The filename must be in the path created in pg_directory. The backslash (\) in the filename will be converted to a slash (/). Only one slash (/) is allowed. The file name must be in the pg_directory_name/file_name format.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
		Oracle Database: User input is not escaped.
DBMS_XMLDOM.GETNODEVALUE	DBE_XMLDOM.GETNODEVALUE	-
DBMS_XMLDOM.GETPARENTNODE	DBE_XMLDOM.GETPARENTNODE	-
DBMS_XMLDOM.HASCHILDNODES	DBE_XMLDOM.HASCHILDNODES	-
DBMS_XMLDOM.MAKEELEMENT	DBE_XMLDOM.MAKEELEMENT	-
DBMS_XMLDOM.SETNODEVALUE	DBE_XMLDOM.SETNODEVALUE	• GaussDB: The input parameter of **nodeValue** can be an empty string or **NULL**, but the node value will not be changed. Oracle Database: If you enter an empty string or **NULL**, **nodeValue** is set to an empty string. • GaussDB: The input parameter of **nodeValue** cannot include the escape character '&'. If the character string contains the escape character, **nodeValue** will be cleared. Oracle Database: Escape characters are supported.
DBMS_XMLDOM.GETELEMENTSBYTAGNAME	DBE_XMLDOM.GETELEMENTSBYTAGNAME	-

Table 3-123 DBMS_XMLPARSER compatibility

Oracle Database	GaussDB	Difference
DBMS_XMLPARSER.FREEPARSER	DBE_XMLPARSER.FREEPARSER	-
DBMS_XMLPARSER.GETDOCUMENT	DBE_XMLPARSER.GETDOCUMENT	-
DBMS_XMLPARSER.GETVALIDATIONMODE	DBE_XMLPARSER.GETVALIDATIONMODE	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_XMLPARSER.NEWPARSER	Supported, but with differences in DBE_XMLPARSER.NEWPARSER.	The maximum number of parser objects in GaussDB is 16,777,215, and that in Oracle Database is about 100 million.
DBMS_XMLPARSER.PARSEBUFFER	Supported, but with differences in DBE_XMLPARSER.PARSEBUFFER.	• Difference in parsing columns: Only UTF-8 is supported in terms of character encoding, and **version** can only be set to **1.0**. If versions 1.0 to 1.9 are parsed, a warning appears but the execution is normal. For versions later than 1.9, an error is reported. • Namespace validation difference: Undeclared namespace tags are parsed. However, Oracle Database reports an error. • Difference in parsing XML predefined entities: **'** and **"** are parsed and escaped to **'** and **"**. However, predefined entities in Oracle Database are not escaped to characters. • DTD validation differences: – An error will be reported for **! ATTLIST to type (CHECK|check|Check) "Ch..."** because the default value **"Ch..."** is not an enumerated value in the brackets. However, Oracle Database does not report this error. – An error will be reported for **<!ENTITY baidu "www.baidu.com">..... &Baidu;&writer**, because the letters are case-sensitive and **Baidu** cannot correspond to **baidu**. However, Oracle Database does not report this error.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_XMLPARSER.PARSECLOB	Supported, but with differences in DBE_XMLPARSER.PARSECLOB.	• PARSECLOB cannot parse CLOBs greater than or equal to 2 GB. • Difference in parsing columns: Only UTF-8 is supported in terms of character encoding, and **version** can only be set to **1.0**. If versions 1.0 to 1.9 are parsed, a warning appears but the execution is normal. For versions later than 1.9, an error is reported. • Namespace validation difference: Undeclared namespace tags are parsed. However, Oracle Database reports an error. • Difference in parsing XML predefined entities: **'** and **"** are parsed and escaped to **'** and **"**. However, predefined entities in Oracle Database are not escaped to characters. • DTD validation differences: – An error will be reported for **! ATTLIST to type (CHECK|check|Check) "Ch..."** because the default value **"Ch..."** is not an enumerated value in the brackets. However, Oracle Database does not report this error. – An error will be reported for **<!ENTITY baidu "www.baidu.com">..... &Baidu;&writer**, because the letters are case-sensitive and **Baidu** cannot correspond to **baidu**. However, Oracle Database does not report this error.
DBMS_XMLPARSER.SETVALIDATIONMODE	DBE_XMLPARSER.SETVALIDATIONMODE	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-124 DBMS_ILM compatibility

Oracle Database	GaussDB	Difference
DBMS_ILM.ADD_TO_ILM	Not supported.	-
DBMS_ILM.ARCHIVESTATENAME	Not supported.	-
DBMS_ILM.EXECUTE_ILM	DBE_ILM.EXECUTE_ILM	• The input parameter **Schema** in GaussDB corresponds to **owner** in Oracle Database. • GaussDB does not support the operation of specifying **ilm_scope** (specifying multiple objects at a time).
DBMS_ILM.EXECUTE_ILM_TASK	Not supported.	-
DBMS_ILM.PREVIEW_ILM	Not supported.	-
DBMS_ILM.REMOVE_FROM_ILM	Not supported.	-
DBMS_ILM.STOP_ILM	DBE_ILM.STOP_ILM	-

Table 3-125 DBMS_ILM_ADMIN compatibility

Oracle Database	GaussDB	Difference
DBMS_ILM_ADMIN.CLEAR_HEAT_MAP_ALL	Not supported.	-
DBMS_ILM_ADMIN.CLEAR_HEAT_MAP_TABLE	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_ILM_ADMIN.CUSTOMIZE_ILM	DBE_ILM_ADMIN.CUSTOMIZE_ILM	The feature parameters corresponding to the values of input parameters are different. • The value of **param** in GaussDB can be **1**, **2**, **7**, **11**, **12**, **13**, **14**, or **15**. • When the value of **param** in GaussDB is **14**, the corresponding feature parameter is **WIND_DURATION** which is used to control the duration of the execution window in automatic scheduling. However, Oracle Database does not have the corresponding feature parameter.
DBMS_ILM_ADMIN.DISABLE_ILM	DBE_ILM_ADMIN.DISABLE_ILM	-
DBMS_ILM_ADMIN.ENABLE_AUTO_OPTIMIZE	Not supported.	-
DBMS_ILM_ADMIN.ENABLE_ILM	DBE_ILM_ADMIN.ENABLE_ILM	-
DBMS_ILM_ADMIN. IGNORE_AUTO_OPTIMIZE_ CRITERIA	Not supported.	-
DBMS_ILM_ADMIN.SET_HEAT_MAP_ALL	Not supported.	-
DBMS_ILM_ADMIN.SET_HEAT_MAP_START	Not supported.	-
DBMS_ILM_ADMIN.SET_HEAT_MAP_TABLE	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-126 DBMS_COMPRESSION compatibility

Oracle Database	GaussDB	Difference
DBMS_COMPRESSION.GET_COMPRESSION_RATIO	DBE_COMPRESSION.GET_COMPRESSION_RATIO	• GaussDB cannot obtain compression ratios of LOBs. • For obtaining a compression ratio of a single object: – The value of the input parameter **comptype** in GaussDB can only be **1** (uncompressed) or **2** (advanced compression) while Oracle Database also supports values such as **1024** and **2048**. – The value of the input parameter **objtype** in GaussDB can only be **1** (table object) while Oracle Database also supports the value **2** (index object). – Oracle Database uses the **subset_numrows** parameter to directly determine the number of rows to be sampled (that is, the value of the parameter). GaussDB uses **sample_ratio** (sampling rate) to indirectly determine the number of rows to be sampled.
DBMS_COMPRESSION.GET_COMPRESSION_TYPE	DBE_COMPRESSION.GET_COMPRESSION_TYPE	• Oracle Database uses a row ID to specify the row whose compression type is to be obtained, while GaussDB uses a CTID to specify the row. • The value of **comptype** is returned. The value difference is the same as that of GET_COMPRESSION_RATIO. • In GaussDB, this API can be called only on DNs. For details, see **DBE_COMPRESSION**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-127 DBMS_HEAT_MAP compatibility

Oracle Database	GaussDB	Difference
DBMS_HEAT_MAP.BLOCK_HEAT_MAP	Not supported.	-
DBMS_HEAT_MAP.EXTENT_HEAT_MAP	Not supported.	-
DBMS_HEAT_MAP.OBJECT_HEAT_MAP	Not supported.	-
DBMS_HEAT_MAP.SEGMENT_HEAT_MAP	Not supported.	-
DBMS_HEAT_MAP.TABLESPACE_HEAT_MAP	Not supported.	-
Not supported.	DBE_HEAT_MAP.ROW_HEAT_MAP	In GaussDB, this API can be called only on DNs. For details, see **DBE_HEAT_MAP**.

Table 3-128 DBMS_DESCRIBE compatibility

Oracle Database	GaussDB	Difference
DBMS_DESCRIBE.DESCRIBE_PROCEDURE	DBE_DESCRIBE.DESCRIBE_PROCEDURE	• The **datatype** parameter is different from Oracle Database. GaussDB returns the OID of the data type, and Oracle Database returns the ID of the data type within Oracle Database. • The **datalength**, **dataprecision**, and **scale** parameters are set to **0** because type constraints (such as number (7,2) and varchar2(20)) cannot be retained when GaussDB creates stored procedures or functions. Oracle Database can use the %type method to obtain constrained data types. • For details about the usage in GaussDB, see **DBE_DESCRIBE**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-129 DBMS_STATS compatibility

Oracle Database	GaussDB	Difference
DBMS_STATS.ALTER_STATS_HISTORY_RETENTION	Not supported.	-
DBMS_STATS.CANCEL_ADVISOR_TASK	Not supported.	-
DBMS_STATS.CONFIGURE_ADVISOR_FILTER	Not supported.	-
DBMS_STATS.CONFIGURE_ADVISOR_OBJ_FILTER	Not supported.	-
DBMS_STATS.CONFIGURE_ADVISOR_OPR_FILTER	Not supported.	-
DBMS_STATS.CONFIGURE_ADVISOR_RULE_FILTER	Not supported.	-
DBMS_STATS.CREATE_ADVISOR_TASK	Not supported.	-
DBMS_STATS.CONVERT_RAW_VALUE	Not supported.	-
DBMS_STATS.CONVERT_RAW_VALUE_NVARCHAR	Not supported.	-
DBMS_STATS.CONVERT_RAW_VALUE_ROWID	Not supported.	-
DBMS_STATS.COPY_TABLE_STATS	Not supported.	-
DBMS_STATS.CREATE_EXTENDED_STATS	Not supported.	-
DBMS_STATS.CREATE_STAT_TABLE	DBE_STATS.CREATE_STAT_TABLE	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**.
DBMS_STATS.DELETE_COLUMN_STATS	Not supported.	-
DBMS_STATS.DELETE_DATABASE_PREFS	Not supported.	-
DBMS_STATS.DELETE_DATABASE_STATS	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.DELETE_DICTIONARY_STATS	Not supported.	-
DBMS_STATS.DELETE_FIXED_OBJECTS_STATS	Not supported.	-
DBMS_STATS.DELETE_INDEX_STATS	Not supported.	-
DBMS_STATS.DELETE_PENDING_STATS	Not supported.	-
DBMS_STATS.DELETE_PROCESSING_RATE	Not supported.	-
DBMS_STATS.DELETE_SCHEMA_PREFS	Not supported.	-
DBMS_STATS.DELETE_SCHEMA_STATS	Not supported.	-
DBMS_STATS.DELETE_SYSTEM_STATS	Not supported.	-
DBMS_STATS.DELETE_TABLE_PREFS	Not supported.	-
DBMS_STATS.DELETE_TABLE_STATS	Not supported.	-
DBMS_STATS.DIFF_TABLE_STATS_IN_HISTORY	Not supported.	-
DBMS_STATS.DIFF_TABLE_STATS_IN_PENDING	Not supported.	-
DBMS_STATS.DIFF_TABLE_STATS_IN_STATTAB	Not supported.	-
DBMS_STATS.DROP_ADVISOR_TASK	Not supported.	-
DBMS_STATS.DROP_EXTENDED_STATS	Not supported.	-
DBMS_STATS.DROP_STAT_TABLE	DBE_STATS.DROP_STAT_TABLE	-
DBMS_STATS.EXECUTE_ADVISOR_TASK	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.EXPORT_COLUMN_STATS	DBE_STATS.EXPORT_COLUMN_STATS	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**. • The exported column-level statistics are consistent with those in the pg_statistic catalog. Multiple columns are consistent with those in the pg_statistic_ext catalog. • Index expression statistics can be exported. **tabname** must be set to an index name, and **colname** must be set to an index expression name. • Permission: You must have the ANALYZE permission to query tables and the siud permission on the **stattab** table.
DBMS_STATS.EXPORT_DATABASE_PREFS	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.EXPORT_DATABASE_STATS	DBE_STATS.EXPORT_DATABASE_STATS	• In GaussDB, **statown** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**. • In the **stattab** table, the exported table-level and partition-level statistics are **numrows**, **numblocks**, and **relallvisible**, which correspond to **reltuples**, **relpages**, and **relallvisible** in the pg_class and pg_partition system catalogs, respectively. • Permission: You must have the ANALYZE permission to query tables and the siud permission on the **stattab** table.
DBMS_STATS.EXPORT_DICTIONARY_STATS	Not supported.	-
DBMS_STATS.EXPORT_FIXED_OBJECTS_STATS	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.EXPORT_INDEX_STATS	DBE_STATS.EXPORT_INDEX_STATS	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**. • In the **stattab** table, the exported table-level and partition-level statistics are **numrows**, **numblocks**, and **relallvisible**, which correspond to **reltuples**, **relpages**, and **relallvisible** in the pg_class and pg_partition system catalogs, respectively. • Permission: You must have the ANALYZE permission to query tables and the siud permission on the **stattab** table.
DBMS_STATS.EXPORT_PENDING_STATS	Not supported.	-
DBMS_STATS.EXPORT_SCHEMA_PREFS	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.EXPORT_SCHEMA_STATS	DBE_STATS.EXPORT_SCHEMA_STATS	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**. • In the **stattab** table, the exported table-level and partition-level statistics are **numrows**, **numblocks**, and **relallvisible**, which correspond to **reltuples**, **relpages**, and **relallvisible** in the pg_class and pg_partition system catalogs, respectively. The column-level statistics of the exported table are consistent with those of the pg_statistic and pg_statistic_ext catalogs. • Permission: You must have the siud permission on the **stattab** table.
DBMS_STATS.EXPORT_SYSTEM_STATS	Not supported.	-
DBMS_STATS.EXPORT_TABLE_PREFS	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.EXPORT_TABLE_STATS	DBE_STATS.EXPORT_TABLE_STATS	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**. • In the **stattab** table, the exported table-level and partition-level statistics are **numrows**, **numblocks**, and **relallvisible**, which correspond to **reltuples**, **relpages**, and **relallvisible** in the pg_class and pg_partition system catalogs, respectively. The column-level statistics exported in cascading mode are consistent with those in the pg_statistic and pg_statistic_ext catalogs. • Permission: You must have the ANALYZE permission to query tables and the siud permission on the **stattab** table.
DBMS_STATS.FLUSH_DATABASE_MONITORING_INFO	Not supported.	-
DBMS_STATS.GATHER_DATABASE_STATS	Not supported.	-
DBMS_STATS.GATHER_DICTIONARY_STATS	Not supported.	-
DBMS_STATS.GATHER_FIXED_OBJECTS_STATS	Not supported.	-
DBMS_STATS.GATHER_INDEX_STATS	Not supported.	-
DBMS_STATS.GATHER_PROCESSING_RATE	Not supported.	-
DBMS_STATS.GATHER_SCHEMA_STATS	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.GATHER_SYSTEM_STATS	Not supported.	-
DBMS_STATS.GATHER_TABLE_STATS	Not supported.	-
DBMS_STATS.GENERATE_STATS	Not supported.	-
DBMS_STATS.GET_ADVISOR_OPR_FILTER	Not supported.	-
DBMS_STATS.GET_ADVISOR_RECS	Not supported.	-
DBMS_STATS.GET_COLUMN_STATS	Not supported.	-
DBMS_STATS.GET_INDEX_STATS	Not supported.	-
DBMS_STATS.GET_PARAM	Not supported.	-
DBMS_STATS.GET_PREFS	Not supported.	-
DBMS_STATS.GET_STATS_HISTORY_AVAILABILITY	DBE_STATS.GET_STATS_HISTORY_AVAILABILITY	GaussDB queries the collection time of the earliest historical statistics in the entire database.
DBMS_STATS.GET_STATS_HISTORY_RETENTION	DBE_STATS.GET_STATS_HISTORY_RETENTION	-
DBMS_STATS.GET_SYSTEM_STATS	Not supported.	-
DBMS_STATS.GET_TABLE_STATS	Not supported.	-
DBMS_STATS.IMPLEMENT_ADVISOR_TASK	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.IMPORT_COLUMN_STATS	DBE_STATS.IMPORT_COLUMN_STATS	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**. • The exported single-column **col** statistics are the same as those in the pg_statistic catalog. The exported multi-column **ext-col** statistics are consistent with those in the pg_statistic_ext catalog. • Index expression statistics can be imported. **tabname** must be set to an index name, and **colname** must be set to an index expression name. • Permission: You must have the ANALYZE permission to query tables and the siud permission on the **stattab** table.
DBMS_STATS.IMPORT_DATABASE_PREFS	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.IMPORT_DATABASE_STATS	DBE_STATS.IMPORT_DATABASE_STATS	• In GaussDB, **statown** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**. • In the **stattab** table, the imported table-level and partition-level statistics are **numrows**, **numblocks**, and **relallvisible**, which correspond to **reltuples**, **relpages**, and **relallvisible** in the pg_class and pg_partition system catalogs, respectively.
DBMS_STATS.IMPORT_DICTIONARY_STATS	Not supported.	-
DBMS_STATS.IMPORT_FIXED_OBJECTS_STATS	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.IMPORT_INDEX_STATS	DBE_STATS.IMPORT_INDEX_STATS	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**. • In the **stattab** table, the imported table-level and partition-level statistics are **numrows**, **numblocks**, and **relallvisible**, which correspond to **reltuples**, **relpages**, and **relallvisible** in the pg_class and pg_partition system catalogs, respectively. • Permission: You must have the ANALYZE permission to query tables and the siud permission on the **stattab** table.
DBMS_STATS.IMPORT_SCHEMA_PREFS	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.IMPORT_SCHEMA_STATS	DBE_STATS.IMPORT_SCHEMA_STATS	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**. • In the **stattab** table, the imported table-level and partition-level statistics are **numrows**, **numblocks**, and **relallvisible**, which correspond to **reltuples**, **relpages**, and **relallvisible** in the pg_class and pg_partition system catalogs, respectively. The column-level statistics of the imported table are consistent with those of the pg_statistic and pg_statistic_ext catalogs. • Permission: You must have the siud permission on the **stattab** table.
DBMS_STATS.IMPORT_SYSTEM_STATS	Not supported.	-
DBMS_STATS.IMPORT_TABLE_PREFS	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.IMPORT_TABLE_STATS	DBE_STATS.IMPORT_TABLE_STATS	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**. • In the **stattab** table, the imported table-level and partition-level statistics are **numrows**, **numblocks**, and **relallvisible**, which correspond to **reltuples**, **relpages**, and **relallvisible** in the pg_class and pg_partition system catalogs, respectively. The column-level statistics imported in cascading mode are consistent with those in the pg_statistic and pg_statistic_ext catalogs. • Permission: You must have the ANALYZE permission to query tables and the siud permission on the **stattab** table.
DBMS_STATS.INTERRUPT_ADVISOR_TASK	Not supported.	-
DBMS_STATS.LOCK_PARTITION_STATS	DBE_STATS.LOCK_PARTITION_STATS	• In GaussDB, **ownname** should be set to a **Schema** name.
DBMS_STATS.LOCK_SCHEMA_STATS	DBE_STATS.LOCK_SCHEMA_STATS	• In GaussDB, **ownname** should be set to a **Schema** name.
DBMS_STATS.LOCK_TABLE_STATS	DBE_STATS.LOCK_TABLE_STATS	• In GaussDB, **ownname** should be set to a **Schema** name.
DBMS_STATS.MERGE_COL_USAGE	Not supported.	-
DBMS_STATS.PREPARE_COLUMN_VALUES	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.PREPARE_COLUMN_VALUES_ROWID	Not supported.	-
DBMS_STATS.PUBLISH_PENDING_STATS	Not supported.	-
DBMS_STATS.PURGE_STATS	DBE_STATS.PURGE_STATS	-
DBMS_STATS.REMAP_STAT_TABLE	Not supported.	-
DBMS_STATS.REPORT_ADVISOR_TASK	Not supported.	-
DBMS_STATS.REPORT_COL_USAGE	Not supported.	-
DBMS_STATS.REPORT_GATHER_AUTO_STATS	Not supported.	-
DBMS_STATS.REPORT_GATHER_DATABASE_STATS	Not supported.	-
DBMS_STATS.REPORT_GATHER_DICTIONARY_STATS	Not supported.	-
DBMS_STATS.REPORT_GATHER_FIXED_OBJ_STATS	Not supported.	-
DBMS_STATS.REPORT_GATHER_SCHEMA_STATS	Not supported.	-
DBMS_STATS.REPORT_STATS_OPERATIONS	Not supported.	-
DBMS_STATS.RESET_ADVISOR_TASK	Not supported.	-
DBMS_STATS.RESET_COL_USAGE	Not supported.	-
DBMS_STATS.RESET_GLOBAL_PREF_DEFAULTS	Not supported.	-
DBMS_STATS.RESET_PARAM_DEFAULTS	Not supported.	-
DBMS_STATS.RESTORE_DICTIONARY_STATS	Not supported.	-
DBMS_STATS.RESTORE_FIXED_OBJECTS_STATS	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.RESTORE_SCHEMA_STATS	DBE_STATS.RESTORE_SCHEMA_STATS	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**.
DBMS_STATS.RESTORE_SYSTEM_STATS	Not supported.	-
DBMS_STATS.RESTORE_TABLE_STATS	DBE_STATS.RESTORE_TABLE_STATS	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**.
DBMS_STATS.RESUME_ADVISOR_TASK	Not supported.	-
DBMS_STATS.SCRIPT_ADVISOR_TASK	Not supported.	-
DBMS_STATS.SEED_COL_USAGE	Not supported.	-
DBMS_STATS.SET_ADVISOR_TASK_PARAMETER	Not supported.	-
DBMS_STATS.SET_COLUMN_STATS	DBE_STATS.SET_COLUMN_STATS	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**.
DBMS_STATS.SET_DATABASE_PREFS	Not supported.	-
DBMS_STATS.SET_GLOBAL_PREFS	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.SET_INDEX_STATS	DBE_STATS.SET_INDEX_STATS	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**. • The **relallvisible** input parameter is added to GaussDB.
DBMS_STATS.SET_PARAM	Not supported.	-
DBMS_STATS.SET_PROCESSING_RATE	Not supported.	-
DBMS_STATS.SET_SCHEMA_PREFS	Not supported.	-
DBMS_STATS.SET_SYSTEM_STATS	Not supported.	-
DBMS_STATS.SET_TABLE_PREFS	Not supported.	-
DBMS_STATS.SET_TABLE_STATS	DBE_STATS.SET_TABLE_STATS	• In GaussDB, **ownname** should be set to a **Schema** name. • GaussDB supports only some input parameters. For details, see **DBE_STATS**. • The **relallvisible** input parameter is added to GaussDB.
DBMS_STATS.SHOW_EXTENDED_STATS_NAME	Not supported.	-
DBMS_STATS.TRANSFER_STATS	Not supported.	-
DBMS_STATS.UNLOCK_PARTITION_STATS	DBE_STATS.UNLOCK_PARTITION_STATS	In GaussDB, **ownname** should be set to a **Schema** name.
DBMS_STATS.UNLOCK_SCHEMA_STATS	DBE_STATS.UNLOCK_SCHEMA_STATS	In GaussDB, **ownname** should be set to a **Schema** name.
DBMS_STATS.UNLOCK_TABLE_STATS	DBE_STATS.UNLOCK_TABLE_STATS	In GaussDB, **ownname** should be set to a **Schema** name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_STATS.UPGRADE_STAT_TABLE	Not supported.	-

Table 3-130 DBMS_XMLGEN compatibility

Oracle Database	GaussDB	Difference
DBMS_XMLGEN.CONVERT	DBE_XMLGEN.CONVERT	-
DBMS_XMLGEN.NEWCONTEXT	DBE_XMLGEN.NEWCONTEXT	-
DBMS_XMLGEN.NEWCONTEXTFROMHIERARCHY	DBE_XMLGEN.NEWCONTEXTFROMHIERARCHY	• The maximum depth of recursive XML files generated by GaussDB cannot exceed 50 million layers. • XML files generated by the CONNECT BY statement in Oracle Database's newcontextfromhierarchy method contain XML headers. However, the directly constructed data does not contain the XML header. In GaussDB, the files contain XML headers.
DBMS_XMLGEN.SETCONVERTSPECIALCHARS	DBE_XMLGEN.SETCONVERTSPECIALCHARS	-
DBMS_XMLGEN.SETNULLHANDLING	DBE_XMLGEN.SETNULLHANDLING	-
DBMS_XMLGEN.SETROWSETTAG	DBE_XMLGEN.SETROWSETTAG	-
DBMS_XMLGEN.SETROWTAG	DBE_XMLGEN.SETROWTAG	-
DBMS_XMLGEN.USENULLATTRIBUTEINDICATOR	DBE_XMLGEN.USENULLATTRIBUTEINDICATOR	-
DBMS_XMLGEN.USEITEMTAGSFORCOLL	DBE_XMLGEN.USEITEMTAGSFORCOLL	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Oracle Database	GaussDB	Difference
DBMS_XMLGEN.GETNUMROWSPROCESSED	DBE_XMLGEN.GETNUMROWSPROCESSED	-
DBMS_XMLGEN.SETMAXROWS	DBE_XMLGEN.SETMAXROWS	-
DBMS_XMLGEN.SETSKIPROWS	DBE_XMLGEN.SETSKIPROWS	-
DBMS_XMLGEN.RESTARTQUERY	DBE_XMLGEN.RESTARTQUERY	In distributed GaussDB, the cursor cannot be moved reversely. Therefore, the restartquery function is unavailable.
DBMS_XMLGEN.GETXMLTYPE	DBE_XMLGEN.GETXMLTYPE	-
DBMS_XMLGEN.GETXML	DBE_XMLGEN.GETXML	-
DBMS_XMLGEN.CLOSECONTEXT	DBE_XMLGEN.CLOSECONTEXT	-

Table 3-131 DBMS_OBFUSCATION_TOOLKIT compatibility

Oracle Database	GaussDB	Difference
DBMS_OBFUSCATION_TOOLKIT.DESGETKEY	DBE_OBFUSCATION_TOOLKIT.DESGETKEY	-
DBMS_OBFUSCATION_TOOLKIT.DESENCRYPT	DBE_OBFUSCATION_TOOLKIT.DESENCRYPT	-
DBMS_OBFUSCATION_TOOLKIT.DESDECRYPT	DBE_OBFUSCATION_TOOLKIT.DESDECRYPT	-
DBMS_OBFUSCATION_TOOLKIT.DES3GETKEY	DBE_OBFUSCATION_TOOLKIT.DES3GETKEY	-
DBMS_OBFUSCATION_TOOLKIT.DES3ENCRYPT	DBE_OBFUSCATION_TOOLKIT.DES3ENCRYPT	-
DBMS_OBFUSCATION_TOOLKIT.DES3DECRYPT	DBE_OBFUSCATION_TOOLKIT.DES3DECRYPT	-
DBMS_OBFUSCATION_TOOLKIT.MD5	DBE_OBFUSCATION_TOOLKIT.MD5	-

3.3 MySQL Compatibility Description

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.1 Overview of MySQL Compatibility

Overview of M-compatible Mode

M-compatible Mode compares GaussDB in M-compatible mode (**sql_compatibility** set to **'M'**) with MySQL 5.7. Only compatibility features added later than GaussDB 505.2.0 are described. You are advised to view the specifications and restrictions of the features in *M Compatibility Developer Guide*.

GaussDB is compatible with MySQL in terms of data types, SQL functionalities, and database objects.

The execution plans, optimization, and EXPLAIN results in GaussDB are different from those in MySQL.

Due to variances in the underlying framework implementations, there are still some differences between GaussDB and MySQL.

NOTE

The underlying architecture of GaussDB is different from that of MySQL. Therefore, the performance of querying the same schemas under information_schema and m_schema may be different from that in MySQL. For details, see "Schemas" in *M Compatibility Developer Guide*. For example, the execution of the count function cannot be optimized. The time consumed by the SELECT * and SELECT COUNT(*) statements is similar.

Overview of MySQL-compatible Mode

MYSQL-compatible Mode compares GaussDB in MySQL-compatible mode (that is, when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**) with MySQL 5.7. Only the compatibility features added after version 503.0.0 are described.

NOTE

- The MySQL-compatible mode delivers high compatibility with MySQL in terms of syntax, data types, metadata, and protocols. The MySQL-compatible mode will not evolve because it is not compatible with the MySQL architecture.
- Only the features described in **MYSQL-compatible Mode** are compatible with MySQL. The behavior of other features remain the same as that in GaussDB.
- The implementation logic of the MySQL-compatible mode (**sql_compatibility** set to **'MYSQL'**) is similar to that of the B-compatible mode (**sql_compatibility** set to **'B'**) in centralized deployment.

GaussDB is compatible with MySQL in terms of data types, SQL functionalities, and database objects.

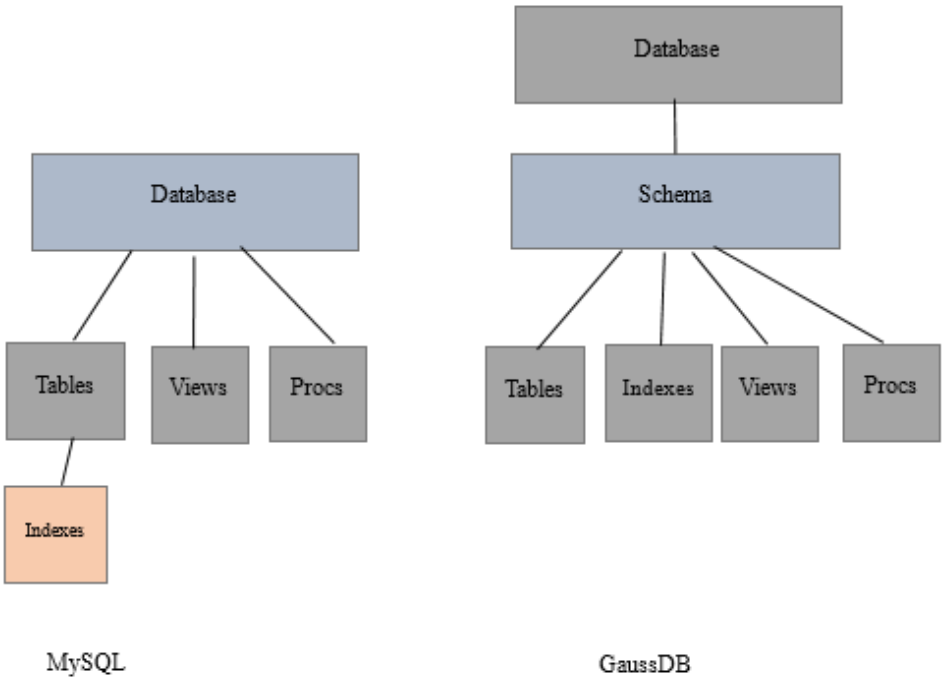
Due to variances in the underlying framework implementations, there are still some differences between GaussDB and MySQL.

Database and Schema Design

MySQL data objects include database, table, index, view, trigger, and proc. From top to bottom, each of them may correspond to multiple objects in GaussDB, as shown in **Figure 3-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Figure 3-1 Differences between databases and schemas in MySQL and GaussDB



- In MySQL, database and schema are synonyms. In GaussDB, a database can have multiple schemas. In this feature, each database in MySQL is mapped to a schema in GaussDB.
- In MySQL, an index belongs to a table. In GaussDB, an index belongs to a schema. As a result, an index name must be unique in a schema in GaussDB and must be unique in a table in MySQL. This difference will be retained as a current constraint.

3.3.2 M-compatible Mode

3.3.2.1 Data Types

3.3.2.1.1 Numeric Data Types

Unless otherwise specified, the precision, scale, and number of digits cannot be set to floating-point values in M-compatible mode by default. You are advised to use valid integer values.

Table 3-132 Integer types

Data Type	Differences Compared with MySQL
BOOL	Output format: The output of SELECT TRUE/FALSE in GaussDB is **t** or **f**, and that in MySQL is **1** or **0**. MySQL: The BOOL/BOOLEAN type is actually mapped to the TINYINT type.
BOOLEAN	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Data Type	Differences Compared with MySQL
TINYINT[(M)] [UNSIGNED] [ZEROFILL] SMALLINT[(M)] [UNSIGNED] [ZEROFILL]	Input format: • MySQL: If a character string with multiple decimal points (such as "1.2.3.4.5") is entered, MySQL will misparse the character string in loose mode, throw a warning, but insert the character string into the table successfully. For example, after "1.2.3.4.5" is inserted into the table, the value is **12**. • GaussDB: If a character string with multiple decimal points (such as "1.2.3.4.5") is entered in loose mode, the characters after the second decimal point are truncated as invalid characters, a warning is thrown, but the character string is inserted into the table successfully. For example, after "1.2.3.4.5" is inserted into the table, the value is **1**. After "1.6.3.4.5" is inserted into the table, the value is **2**.
MEDIUMINT[(M)] [UNSIGNED] [ZEROFILL]	MySQL requires 3 bytes to store MEDIUMINT data. • The signed range is –8388608 to +8388607. • The unsigned range is 0 to +16777215. GaussDB is mapped to the INT type. Four bytes are required for storage. The value range is determined based on boundary values. • The signed range is –8388608 to +8388607. • The unsigned range is 0 to +16777215. For other differences, see the description below the table.
INT[(M)] [UNSIGNED] [ZEROFILL] INTEGER[(M)] [UNSIGNED] [ZEROFILL] BIGINT[(M)] [UNSIGNED] [ZEROFILL]	Input format: • MySQL: If a character string with multiple decimal points (such as "1.2.3.4.5") is entered, MySQL will misparse the character string in loose mode, throw a warning, but insert the character string into the table successfully. For example, after "1.2.3.4.5" is inserted into the table, the value is **12**. • GaussDB: If a character string with multiple decimal points (such as "1.2.3.4.5") is entered in loose mode, the characters after the second decimal point are truncated as invalid characters, a warning is thrown, but the character string is inserted into the table successfully. For example, after "1.2.3.4.5" is inserted into the table, the value is **1**. After "1.6.3.4.5" is inserted into the table, the value is **2**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-133 Arbitrary precision types

Data Type	Differences Compared with MySQL
DECIMAL[(M[,D])] [ZEROFILL]	-
NUMERIC[(M[,D])] [ZEROFILL]	
DEC[(M[,D])] [ZEROFILL]	
FIXED[(M[,D])] [ZEROFILL]	

Table 3-134 Floating-point types

Data Type	Differences Compared with MySQL
FLOAT[(M,D)] [ZEROFILL]	The FLOAT data type does not support partitioned tables with the key partitioning policy.
FLOAT(p) [ZEROFILL]	The FLOAT data type does not support partitioned tables with the key partitioning policy.
DOUBLE[(M,D)] [ZEROFILL]	The DOUBLE data type does not support partitioned tables with the key partitioning policy.
DOUBLE PRECISION[(M,D)] [ZEROFILL]	The DOUBLE PRECISION data type does not support partitioned tables with the key partitioning policy.
REAL[(M,D)] [ZEROFILL]	The REAL data type does not support partitioned tables with the key partitioning policy.

3.3.2.1.2 Date and Time Data Types

Table 3-135 Date and Time Data Types

Data Type	Differences Compared with MySQL
DATE	GaussDB supports the DATE data type and differs from MySQL in terms of the following specifications: A backslash (\) is regarded as an escape character in both MySQL and GaussDB. However, MySQL supports \0, but GaussDB does not support \0. Therefore, GaussDB reports an error when the backslash is used as a separator and followed by 0.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Data Type	Differences Compared with MySQL
DATETIME[(fsp)]	GaussDB supports the DATETIME data type and differs from MySQL in terms of the following specifications: A backslash (\) is regarded as an escape character in both MySQL and GaussDB. However, MySQL supports \0, but GaussDB does not support \0. Therefore, GaussDB reports an error when the backslash is used as a separator and followed by 0.
TIMESTAMP[(fsp)]	GaussDB supports the TIMESTAMP data type and differs from MySQL in terms of the following specifications: ● A backslash (\) is regarded as an escape character in both MySQL and GaussDB. However, MySQL supports \0, but GaussDB does not support \0. Therefore, GaussDB reports an error when the backslash is used as a separator and followed by 0. ● MySQL supports **explicit_defaults_for_timestamp**. When **explicit_defaults_for_timestamp** is set to **off**, setting the default value of the TIMESTAMP column and inserting **NULL** are non-standard behaviors. The default value of **explicit_defaults_for_timestamp** is **off** in MySQL 5.7 and **on** in MySQL 8.0. GaussDB does not support setting **explicit_defaults_for_timestamp**. Its behavior is equivalent to MySQL with **explicit_defaults_for_timestamp** set to **on**. For details about **explicit_defaults_for_timestamp**, see the note below the table.
TIME[(fsp)]	GaussDB supports the TIME data type and differs from MySQL in terms of the following specifications: ● A backslash (\) is regarded as an escape character in both MySQL and GaussDB. However, MySQL supports \0, but GaussDB does not support \0. Therefore, GaussDB reports an error when the backslash is used as a separator and followed by 0. ● When the hour, minute, second, and nanosecond of the TIME type are 0, the sign bits of GaussDB and MySQL may be different.
YEAR[(4)]	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- GaussDB does not support ODBC syntax literals:

  { d 'str' }

  { t 'str' }

  { ts 'str' }
- If you specify precision for the DATETIME, TIME, or TIMESTAMP data type greater than the maximum precision supported by the data type, GaussDB truncates the precision to the maximum precision supported by the data type, whereas MySQL reports an error.
- In MySQL, when **explicit_defaults_for_timestamp** is set to **off**, the processing logic for TIMESTAMP columns is as follows:
  - If the **NULL** or **NOT NULL** attribute is not explicitly specified for a column, the **NOT NULL** attribute will be automatically added. When the **NULL** value is inserted into such a column, the **NULL** value will be replaced with the current timestamp.
  - If the **NULL** attribute is not specified for the first TIMESTAMP column in a table, the **DEFAULT CURRENT_TIMESTAMP** and **ON UPDATE CURRENT_TIMESTAMP** attributes will be automatically added to the column.
  - If the **NULL** attribute is not specified for the second and subsequent TIMESTAMP columns in a table, the **DEFAULT '0000-00-00 00:00:00'** attribute will be automatically added to the columns.
- In MySQL, when **explicit_defaults_for_timestamp** is set to **on**, the processing logic for TIMESTAMP columns is as follows:
  - When the **NULL** value is inserted into a TIMESTAMP column, the **NULL** value will not be replaced with the current timestamp.
  - If the **NULL** or **NOT NULL** attribute is not explicitly specified for a column, the **NULL** attribute will be automatically added.
  - When the **NULL** value is inserted into a column with the **NOT NULL** attribute specified, an error will be reported in strict mode, and **'0000-00-00 00:00:00'** will be inserted in loose mode.
  - The **DEFAULT CURRENT_TIMESTAMP** and **ON UPDATE CURRENT_TIMESTAMP** attributes will not be automatically added to any TIMESTAMP columns.

3.3.2.1.3 String Data Types

Table 3-136 String data types

Data Type	Differences Compared with MySQL
CHAR(M)	Input format: If a binary or hexadecimal character string is input, GaussDB outputs a hexadecimal character string, and MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Data Type	Differences Compared with MySQL
VARCHAR(M)	Input format: • GaussDB cannot verify the length of parameters and return values of user-defined functions or the length of stored procedure parameters. However, MySQL supports their verification. • GaussDB can verify the length of temporary variables in user-defined functions and stored procedures, and an error or truncation alarm is reported in strict or loose mode. However, MySQL does not support these functions. • After a binary or hexadecimal character string is entered, GaussDB outputs a hexadecimal character string, and MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty.
TINYTEXT	• Input format: If a binary or hexadecimal character string is input, GaussDB outputs a hexadecimal character string, and MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty. • Default value: The default value cannot be set in MySQL 5.7 but can be set in GaussDB and MySQL 8.0. • Primary key: When creating a primary key, you must specify the prefix length in MySQL, but you cannot specify the prefix length in GaussDB. • Index: In MySQL, the TINYTEXT type does not support other index methods except prefix indexes. GaussDB supports these index methods.
TEXT	• Input format: If a binary or hexadecimal character string is input, GaussDB outputs a hexadecimal character string, and MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty. • Default value: The default value cannot be set in MySQL 5.7 but can be set in GaussDB and MySQL 8.0. • Primary key: When creating a primary key, you must specify the prefix length in MySQL, but you cannot specify the prefix length in GaussDB. • Index: In MySQL, the TEXT type does not support other index methods except prefix indexes. GaussDB supports these index methods.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Data Type	Differences Compared with MySQL
MEDIUMTEXT	• Input format: If a binary or hexadecimal character string is input, GaussDB outputs a hexadecimal character string, and MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty. • Default value: The default value cannot be set in MySQL 5.7 but can be set in GaussDB and MySQL 8.0. • Primary key: When creating a primary key, you must specify the prefix length in MySQL, but you cannot specify the prefix length in GaussDB. • Index: In MySQL, the MEDIUMTEXT type does not support other index methods except prefix indexes. GaussDB supports these index methods.
LONGTEXT	• Input format: If a binary or hexadecimal character string is input, GaussDB outputs a hexadecimal character string, and MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty. • Default value: The default value cannot be set in MySQL 5.7 but can be set in GaussDB and MySQL 8.0. • Primary key: When creating a primary key, you must specify the prefix length in MySQL, but you cannot specify the prefix length in GaussDB. • Index: In MySQL, the LONGTEXT type does not support other index methods except prefix indexes. GaussDB supports these index methods.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.2.1.4 Binary Data Types

Table 3-137 Binary data types

Data Type	Differences Compared with MySQL
BINARY[(M)]	• Input format: – After a binary or hexadecimal character string is entered, GaussDB outputs a hexadecimal character string, and MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty. – If the length of the inserted string is less than the target length, the padding character is 0x20 in GaussDB and 0x00 in MySQL. • Character set: The default character set is the initialized character set of the database. For MySQL, the default character set is BINARY. • Output formats: – When the JDBC protocol is used, a space at the end of the BINARY type is displayed as a space, and that in MySQL is displayed as \x00. – In loose mode, if a character (such as Chinese character) of the BINARY type exceeds *n* bytes, the entire character is truncated. MySQL retains the first *n* bytes. However, garbled characters are displayed in the output. – In MySQL 8.0 and later versions, results starting with 0x are returned by default. In GaussDB, results in the format of "\x...\x...\x..." are returned. **NOTE** Due to the differences between GaussDB and MySQL in BINARY fillers and \0 truncation, GaussDB and MySQL have different performance in scenarios such as operator comparison calculation, character string-related system function calculation, index matching, and data import and export. For details about the difference scenarios, see the examples in this section.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Data Type	Differences Compared with MySQL
VARBINARY(M)	• Input format: If a binary or hexadecimal character string is input, GaussDB outputs a hexadecimal character string, and MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty. • Character set: The default character set is the initialized character set of the database. For MySQL, the default character set is BINARY. • Output formats: – When the JDBC protocol is used, a space at the end of the BINARY type is displayed as a space, and that in MySQL is displayed as \x00. – In MySQL 8.0 and later versions, results starting with 0x are returned by default. In GaussDB, results in the format of "\x...\x...\x..." are returned.
TINYBLOB	• Input format: – Default value: When creating a table column, you can set a default value in the syntax. However, this operation is not allowed in MySQL. – After a binary or hexadecimal character string is entered, GaussDB outputs a hexadecimal character string, and MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty. • Primary key: In MySQL, the TINYBLOB type does not support primary keys, but GaussDB supports. • Index: In MySQL, the TINYBLOB type does not support other index methods except prefix indexes. GaussDB supports these index methods. • Foreign key: In MySQL, the TINYTEXT type cannot be used as the referencing column or referenced column of a foreign key, but GaussDB supports this operation. • Output format: In MySQL 8.0 and later versions, results starting with 0x are returned by default. In GaussDB, results in the format of "\x...\x...\x..." are returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Data Type	Differences Compared with MySQL
BLOB	• Input format: – Default value: When creating a table column, you can set a default value in the syntax. However, this operation is not allowed in MySQL. – After a binary or hexadecimal character string is entered, GaussDB outputs a hexadecimal character string, and MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty. • Primary key: In MySQL, the BLOB type does not support primary keys, but GaussDB supports. • Index: In MySQL, the BLOB type does not support other index methods except prefix indexes. GaussDB supports these index methods. • Foreign key: In MySQL, the TINYTEXT type cannot be used as the referencing column or referenced column of a foreign key, but GaussDB supports this operation. • Output format: In MySQL 8.0 and later versions, results starting with 0x are returned by default. In GaussDB, results in the format of "\x...\x...\x..." are returned.
MEDIUMBLOB	• Input format: – Default value: When creating a table column, you can set a default value in the syntax. However, this operation is not allowed in MySQL. – After a binary or hexadecimal character string is entered, GaussDB outputs a hexadecimal character string, and MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty. • Primary key: In MySQL, the MEDIUMBLOB type does not support primary keys, but GaussDB supports. • Index: In MySQL, the MEDIUMBLOB type does not support other index methods except prefix indexes. GaussDB supports these index methods. • Foreign key: In MySQL, the TINYTEXT type cannot be used as the referencing column or referenced column of a foreign key, but GaussDB supports this operation. • Output format: In MySQL 8.0 and later versions, results starting with 0x are returned by default. In GaussDB, results in the format of "\x...\x...\x..." are returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Data Type	Differences Compared with MySQL
LONGBLOB	• Value range: a maximum of 1 GB. MySQL supports a maximum of 4 GB minus 1 byte. • Input format: – Default value: When creating a table column, you can set a default value in the syntax. However, this operation is not allowed in MySQL. – After a binary or hexadecimal character string is entered, GaussDB outputs a hexadecimal character string, and MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty. • Primary key: In MySQL, the LONGBLOB type does not support primary keys, but GaussDB supports. • Index: In MySQL, the LONGBLOB type does not support other index methods except prefix indexes. GaussDB supports these index methods. • Foreign key: In MySQL, the TINYTEXT type cannot be used as the referencing column or referenced column of a foreign key, but GaussDB supports this operation. • Output format: In MySQL 8.0 and later versions, results starting with 0x are returned by default. In GaussDB, results in the format of "\x...\x...\x..." are returned.
BIT[(M)]	Output formats: • All outputs are displayed as binary character strings. MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty. • In MySQL 8.0 and later versions, 0 is added at the beginning of each result by default. In GaussDB, 0 is not added.

Example:

```
-- GaussDB
m_db=# CREATE TABLE test(a BINARY(10)) DISTRIBUTE BY REPLICATION;
CREATE TABLE
m_db=# INSERT INTO test VALUES(0x8000);
INSERT 0 1
m_db=# SELECT hex(a) FROM test;
        hex
----------------------
 80202020202020202020
(1 row)

m_db=# SELECT * FROM test WHERE hex(a) = 80000000000000000000;
 a
---
(0 rows)

m_db=# CREATE TABLE test2(a BINARY(10)) DISTRIBUTE BY REPLICATION;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
CREATE TABLE
m_db=# INSERT INTO test2 VALUES(0x80008000);
INSERT 0 1
m_db=# SELECT hex(a) FROM test2;
        hex
---------------------
 80202020202020202020
(1 row)

m_db=# DROP TABLE test;
DROP TABLE
m_db=# DROP TABLE test2;
DROP TABLE

-- MySQL
mysql> CREATE TABLE test(a BINARY(10));
Query OK, 0 rows affected (0.01 sec)

mysql> INSERT INTO test VALUES(0x8000);
Query OK, 1 row affected (0.00 sec)

mysql> SELECT hex(a) FROM test;
+----------------------+
| hex(a)               |
+----------------------+
| 80000000000000000000 |
+----------------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM test WHERE hex(a) = 80000000000000000000;
+------------+
| a          |
+------------+
| ▒          |
+------------+
1 row in set (0.00 sec)

mysql> CREATE TABLE test2(a binary(10));
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO test2 VALUES(0x80008000);
Query OK, 1 row affected (0.00 sec)

mysql> SELECT hex(a) FROM test2;
+----------------------+
| hex(a)               |
+----------------------+
| 80008000000000000000 |
+----------------------+
1 row in set (0.00 sec)
mysql> DROP TABLE test;
Query OK, 0 rows affected (0.00 sec)
mysql> DROP TABLE test2;
Query OK, 0 rows affected (0.00 sec)
```

3.3.2.1.5 Attributes Supported by Data Types

Table 3-138 Attributes supported by data types

MySQL	GaussDB
NULL	Supported.
NOT NULL	Supported.
DEFAULT	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB
ON UPDATE	Supported.
PRIMARY KEY	Supported.
AUTO_INCREMENT	Supported.
CHARACTER SET name	Supported.
COLLATE name	Supported.
ZEROFILL	Supported.

When CREATE TABLE AS is used to create a table and default values are set for columns of the VARBINARY type, the command output of **SHOW CREATE TABLE**, **DESC**, or **\d** is different from that of MySQL. The value displayed in GaussDB is a hexadecimal value, but MySQL displays the original value.

Example:

```
m_db=# CREATE TABLE test_int(
        int_col INT
);
m_db=# CREATE TABLE test_varbinary(
        varbinary_col VARBINARY(20) default 'gauss'
) AS SELECT * FROM test_int;
m_db=# SHOW CREATE TABLE test_varbinary;
     Table      |                              Create Table
----------------+-----------------------------------------------------------------------------
 test_varbinary | SET search_path = public;                                                    +
                | CREATE TABLE test_varbinary (                                               +
                |     varbinary_col varbinary(20) DEFAULT X'6761757373',                      +
                |     int_col integer                                                         +
                | )                                                                           +
                | CHARACTER SET = "UTF8" COLLATE = "utf8mb4_general_ci"                       +
                | WITH (orientation=row, compression=no, storage_type=USTORE, segment=off);
(1 row)
m_db=# DROP TABLE test_int, test_varbinary;

mysql> CREATE TABLE test_int(
        int_col INT
);
mysql> CREATE TABLE test_varbinary(
        varbinary_col VARBINARY(20) default 'gauss'
) AS SELECT * FROM test_int;
mysql> SHOW CREATE TABLE test_varbinary;
+----------------
+-------------------------------------------------------------------------------------------------------------------+
| Table          | Create
Table                                                                                                              |
+----------------
+-------------------------------------------------------------------------------------------------------------------+
| test_varbinary | CREATE TABLE `test_varbinary` (
  `varbinary_col` varbinary(20) DEFAULT 'gauss',
  `int_vol` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 |
+----------------
+-------------------------------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)
mysql> DROP TABLE test_int, test_varbinary;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.2.1.6 Data Type Conversion

Conversion between different data types is supported. Data type conversion is involved in the following scenarios:

- The data types of operands of operators (such as comparison and arithmetic operators) are inconsistent. It is commonly used for comparison operations in query conditions or join conditions.
- The data types of arguments and parameters are inconsistent when a function is called.
- The data types of target columns to be updated by DML statements (including INSERT, UPDATE, MERGE, and REPLACE) and the defined column types are inconsistent.
- Explicit type conversion: CAST(expr AS datatype), which converts an expression to a data type.
- After the target data type of the final projection column is determined by set operations (UNION and EXCEPT), the type of the projection column in each SELECT statement is inconsistent with the target data type.
- In other expression calculation scenarios, the target data type used for comparison or final result is determined based on the data type of different expressions.
- When the operand of an operator is a function, implicit conversion may occur on the return value type of the function (such as GREATEST, LEAST, IF, NULLIF, IFNULL, ELTM, DAYNAME, UUID, ACOS, ASIN, TIMESTAMP, CONVERTTZ, and AVGFINALF). The return value type is different from that when the function is used independently.

There are three types of data type conversion differences: implicit conversion, explicit conversion, and UNION/CASE.

Differences in Double Colon Conversion

- In GaussDB, if you use double colons to convert input parameters of a function to another type, the result may be unexpected. In MySQL, double colons do not take effect.

  Example:

```
m_db=# SELECT POW("12"::VARBINARY,"12"::VARBINARY);
ERROR:  The value is out of range, and an overflow occurred.
CONTEXT:  referenced column: pow

varbinary col
m_db=# CREATE TABLE test_varbinary (
       A VARBINARY(10)
);
m_db=# INSERT INTO test_varbinary VALUES ('12');
m_db=# SELECT POW(A, A) FROM test_varbinary;
     pow
---------------
 8916100448256
(1 row)
```

Differences in Implicit Type Conversion

- In GaussDB, the conversion rules from small types to small types are used. In MySQL, the conversion rules from small types to large types and from large types to small types are used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Due to data type differences, some output formats of implicit conversion in GaussDB are inconsistent.
- During implicit conversion from the BIT data type to the character data type and binary data type in GaussDB, some output behaviors are inconsistent. GaussDB outputs a hexadecimal character string, and MySQL escapes the character string based on the ASCII code table. If the character string cannot be escaped, the output is empty.

  Example:

```
m_db=# CREATE TABLE bit_storage (
    VS_COL1 BIT(4),
    VS_COL2 BIT(4),
    VS_COL3 BIT(4),
    VS_COL4 BIT(4),
    VS_COL5 BIT(4),
    VS_COL6 BIT(4),
    VS_COL7 BIT(4),
    VS_COL8 BIT(4)
) DISTRIBUTE BY REPLICATION;
m_db=# CREATE TABLE string_storage (
    VS_COL1 BLOB,
    VS_COL2 TINYBLOB,
    VS_COL3 MEDIUMBLOB,
    VS_COL4 LONGBLOB,
    VS_COL5 TEXT,
    VS_COL6 TINYTEXT,
    VS_COL7 MEDIUMTEXT,
    VS_COL8 LONGTEXT
) DISTRIBUTE BY REPLICATION;
m_db=# INSERT INTO bit_storage VALUES(B'101', B'101', B'101', B'101', B'101', B'101', B'101', B'101');
m_db=# INSERT INTO string_storage SELECT * FROM bit_storage;
m_db=# SELECT * FROM string_storage;
 VS_COL1 | VS_COL2 | VS_COL3 | VS_COL4 | VS_COL5 | VS_COL6 | VS_COL7 | VS_COL8
---------+---------+---------+---------+---------+---------+---------+---------
 \x05    | \x05    | \x05    | \x05    | \x05    | \x05    | \x05    | \x05
(1 row)
m_db=# DROP TABLE bit_storage, string_storage;

mysql> CREATE TABLE bit_storage (
    VS_COL1 BIT(4),
    VS_COL2 BIT(4),
    VS_COL3 BIT(4),
    VS_COL4 BIT(4),
    VS_COL5 BIT(4),
    VS_COL6 BIT(4),
    VS_COL7 BIT(4),
    VS_COL8 BIT(4)
);
mysql> CREATE TABLE bit_storage (
    VS_COL1 BIT(4),
    VS_COL2 BIT(4),
    VS_COL3 BIT(4),
    VS_COL4 BIT(4),
    VS_COL5 BIT(4),
    VS_COL6 BIT(4),
    VS_COL7 BIT(4),
    VS_COL8 BIT(4)
);
mysql> INSERT INTO bit_storage VALUES(B'101', B'101', B'101', B'101', B'101', B'101', B'101', B'101');
mysql> INSERT INTO string_storage SELECT * FROM bit_storage;
mysql> SELECT * FROM string_storage;
+---------+---------+---------+---------+---------+---------+---------+---------+
| VS_COL1 | VS_COL2 | VS_COL3 | VS_COL4 | VS_COL5 | VS_COL6 | VS_COL7 | VS_COL8 |
+---------+---------+---------+---------+---------+---------+---------+---------+
|         |         |         |         |         |         |         |         |
+---------+---------+---------+---------+---------+---------+---------+---------+
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
1 row in set (0.00 sec)
mysql> DROP TABLE bit_storage, string_storage;
```

- When a binary or hexadecimal character string with 0x00 is inserted into the binary data type, GaussDB inserts part of the string and truncates the characters following 0x00. MySQL can insert the entire string.

  Example:

```
m_db=# CREATE TABLE blob_storage (
        A BLOB
) DISTRIBUTE BY REPLICATION;
m_db=# INSERT INTO blob_storage VALUES (0xBB00BB);
m_db=# SELECT hex(A) FROM blob_storage;
 hex
-----
 BB
(1 row)
m_db=# DROP TABLE blob_storage;

mysql> CREATE TABLE blob_storage (
        A BLOB
);
mysql> INSERT INTO blob_storage VALUES (0xBB00BB);
mysql> SELECT hex(A) FROM blob_storage;
+--------+
| hex(a) |
+--------+
| BB00BB |
+--------+
1 row in set (0.01 sec)
mysql> DROP TABLE blob_storage;
```

- When a binary or hexadecimal string with 0x00 in the middle is inserted into the string data type, GaussDB inserts part of the string and truncates the characters following 0x00. In MySQL, the string cannot be inserted in strict mode, and an empty string is inserted in loose mode.

  Example:

```
m_db=# CREATE TABLE text_storage (
        A TEXT
);
m_db=# INSERT INTO text_storage VALUES (b'101110110000000010111011');
m_db=# SELECT hex(A) FROM text_storage;
 hex
-----
 BB
(1 row)
m_db=# DROP TABLE text_storage;

mysql> CREATE TABLE text_storage (
        A TEXT
);
mysql> INSERT INTO text_storage VALUES (b'101110110000000010111011');
ERROR 1366 (HY000): Incorrect string value: '\xBB\x00\xBB' for column 'A' at row 1
mysql> SELECT hex(A) FROM text_storage;
Empty set (0.00 sec)
mysql> SET SQL_MODE='';
mysql> INSERT INTO text_storage VALUES (b'101110110000000010111011');
mysql> SELECT hex(A) FROM text_storage;
+--------+
| hex(A) |
+--------+
|        |
+--------+
1 row in set (0.01 sec)
mysql> DROP TABLE text_storage;
```

- The WHERE clause contains only common character strings. GaussDB returns **TRUE** for **'t'**, **'true'**, **'yes'**, **'y'**, and **'on'**, returns **FALSE** for **'no'**, **'f'**, **'off'**, **'false'**,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

and **'n'**, and reports an error for other character strings. MySQL determines whether to return **TRUE** or **FALSE** by converting a character string to an INT1 value.

Example:

```
m_db=# CREATE TABLE test_where (
       A INT
);
m_db=# INSERT INTO test_where VALUES (1);
m_db=# SELECT * FROM test_where WHERE '111';
ERROR:  invalid input syntax for type boolean: "111"
LINE 1: SELECT * FROM test_where WHERE '111';
m_db=# DROP TABLE test_where;

mysql> CREATE TABLE test_where (
       A INT
);
mysql> INSERT INTO test_where VALUES (1);
mysql> SELECT * FROM test_where WHERE '111';
+------+
| a    |
+------+
|    1 |
+------+
1 row in set (0.01 sec)
mysql> DROP TABLE test_where;
```

- When converting strings of the YEAR type to integers, MySQL uses scientific notation, but GaussDB does not support scientific notation and truncates the strings.

  Example:

```
m_db=# CREATE TABLE test_year (
       A YEAR
);
m_db=# SET sql_mode = '';
m_db=# INSERT INTO test_year VALUES ('2E3x');
WARNING:  Data truncated for column.
LINE 1: INSERT INTO test_year VALUES ('2E3x');
                                      ^
CONTEXT:  referenced column: a
m_db=# SELECT * FROM test_year ORDER BY A;
  a
------
 2002
(1 row)
m_db=# DROP TABLE test_year;

mysql> CREATE TABLE test_year (
       A YEAR
);
mysql> INSERT INTO test_year VALUES ('2E3x');
mysql> SELECT * FROM test_year ORDER BY A;
+------+
| a    |
+------+
| 2000 |
+------+
1 row in set (0.01 sec)
mysql> DROP TABLE test_year;
```

- MySQL 5.7 has two different behaviors when converting strings to integers. The criteria for determining whether a string prefix is blank are different. As a result, the result of converting the same string to an integer varies in different scenarios. MySQL 8.0 and GaussDB use the same behavior when converting a string to an integer. Only spaces and tab characters are considered as leading spaces.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- GaussDB: The result of converting a string to an integer is consistent in all scenarios.
m_db=# SELECT cast('\n1' as signed);
WARNING:  The INTEGER value '
1' is incorrect.
CONTEXT:  referenced column: cast
 cast
------
    0
(1 row)

m_db=# SELECT left('Hello', '\n1');
WARNING:  The INTEGER value '
1' is incorrect.
CONTEXT:  referenced column: left
 left
------

(1 row)

m_db=# SELECT left('Hello', concat('\n', '1'));
 left
------

(1 row)

-- In MySQL 5.7, when the left function is nested with the concat function, the \n character is skipped as a prefix. In other scenarios, the \n character is not skipped.
mysql> SELECT cast('\n1' as signed);
+-----------------------+
| cast('\n1' as signed) |
+-----------------------+
|                     0 |
+-----------------------+
1 row in set, 1 warning (0.00 sec)

Warning (Code 1292): Truncated incorrect INTEGER value: '
1'
mysql>  SELECT left('Hello', '\n1');
+----------------------+
| left('Hello', '\n1') |
+----------------------+
|                      |
+----------------------+
1 row in set, 2 warnings (0.00 sec)

Warning (Code 1292): Truncated incorrect INTEGER value: '
1'
Warning (Code 1292): Truncated incorrect INTEGER value: '
1'
mysql>  SELECT left('Hello', concat('\n', '1'));
+-----------------------------------+
| left('Hello', concat('\n', '1')) |
+-----------------------------------+
| H                                 |
+-----------------------------------+
1 row in set (0.00 sec)

-- In MySQL 8.0, the result of converting a string to an integer is consistent in all scenarios.
mysql>  SELECT cast('\n1' as signed);
+-----------------------+
| cast('\n1' as signed) |
+-----------------------+
|                     0 |
+-----------------------+
1 row in set, 1 warning (0.00 sec)

Warning (Code 1292): Truncated incorrect INTEGER value: '
1'
mysql>  SELECT left('Hello', '\n1');
```

```
+----------------------+
| left('Hello', '\n1') |
+----------------------+
|                      |
+----------------------+
1 row in set, 2 warnings (0.00 sec)

Warning (Code 1292): Truncated incorrect INTEGER value: '
1'
Warning (Code 1292): Truncated incorrect INTEGER value: '
1'
mysql> SELECT left('Hello', concat('\n', '1'));
+-----------------------------------+
| left('Hello', concat('\n', '1')) |
+-----------------------------------+
|                                   |
+-----------------------------------+
1 row in set, 2 warnings (0.00 sec)

Warning (Code 1292): Truncated incorrect INTEGER value: '
1'
Warning (Code 1292): Truncated incorrect INTEGER value: '
1'
```

Differences in Explicit Type Conversion

- In GaussDB, the conversion rules for each target type are used. In MySQL, C++ polymorphic overloading functions are used, causing inconsistent behavior in nesting scenarios.

  Example:

```
m_db=# SELECT CAST(GREATEST(date'2023-01-01','2023-01-01') AS SIGNED);
WARNING:  Truncated incorrect INTEGER value: '2023-01-01'
CONTEXT:  referenced column: cast
 cast
------
 2023
(1 row)

mysql> SELECT CAST(GREATEST(date'2023-01-01','2023-01-01') AS SIGNED);
+-----------------------------------------------------------+
| CAST(GREATEST(date'2023-01-01','2023-01-01') AS SIGNED) |
+-----------------------------------------------------------+
|                                                  20230101 |
+-----------------------------------------------------------+
```

Differences Between UNION, CASE, and Related Structures

- In MySQL, POLYGON+NULL, POINT+NULL, and POLYGON+POINT return the GEOMETRY type. They are not involved in GaussDB and considered as errors.
- The SET and ENUM types are not supported currently and are considered as errors.
- When the constant type is aggregated with other types, the precision of the output type is the precision of other types. For example, the precision of the result of "SELECT "helloworld" UNION SELECT p FROM t;" is the precision of attribute p.
- When fixed-point constants and types without precision constraints (non-string types such as int, bool, and year, and the type of the aggregation result is the fixed-point type) are aggregated, the precision constraint is output based on the default precision 31 of fixed-point numbers.
- Differences in merge rules:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

In MySQL 5.7, if YEAR is aggregated with TINYINT, INT, MEDIUMINT, BIGINT, or BOOL, the result is of the type with UNSIGNED. In GaussDB, it is of the type without UNSIGNED. In MySQL, if BIT is aggregated with a numeric type such as INT, NUMERIC, FLOAT, or DOUBLE, the result type is VARBINARY. In GaussDB, the result type is NUMERIC for aggregation between BIT and INT or NUMERIC, DOUBLE for aggregation between BIT and FLOAT or DOUBLE, and UINT8 for aggregation between BIT and unsigned integers.

- In MySQL, BINARY and CHAR use different padding characters. BINARY is padded with '\0', and CHAR is padded with spaces. In GaussDB, BINARY and CHAR are padded with spaces.

3.3.2.2 System Functions

3.3.2.2.1 System Function Compatibility Overview

GaussDB is compatible with most MySQL system functions, but there are some differences.

Currently, some system functions in GaussDB with the same names as those in MySQL are not supported in M-compatible mode. For some of them, the message indicating that they are not supported in M-compatible mode is displayed. Other functions still retain the behaviors of the original GaussDB system functions. The behavior of functions with the same name is greatly different from that of MySQL. Therefore, you are advised to avoid using them but use only system functions in M-compatible mode.

The following table lists the functions with the same name.

Table 3-139 Same-name functions for which a message indicating that they are not supported in M-compatible mode is displayed

cot	isEmpty	mod	octet_length	overlaps
point	radians	regexp_instr	regexp_like	regexp_replace
regexp_substr	stddev_pop	stddev_samp	var_pop	var_samp
variance	-	-	-	-

Table 3-140 Same-name functions that retain the behaviors of the original GaussDB system functions in M-compatible mode

ceil	decode	encode	format	instr
position	round	stddev	row_num	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- MySQL allows you to add user-defined functions to the database through the loadable functions. When such functions are called, aliases can be specified in the input parameters of the functions. GaussDB does not support loadable functions. When a function is called, aliases cannot be specified for input parameters of the function.
- In M-compatible mode, system functions have the following differences:
  - The return value type of a system function is the same as that of MySQL only when the node type of the input parameter is Var (table data) or Const (constant input). In other cases (for example, the input parameter is a calculation expression or function expression), the return value type may be different from that of MySQL.
  - When an aggregate function uses an expression such as another function, operator, or SELECT clause as the input parameter (for example, **SELECT sum(abs(n)) FROM t;**), the aggregate function cannot obtain the precision passed by the input parameter expression. As a result, the precision of the function result is different from that of MySQL.
  - Calling system functions by pg_catalog.func_name() is not recommended. If the called function has input parameters in the format of syntax (such as SELECT substr('demo' from 1 for 2)), an error may occur when the function is called.
  - For a system function that supports multiple input parameters, if an input parameter is **NULL**, GaussDB does not verify the input parameters and directly returns **NULL**; MySQL verifies the input parameters that are not **NULL**. If an input parameter is invalid, the function returns **NULL** and generates a warning.
  - When MySQL calls a system function with backquotes (``), the system function is case-insensitive. When GaussDB calls a system function with backquotes (``), the function name must be in lowercase.

```
-- GaussDB
m_db=# SELECT `abs`(1);
 abs
-----
   1
(1 row)

m_db=# SELECT `Abs`(1);
ERROR:  Function Abs(integer) does not exist.
LINE 1: select `Abs`(1);
               ^
HINT:  No function matches the given name and argument types. You might need to add explicit type casts.
CONTEXT:  referenced column: Abs

-- MySQL
mysql> SELECT `abs`(1);
+----------+
| `abs`(1) |
+----------+
|        1 |
+----------+
1 row in set (0.00 sec)

mysql> SELECT `Abs`(1);
+----------+
| `Abs`(1) |
+----------+
|        1 |
+----------+
1 row in set (0.01 sec)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.2.2.2 Flow Control Functions

Table 3-141 Flow control functions

Function	Differences Compared with MySQL
IF()	If the first parameter is **TRUE** and the third parameter expression contains an implicit type conversion error, or if the first parameter is **FALSE** and the second parameter expression contains an implicit type conversion error, MySQL ignores the error while GaussDB displays a type conversion error.
IFNULL()	If the first parameter is not **NULL** and the expression of the second parameter contains an implicit type conversion error, MySQL ignores the error while GaussDB reports a type conversion error.
NULLIF()	The return type of a function differs in MySQL 5.7 and MySQL 8.0. Return types are compatible with MySQL 8.0 because it is more appropriate.

3.3.2.2.3 Date and Time Functions

The date and time functions in the M-compatible mode in GaussDB, with the same behavior as MySQL, are described as follows:

- Functions may use time expressions as their input parameters.

  Time expressions (mainly including TEXT, DATETIME, DATE, and TIME) and types that can be implicitly converted to time expressions can be used as input parameters. For example, a number can be implicitly converted to text and then used as a time expression.

  However, different functions take effect in different ways. For example, the DATEDIFF function calculates only the difference between dates. Therefore, the time expression is parsed as the date type. The TIMESTAMPDIFF function parses the time expression as DATE, TIME, or DATETIME based on the **UNIT** parameter before calculating the time difference.

- The input parameters of functions may contain an invalid date.

  Generally, the supported DATE and DATETIME ranges are the same as those in MySQL. The value of DATE ranges from '0000-01-01' to '9999-12-31', and the value of DATETIME ranges from '0000-01-01 00:00:00' to '9999-12-31 23:59:59'. Although the DATE and DATETIME ranges supported by GaussDB are greater than those supported by MySQL, out-of-bounds dates are still invalid.

  In most cases, time functions report an alarm and return NULL if the input date is invalid, unless the invalid date can be converted by CAST.

Most date and time functions in the GaussDB M-compatible framework are the same as those in MySQL. **Table 3-142** lists the differences between them in terms of some functions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-142 Date and time functions

Function	Differences Compared with MySQL
ADDDATE()	-
ADDTIME()	-

CONVERT_TZ()	• MySQL allows you to use 'system' to specify the system time zone. GaussDB does not support the use of 'system' to specify the time zone.
	• Before converting the time, CONVERT_TZ checks whether the time is valid and within the range of the TIMESTAMP type ('1970-01-01 00:00:01' UTC to '2038-01-19 03:14:07' UTC). If the time is valid but not within the range, the original value is returned. MySQL checks whether the value of the first input parameter converted to UTC using the second input parameter (from_tz) meets the requirements. GaussDB only checks the original value of the first input parameter.
	-- GaussDB
	m_db=# SELECT convert_tz('1970-01-01 00:00:01', '-01:00', '+01:00'),
	convert_tz('1970-01-01 00:00:01', '+01:00', '+03:00');
	convert_tz | convert_tz
	---+---
	1970-01-01 02:00:01 | 1970-01-01 02:00:01
	(1 row)
	
	m_db=# SELECT convert_tz('1970-01-01 00:00:00', '-01:00', '+01:00'),
	convert_tz('1970-01-01 00:00:00', '+01:00', '+03:00');
	convert_tz | convert_tz
	---+---
	1970-01-01 00:00:00 | 1970-01-01 00:00:00
	(1 row)
	
	-- MySQL
	mysql> SELECT convert_tz('1970-01-01 00:00:01', '-01:00', '+01:00'),
	convert_tz('1970-01-01 00:00:01', '+01:00', '+03:00');
	+---
	+---+
	| convert_tz('1970-01-01 00:00:01', '-01:00', '+01:00') |
	convert_tz('1970-01-01 00:00:01', '+01:00', '+03:00') |
	+---
	+---+
	| 1970-01-01 02:00:01 | 1970-01-01
	00:00:01 |
	+---
	+---+
	1 row in set (0.00 sec)
	
	mysql> SELECT convert_tz('1970-01-01 00:00:00', '-01:00', '+01:00'),
	convert_tz('1970-01-01 00:00:00', '+01:00', '+03:00');
	+---
	+---+
	| convert_tz('1970-01-01 00:00:00', '-01:00', '+01:00') |
	convert_tz('1970-01-01 00:00:00', '+01:00', '+03:00') |
	+---
	+---+
	| 1970-01-01 02:00:00 | 1970-01-01
	00:00:00 |
	+---
	+---+
	1 row in set (0.00 sec)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Differences Compared with MySQL
CURDATE()	-
CURRENT_DATE() /CURRENT_DATE	-
CURRENT_TIME() /CURRENT_TIME	In MySQL, an integer input value is wrapped when it reaches **255** (maximum value of a one-byte integer value), for example, **SELECT CURRENT_TIME(257) == SELECT CURRENT_TIME(1)**. GaussDB supports only valid values ranging from 0 to 6. For other values, an error is reported.
CURRENT_TIMESTAMP()/ CURRENT_TIMESTAMP	
CURTIME()	
LOCALTIME()/ LOCALTIME	
LOCALTIMESTAMP/ LOCALTIMESTAMP()	
NOW()	
SYSDATE()	
UTC_TIME()	
UTC_TIMESTAMP()	
DATE()	-
DATE_ADD()	-
DATE_FORMAT()	-
DATE_SUB()	-
DATEDIFF()	-
DAY()	-
DAYNAME()	-
DAYOFMONTH()	-
DAYOFWEEK()	-
DAYOFYEAR()	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Differences Compared with MySQL

EXTRACT()	• When the value of **unit** in EXTRACT(unit FROM date) is **DAY_HOUR**, **DAY_MINUTE**, DAY_SECOND, or **DAY_MICROSECOND**, GaussDB parses the date input parameter based on the datetime format, while MySQL parses the date input parameter based on the time format.
	-- GaussDB
	m_db=# SELECT EXTRACT(DAY_HOUR FROM '10:30:45') AS res;
	WARNING: Incorrect datetime value: '10:30:45'
	CONTEXT: referenced column: res
	res
	----- 	
	
	(1 row)
	
	-- MySQL
	mysql> SELECT EXTRACT(DAY_HOUR FROM '10:30:45') AS res;
	+------+
	| res |
	+------+
	| 10 |
	+------+
	1 row in set (0.00 sec)
	• In EXTRACT(unit FROM date), when the input parameter **date** is of the datetime type and the value of the input parameter **unit** is **DAY_HOUR**, **DAY_MINUTE**, **DAY_SECOND**, or **DAY_MICROSECOND**, GaussDB returns the DAY part in the result, while MySQL returns the DAY part only when **date** is the table data. For example:
	-- Behaviors in GaussDB:
	m_db=# DROP TABLE IF EXISTS t_datetime;
	DROP TABLE
	
	m_db=# CREATE TABLE t_datetime(c1 datetime(6));
	CREATE TABLE
	
	m_db=# INSERT INTO t_datetime VALUES('2023-01-27 10:11:12.400000');
	INSERT 0 1
	
	-- When the input parameter **date** is table data, the result contains the DAY part.
	m_db=# SELECT EXTRACT(DAY_MICROSECOND FROM c1) FROM t_datetime;
	extract
	---------- 	
	27101112400000
	(1 row)
	
	-- When the input parameter **date** is constant data, the result also contains the DAY part.
	m_db=# SELECT EXTRACT(DAY_MICROSECOND FROM cast('2023-01-27 10:11:12.400000' as datetime(6)));
	extract
	---------- 	
	27101112400000
	(1 row)
	
	m_db=# DROP TABLE IF EXISTS t_datetime;
	DROP TABLE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Differences Compared with MySQL

	-- Behaviors in MySQL:
	mysql> DROP TABLE IF EXISTS t_datetime;
	Query OK, 0 rows affected (0.04 sec)
	
	mysql> CREATE TABLE t_datetime(c1 datetime(6));
	Query OK, 0 rows affected (0.05 sec)
	
	mysql> INSERT INTO t_datetime VALUES('2023-01-27 10:11:12.400000');
	Query OK, 1 row affected (0.00 sec)
	
	-- When the input parameter **date** is table data, the result contains the DAY part.
	mysql> SELECT EXTRACT(DAY_MICROSECOND FROM c1) FROM t_datetime;
	+----------------------------------+
	| EXTRACT(DAY_MICROSECOND FROM c1) |
	+----------------------------------+
	| 27101112400000 |
	+----------------------------------+
	1 row in set (0.01 sec)
	
	-- When the input parameter **date** is constant data, the result does not contain the DAY part.
	mysql> SELECT EXTRACT(DAY_MICROSECOND FROM cast('2023-01-27 10:11:12.400000' as datetime(6)));
	+---+
	| EXTRACT(DAY_MICROSECOND FROM cast('2023-01-27 10:11:12.400000' as datetime(6))) |
	+---+
	| 101112400000 |
	+---+
	1 row in set (0.00 sec)
	
	mysql> DROP TABLE IF EXISTS t_datetime;
	Query OK, 0 rows affected (0.04 sec)

FROM_DAYS()	-
FROM_UNIXTIME()	-
GET_FORMAT()	-
HOUR()	-
LAST_DAY()	-
MAKEDATE()	-
MAKETIME()	In the distributed pushdown scenario, if no second precision is specified for the TIME type, MySQL pads six trailing zeros by default, but GaussDB does not pad anything.
MICROSECOND()	-
MINUTE()	-
MONTH()	-
MONTHNAME()	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Differences Compared with MySQL

PERIOD_ADD()
PERIOD_DIFF()
	MySQL 8.0 has fixed the following issues. In the following scenarios, the behavior of this function is the same as that in MySQL 8.0 and is different from that in MySQL 5.7.
	● Processing integer overflow:
	In MySQL 5.7, the maximum value of an input parameter result of this function is **2^32=4294967296**. When the accumulated value of the month corresponding to **period** and the value of **month_number** in the input parameter or result exceed the uint32 range, integer wraparound occurs. This issue does not occur in GaussDB.
	● Performance when the value of **period** is negative:
	In MySQL 5.7, a negative year is parsed as an abnormal value instead of an error. Conversely, GaussDB reports an error when any input parameter or result is negative (for example, January 100 minus 10000 months).
	● Performance when the month in **period** exceeds the range:
	When dealing with a month greater than 12 or equal to 0, for example, **200013** or **199900**, MySQL 5.7 postpones it to the next year or views month 0 as December of the previous year. GaussDB reports an error for months beyond the range.
	● Behavior when the input parameter cannot be strictly converted to an integer:
	In MySQL 5.7, if the input parameter cannot be strictly converted to an integer, the parameter is forcibly converted to an integer. In GaussDB, an error is reported.
	-- GaussDB
	m_db=# SELECT period_add('20.1', 1);
	WARNING: The INTEGER value '20.1' is incorrect.
	CONTEXT: referenced column: period_add
	ERROR: Incorrect arguments to period_add
	CONTEXT: referenced column: period_add
	m_db=# SELECT period_add('20ab', 1);
	WARNING: The INTEGER value '20ab' is incorrect.
	CONTEXT: referenced column: period_add
	ERROR: Incorrect arguments to period_add
	CONTEXT: referenced column: period_add
	
	-- MySQL
	mysql> SELECT period_add('20.1', 1);
	+-----------------------+
	| period_add('20.1', 1) |
	+-----------------------+
	| 200109 |
	+-----------------------+
	1 row in set, 1 warning (0.00 sec)
	
	Warning (Code 1292): Truncated incorrect INTEGER value: '20.1'
	mysql> SELECT period_add('20ab', 1);
	+-----------------------+
	| period_add('20ab', 1) |
	+-----------------------+
	| 200109 |
	+-----------------------+

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Differences Compared with MySQL
	1 row in set, 1 warning (0.00 sec) Warning (Code 1292): Truncated incorrect INTEGER value: '20ab'
QUARTER()	-
SEC_TO_TIME()	-
SECOND()	-
STR_TO_DATE()	GaussDB returns values of the text type, while MySQL returns values of the datetime or date type.
SUBDATE()	-
SUBTIME()	-
TIME()	-
TIME_FORMAT()	-
TIME_TO_SEC()	-
TIMEDIFF()	-
TIMESTAMP()	-
TIMESTAMPADD()	-
TIMESTAMPDIFF()	-
TO_DAYS()	-
TO_SECONDS()	-
UNIX_TIMESTAMP()	MySQL determines whether to return a fixed-point value or an integer based on whether an input parameter contains decimal places. When operators or functions are nested in the input parameter, GaussDB may return a value of the type different from that in MySQL. If the inner node returns a value of the fixed-point, floating-point, string, or time type (excluding the DATE type), MySQL may return an integer, while GaussDB returns a fixed-point value.
UTC_DATE()	-
WEEK()	-
WEEKDAY()	-
WEEKOFYEAR()	-
YEAR()	-
YEARWEEK()	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.2.2.4 String Functions

Table 3-143 String functions

Function	Differences Compared with MySQL
ASCII()	-
BIT_LENGTH()	-
CHAR()	If the return value contains invalid characters, GaussDB directly returns the invalid characters, while MySQL returns the truncated string in loose mode and returns NULL in strict mode.
CHAR_LENGTH()	In GaussDB, if the character set is SQL_ASCII, CHAR_LENGTH() returns the number of bytes instead of characters.
CHARACTER_LENGTH()	In GaussDB, if the character set is SQL_ASCII, CHARACTER_LENGTH() returns the number of bytes instead of characters.
CONCAT()	For binary return values, MySQL offers various options (including BINARY, VARBINARY, and BLOB), while GaussDB offers only one—LONGBLOB. For non-binary return values, MySQL offers various options (including CHAR, VARCHAR, and TEXT), while GaussDB only offers TEXT.
CONCAT_WS()	For binary return values, MySQL offers various options (including BINARY, VARBINARY, and BLOB), while GaussDB offers only one—LONGBLOB. For non-binary return values, MySQL offers various options (including CHAR, VARCHAR, and TEXT), while GaussDB only offers TEXT.
HEX()	-
LENGTH()	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Differences Compared with MySQL
LPAD()	• The default maximum padding length in MySQL is **1398101**, and that in GaussDB is **1048576**. The maximum padding length varies depending on the character set. For example, if the character set is GBK, the default maximum padding length in GaussDB is **2097152**. • When GaussDB uses the SQL_ASCII, the server interprets byte values 0 to 127 according to the ASCII standard, and byte values 128 to 255 are regarded as characters that cannot be parsed. If the input and output of the function contain any non-ASCII data, the database cannot convert or verify non-ASCII characters. As a result, the behavior of the function is greatly different from that of MySQL. • For binary return values, MySQL offers various options (including BINARY, VARBINARY, and BLOB), while GaussDB offers only one—LONGBLOB. For non-binary return values, MySQL offers various options (including CHAR, VARCHAR, and TEXT), while GaussDB only offers TEXT.
REPEAT()	For binary return values, MySQL offers various options (including BINARY, VARBINARY, and BLOB), while GaussDB offers only one—LONGBLOB. For non-binary return values, MySQL offers various options (including CHAR, VARCHAR, and TEXT), while GaussDB only offers TEXT.
REPLACE()	For binary return values, MySQL offers various options (including BINARY, VARBINARY, and BLOB), while GaussDB offers only one—LONGBLOB. For non-binary return values, MySQL offers various options (including CHAR, VARCHAR, and TEXT), while GaussDB only offers TEXT.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Differences Compared with MySQL
RPAD()	• The default maximum padding length in MySQL is **1398101**, and that in GaussDB is **1048576**. The maximum padding length varies depending on the character set. For example, if the character set is GBK, the default maximum padding length in GaussDB is **2097152**. • When GaussDB uses the SQL_ASCII, the server interprets byte values 0 to 127 according to the ASCII standard, and byte values 128 to 255 are regarded as characters that cannot be parsed. If the input and output of the function contain any non-ASCII data, the database cannot convert or verify non-ASCII characters. As a result, the behavior of the function is greatly different from that of MySQL. • For binary return values, MySQL offers various options (including BINARY, VARBINARY, and BLOB), while GaussDB offers only one—LONGBLOB. For non-binary return values, MySQL offers various options (including CHAR, VARCHAR, and TEXT), while GaussDB only offers TEXT.
SPACE()	-
STRCMP()	When GaussDB uses the SQL_ASCII, the server interprets byte values 0 to 127 according to the ASCII standard, and byte values 128 to 255 are regarded as characters that cannot be parsed. If the input and output of the function contain any non-ASCII data, the database cannot convert or verify non-ASCII characters. As a result, the behavior of the function is greatly different from that of MySQL.
FIND_IN_SET()	When GaussDB uses the SQL_ASCII, the server interprets byte values 0 to 127 according to the ASCII standard, and byte values 128 to 255 are regarded as characters that cannot be parsed. If the input and output of the function contain any non-ASCII data, the database cannot convert or verify non-ASCII characters. As a result, the behavior of the function is greatly different from that of MySQL. For binary return values, MySQL offers various options (including BINARY, VARBINARY, and BLOB), while GaussDB offers only one—LONGBLOB. For non-binary return values, MySQL offers various options (including CHAR, VARCHAR, and TEXT), while GaussDB only offers TEXT.
LCASE()	
LEFT()	
LOWER()	
LTRIM()	
REVERSE()	
RIGHT()	
RTRIM()	
SUBSTR()	
SUBSTRING()	
SUBSTRING_INDEX()	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Differences Compared with MySQL
TRIM()	
UCASE()	
UPPER()	
UNHEX()	The return type in MySQL is BINARY, VARBINARY, BLOB, MEDIUMBLOB, or LONGBLOB, while the return type in GaussDB is fixed to LONGBLOB.
FIELD()	-
FORMAT()	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.2.2.5 Forced Conversion Functions

Table 3-144 Forced conversion functions

Function	Differences Compared with MySQL

CAST()	• In GaussDB, CAST(expr AS CHAR[(N)] charset_info or CAST(expr AS NCHAR[(N)]) cannot be used to convert character sets.
	• In GaussDB, you can use CAST(expr AS FLOAT[(p)]) or CAST(expr AS DOUBLE) to convert an expression to the floating-point type. MySQL 5.7 does not support this conversion.
	• When the CAST(expr AS FLOAT[(p)]) function is used, if the parameter **p** meets the condition of 0 ≤ **p** < 24, the return value type of the function is FLOAT. However, when the function is used in the CREATE TABLE/VIEW [AS] SELECT statement, the type of the corresponding column in the created table or view changes to DOUBLE. This behavior is inconsistent with that in MySQL 8.0.
	• In GaussDB, CAST(expr AS JSON) cannot be used to convert expressions to JSON.
	• In the CAST nested subquery scenario, if the subquery statement returns the FLOAT type, an accurate value is returned in GaussDB while a distorted value is returned in MySQL 5.7. The same rule applies to the BINARY function implemented using CAST.
	--GaussDB
	m_db=# CREATE TABLE sub_query_table(myfloat float) DISTRIBUTE BY REPLICATION;
	CREATE TABLE
	m_db=# INSERT INTO sub_query_table(myfloat) VALUES (1.23);
	INSERT 0 1
	m_db=# SELECT binary(SELECT myfloat FROM sub_query_table) FROM sub_query_table;
	binary
	-------- 	
	1.23
	(1 row)
	m_db=# SELECT cast((SELECT myfloat FROM sub_query_table) AS char);
	cast
	------ 	
	1.23
	(1 row)
	--MySQL 5.7
	mysql> CREATE TABLE sub_query_table(myfloat float);
	Query OK, 0 rows affected (0.02 sec)
	mysql> INSERT INTO sub_query_table(myfloat) VALUES (1.23);
	Query OK, 1 row affected (0.00 sec)
	mysql> SELECT binary(SELECT myfloat FROM sub_query_table) FROM sub_query_table;
	+---+
	| binary(SELECT myfloat FROM sub_query_table) |
	+---+
	| 1.2300000190734863 |
	+---+
	1 row in set (0.00 sec)
	mysql> SELECT cast((SELECT myfloat FROM sub_query_table) AS char);
	+---+

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Differences Compared with MySQL

		cast((SELECT myfloat FROM sub_query_table) AS char) |
		+---+
		| 1.2300000190734863 |
		+---+
		1 row in set (0.00 sec)
		• When CAST(expr as CHAR[(p)]) is used, MySQL supports the specified character set, but GaussDB does not.
		-- GaussDB
		m_db=# SELECT CAST('hello' AS CHAR CHARACTER SET utf8);
		ERROR: syntax error at or near "CHARACTER"
		LINE 1: SELECT CAST('hello' AS CHAR CHARACTER SET utf8);
		^
		-- Mysql 5.7
		mysql> SELECT CAST('hello' AS CHAR CHARACTER SET utf8);
		+---+
		| CAST('hello' AS CHAR CHARACTER SET utf8) |
		+---+
		| hello |
		+---+
		1 row in set (0.00 sec)
		• When CAST(expr AS CHAR[(p)]) is used, if the returned result is an invalid character, MySQL 5.7 and GaussDB directly return the invalid character, and MySQL 8.0 returns the truncated string in loose mode and NULL in strict mode.

CONVERT()	• In GaussDB, CONVERT(expr, CHAR[(N)] charset_info or CAST(expr, NCHAR[(N)]) cannot be used to convert character sets. • In GaussDB, you can use **CONVERT(expr, FLOAT[(p)])** or **CONVERT(expr, DOUBLE)** to convert an expression to the one of the floating-point type. MySQL 5.7 does not support this conversion. • When the CONVERT(expr, FLOAT[(p)]) function is used, if the parameter **p** meets the condition of 0 ≤ **p** < 24, the return value type of the function is FLOAT. However, when the function is used in the CREATE TABLE/VIEW [AS] SELECT statement, the type of the corresponding column in the created table or view changes to DOUBLE. This behavior is inconsistent with that in MySQL 8.0. • In GaussDB, CONVERT(expr, JSON) cannot be used to convert expressions to JSON.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.2.2.6 Encryption Functions

Table 3-145 Encryption functions

Function	Differences Compared with MySQL
AES_DECRYPT()	• GaussDB does not support ECB mode, which is an insecure encryption mode, but uses CBC mode by default. • When characters are specified to be encoded in SQL_ASCII for GaussDB, the server parses byte values 0 to 127 according to the ASCII standard, and byte values 128 to 255 cannot be parsed. If the input and output of the function contain any non-ASCII characters, the database cannot convert or verify them. • The return type in MySQL is BINARY, VARBINARY, BLOB, MEDIUMBLOB, or LONGBLOB, while the return type in GaussDB is fixed to LONGBLOB.
AES_ENCRYPT()	
SHA()/SHA1()	-
SHA2()	-

3.3.2.2.7 Comparison Functions

Table 3-146 Comparison functions

Function	Differences Compared with MySQL
COALESCE()	In the union distinct scenario, the precision of the return value is different from that in MySQL. • When an implicit type conversion error occurs in the subsequent parameter expression of the first parameter that is not NULL, MySQL ignores the error, but GaussDB displays a type conversion error. When the parameter is the MIN or MAX function, the return value type is different from that in MySQL. • If all parameters passed to a function are integers and there are unsigned integer input parameters, the return value type in MySQL 5.7 is bigint. If the result is greater than the upper limit of bigint, the result will be truncated. GaussDB and MySQL 8.0 return a value of the decimal type. When the parameter is greater than the upper limit of bigint, the result is not truncated.
INTERVAL()	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Differences Compared with MySQL
GREATEST()	• When the data types of parameters passed to a function are inconsistent, the return value type of the function is different from that in MySQL. • For binary return values, MySQL offers various options (including BINARY, VARBINARY, and BLOB), while GaussDB offers only one—LONGBLOB. For non-binary return values, MySQL offers various options (including CHAR, VARCHAR, and TEXT), while GaussDB only offers TEXT. • If the input parameter of the function contains **NULL** and the function is called after the WHERE keyword, the returned result is inconsistent with that of MySQL 5.7. This problem lies in MySQL 5.7. Since MySQL 8.0 has resolved this problem, GaussDB are consistent with MySQL 8.0.
LEAST()	
ISNULL()	-

3.3.2.2.8 Aggregate Functions

Table 3-147 Aggregate functions

Function	Differences Compared with MySQL
AVG()	• If **DISTINCT** is specified and the SQL statement contains the GROUP BY clause, GaussDB does not sort the results, while MySQL sorts the results. • In GaussDB, if the columns in **expr** are of the BIT, BOOL, or integer type and the sum of all rows exceeds the range of BIGINT, overflow occurs, reversing integers. • In GaussDB, if the type of expr is BIT, the return value type is double. In MySQL, the return value type is decimal.
BIT_AND()	-
BIT_OR()	-
BIT_XOR()	-
COUNT()	If **DISTINCT** is specified and the SQL statement contains the GROUP BY clause, GaussDB does not sort the results, while MySQL sorts the results.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Differences Compared with MySQL
GROUP_CONCAT()	• If **DISTINCT** is specified and the SQL statement contains the GROUP BY clause, GaussDB does not sort the results, while MySQL sorts the results. • If GROUP_CONCAT is to return a binary type, only the BLOB type will be returned. In other cases, the TEXT type will be returned. MySQL may return the longtext, tinytext, longblob, or tinyblob type based on the return length. • In GaussDB, if the parameters in GROUP_CONCAT contain both the DISTINCT and ORDER BY syntaxes, all expressions following ORDER BY must be in the DISTINCT expression. • In GaussDB, **GROUP_CONCAT(... ORDER BY** *Number***)** does not indicate the sequence of the parameter. The number is only a constant expression, which is equivalent to no sorting. • In GaussDB, the **group_concat_max_len** parameter is used to limit the maximum return length of GROUP_CONCAT. If the return length exceeds the maximum, the length is truncated. Currently, the maximum length that can be returned is **1073741823**, which is smaller than that in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

<table>
<tr><th>Function</th><th>Differences Compared with MySQL</th></tr>
<tr><td>MAX()
MIN()</td><td>

If DISTINCT is specified and the SQL statement contains the GROUP BY clause, GaussDB does not sort the results, while MySQL sorts the results.
When the parameter is not a table column, the return types of the MAX and MIN functions are different from those in MySQL 5.7.
When the parameter is of the float type, the return values of the MAX and MIN functions are the same as those in MySQL 5.7. The behavior of MySQL 5.7 and MySQL 8.0 is different. As a result, the return values of the MAX and MIN functions with CAST(expr AS FLOAT[(p)]) nested are different from those in MySQL 8.0.

<pre>
-- GaussDB:
m_db=# CREATE TABLE t1(c1 float);
CREATE TABLE

m_db=# INSERT INTO t1 VALUES(1.2);
INSERT 0 1

m_db=# SELECT MAX(c1) FROM t1;
        max

 1.2000000476837158
(1 row)

m_db=# SELECT MAX(CAST(1.2 AS FLOAT));
        max

 1.2000000476837158
(1 row)

m_db=# DROP TABLE t1;
DROP TABLE

-- MySQL 5.7:
mysql> CREATE TABLE t1(c1 float);
Query OK, 0 rows affected (0.02 sec)

mysql> INSERT INTO t1 VALUES(1.2);
Query OK, 1 row affected (0.00 sec)

mysql> SELECT MAX(c1) FROM t1;
+--------------------+
| MAX(c1) |
+--------------------+
| 1.2000000476837158 |
+--------------------+
1 row in set (0.00 sec)
-- MySQL 5.7 does not support the CAST(expr AS FLOAT[(p)]) expression.
mysql> SELECT MAX(CAST(1.2 AS FLOAT));
ERROR 1064 (42000): You have an error in your SQL syntax; check the
manual that corresponds to your MySQL server version for the right
syntax to use near 'FLOAT))' at line 1

mysql> DROP TABLE t1;
Query OK, 0 rows affected (0.01 sec)

-- MySQL 8.0:
mysql> CREATE TABLE t1(c1 float);
Query OK, 0 rows affected (0.03 sec)
</pre>
</td></tr>
</table>

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Differences Compared with MySQL

	mysql> INSERT INTO t1 VALUES(1.2);
	Query OK, 1 row affected (0.00 sec)
	
	mysql> SELECT MAX(c1) FROM t1;
	+---------+
	| MAX(c1) |
	+---------+
	| 1.2 |
	+---------+
	1 row in set (0.00 sec)
	
	mysql> SELECT MAX(CAST(1.2 AS FLOAT));
	+-------------------------+
	| MAX(CAST(1.2 AS FLOAT)) |
	+-------------------------+
	| 1.2 |
	+-------------------------+
	1 row in set (0.00 sec)
	
	mysql> DROP TABLE t1;
	Query OK, 0 rows affected (0.01 sec)

SUM()	• If **DISTINCT** is specified and the SQL statement contains the GROUP BY clause, GaussDB does not sort the results, while MySQL sorts the results. • In GaussDB, if the columns in **expr** are of the BIT, BOOL, or integer type and the sum of all rows exceeds the range of BIGINT, overflow occurs, reversing integers. • In GaussDB, if the type of expr is BIT, the return value type is double. In MySQL, the return value type is decimal.

3.3.2.2.9 Numeric Operation Functions

Table 3-148 Numeric operation functions

Function	Differences Compared with MySQL
ABS()	-
ACOS()	-
ASIN()	-
ATAN()	-
ATAN2()	-
CEILING()	Some operation result types in GaussDB are inconsistent with those in MySQL. If the derived result is of the NUMERIC or integer type and can be stored as an integer type, the result type in MySQL is integer, but is still NUMERIC in GaussDB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Function	Differences Compared with MySQL
COS()	-
DEGREES()	-
EXP()	-
FLOOR()	The return types of the FLOOR function in GaussDB are different from those in MySQL. When the input parameter type is INT, the return type is BIGINT in GaussDB, but is INT in MySQL. Some operation result types in GaussDB are inconsistent with those in MySQL. If the derived result is of the NUMERIC or integer type and can be stored as an integer type, the result type in MySQL is integer, but is still NUMERIC in GaussDB.
LN()	-
LOG()	-
LOG10()	-
LOG2()	-
PI()	The precision of the return value of the PI function in GaussDB is different from that in MySQL. It is rounded off to 15 decimal places in GaussDB but to six decimal places in MySQL.
POW()	-
POWER()	-
RAND()	-
SIGN()	-
SIN()	-
SQRT()	-
TAN()	-
TRUNCATE()	-
CEIL()	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.2.2.10 Other Functions

Table 3-149 Other functions

Function	Differences Compared with MySQL
DATABASE()	-
UUID()	-
UUID_SHORT()	-

3.3.2.3 Operators

GaussDB is compatible with most MySQL operators, but there are some differences. If not listed, the operator behavior is the native behavior of GaussDB by default. Currently, there are statements that are not supported by MySQL but supported by GaussDB. In MySQL-compatible mode, they are usually used inside the system, so they are not recommended. You are advised to combine operators according to the rules in MySQL.

Operator Differences

- NULL values in ORDER BY are sorted in different ways. MySQL sorts NULL values first, while GaussDB sorts NULL values last. In GaussDB, you can use **NULLS FIRST** and **NULLS LAST** to set the sorting sequence of NULL values.
- If ORDER BY is used, the output sequence of GaussDB is the same as that of MySQL. Without ORDER BY, GaussDB does not guarantee that the results are ordered.
- When using MySQL operators, use parentheses to ensure the combination of expressions. Otherwise, an error is reported. For example, SELECT 1 regexp ('12345' regexp '123').

  The GaussDB M-compatible operators can be successfully executed without using parentheses to strictly combine expressions.
- NULL values are displayed in different ways. MySQL displays a NULL value as **"NULL"**. GaussDB displays a NULL value as empty.

  MySQL output:

  ```
  mysql> SELECT NULL;
  +------+
  | NULL |
  +------+
  | NULL |
  +------+
  1 row in set (0.00 sec)
  ```

  GaussDB output:

  ```
  m_db=# SELECT NULL;
   ?column?
  ----------

  (1 row)
  ```
- After the operator is executed, the column names are displayed in different ways. MySQL displays a NULL value as **"NULL"**. GaussDB displays a NULL value as empty.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- When character strings are being converted to the double type but there is an invalid one, the alarm is reported differently. MySQL reports an error when there is an invalid constant character string, but does not report an error for an invalid column character string. GaussDB reports an error in either situation.
- The results returned by the comparison operator are different. For MySQL, **1** or **0** is returned. For GaussDB, **t** or **f** is returned.

Table 3-150 Operators

Operators	Differences Compared with MySQL
<>	MySQL supports indexes, but GaussDB does not.
<=>	MySQL supports indexes, but GaussDB does not support indexes, hash joins, or merge joins.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operators	Differences Compared with MySQL

Row expressions	• MySQL supports row comparison using the <=> operator, but GaussDB does not support row comparison using the <=> operator.
	• MySQL does not support comparison between row expressions and NULL values. In GaussDB, the <, <=, =, >=, >, and <> operators can be used to compare row expressions with NULL values.
	• IS NULL or ISNULL operations on row expressions are not supported in MySQL. GaussDB supports the IS NULL operator but does not support the ISNULL operator for row expressions.
	• For operations by using operators that cannot be performed on row expressions, the error information in GaussDB is inconsistent with that in MySQL.
	GaussDB:
	m_db=# SELECT (1,2) <=> row(2,3);
	ERROR: could not determine interpretation of row comparison operator <=>
	LINE 1: SELECT (1,2) <=> row(2,3);
	^
	HINT: unsupported operator.
	m_db=# SELECT (1,2) < NULL;
	?column?
	---------- 	
	
	(1 row)
	m_db=# SELECT (1,2) <> NULL;
	?column?
	---------- 	
	(1 row)
	m_db=# SELECT (1, 2) IS NULL;
	?column?
	---------- 	
	f
	(1 row)
	m_db=# SELECT ROW(0,0) BETWEEN ROW(1,1) AND ROW(2,2);
	ERROR: Unsupport type.
	MySQL:
	mysql> SELECT (1,2) <=> row(2,3);
	+--------------------+
	| (1,2) <=> row(2,3) |
	+--------------------+
	| 0 |
	+--------------------+
	1 row in set (0.00 sec)
	
	mysql> SELECT (1,2) < NULL;
	ERROR 1241 (21000): Operand should contain 2 column(s)
	mysql> SELECT (1,2) <> NULL;
	ERROR 1241 (21000): Operand should contain 2 column(s)
	mysql> SELECT (1, 2) IS NULL;
	ERROR 1241 (21000): Operand should contain 1 column(s)
	mysql> SELECT ISNULL((1, 2));
	ERROR 1241 (21000): Operand should contain 1 column(s)
	mysql> SELECT NULL BETWEEN NULL AND ROW(2,2);
	ERROR 1241 (21000): Operand should contain 1 column(s)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operators	Differences Compared with MySQL
--	MySQL indicates that an operand is negated twice and the result is equal to the original operand. GaussDB indicates a comment.
!!	MySQL: The meaning of !! is the same as that of !, indicating NOT. GaussDB: ! indicates NOT. If there is a space between two exclamation marks, it indicates double NOT (! !). If there is no space between them, it indicates factorial (!!). **NOTE** • In GaussDB, when both factorial (!!) and NOT (!) are used, a space must be added between them. Otherwise, an error is reported. • In GaussDB, when multiple NOT operations are required, use a space between exclamation marks (! !).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operators	Differences Compared with MySQL
[NOT] REGEXP	• GaussDB and MySQL support different metacharacters in regular expressions. For example, GaussDB allows \d to indicate digits, \w to indicate letters, digits, and underscores (_), and \s to indicate spaces. However, MySQL does not support these metacharacters and considers them as normal character strings. • In GaussDB, "\b" can match "\\b", but in MySQL, the matching will fail. • In GaussDB, a backslash (\) indicates an escape character. In MySQL, two backslashes (\\) are used. • MySQL does not support two operators to be used together. • If the input parameter of the pattern string is invalid with only the right parenthesis ()), GaussDB and MySQL 5.7 will report an error, but MySQL 8.0 will not. • In the rule of matching the de|abc sequence with de or abc, when there are empty values on the left and right of the pipe symbol (|), MySQL 5.7 will report an error, but GaussDB and MySQL 8.0 will not. • The regular expression of the tab character "\t" can match the character class [:blank:] in GaussDB and MySQL 8.0 but cannot in MySQL 5.7. • GaussDB supports non-greedy pattern matching. That is, the number of matching characters is as small as possible. A question mark (?) is added after some special characters, for example, ?? *? +? {n}? {n,}? {n,m}? MySQL 5.7 does not support non-greedy pattern matching, and the error message "Got error 'repetition-operator operand invalid' from regexp" is displayed. MySQL 8.0 already supports this function. • In the BINARY character set, the TEXT and BLOB types are converted to the BYTEA type. The REGEXP operator does not support the BYTEA type. Therefore, the two types cannot be matched.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operators	Differences Compared with MySQL
LIKE	• MySQL: The left operand of LIKE can only be an expression of a bitwise or arithmetic operation, or expression consisting of parentheses. The right operand of LIKE can only be an expression consisting of unary operators (excluding NOT) or parentheses. GaussDB: The left and right operands of LIKE can be any expression. • When the right operand of LIKE uses the setFloat API of JDBC for binding parameters and JDBC uses the PBE mode for execution, the query results in GaussDB and MySQL are different. However, in an equivalent scenario where the parameter bound by setFloat is replaced with float data in the table, the behavior of GaussDB remains consistent, while the behavior of MySQL is inconsistent.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operators	Differences Compared with MySQL

[NOT] BETWEEN AND	MySQL: [NOT] BETWEEN AND is nested from right to left. The first and second operands of [NOT] BETWEEN AND can only be expressions of bitwise or arithmetic operations, or expressions consisting of parentheses.
	GaussDB: [NOT] BETWEEN AND is nested from left to right. The first and second operands of [NOT] BETWEEN AND can be any expression.
	In some special scenarios, for example, the expression contains the repeat() function, the result of [NOT] BETWEEN AND may be different from that of [NOT] (>= AND <=) in MySQL. However, GaussDB ensures that the result of [NOT] BETWEEN AND is consistent with that of [NOT] (>= AND <=).
	-- GaussDB: The behavior of BETWEEN AND is the same as that of (>= AND <=).
	m_db=# CREATE TABLE test_between(c0 YEAR);
	CREATE TABLE
	m_db=# INSERT INTO test_between(c0) VALUES('1963');
	INSERT 0 1
	m_db=# SELECT (''<>'Lt') BETWEEN '' AND c0 AS ref0 FROM test_between;
	 ref0
	------ 	
	 f
	(1 row)
	
	m_db=# SELECT (''<>'Lt') >= '' AND (''<>'Lt') <= c0 AS ref0 FROM test_between;
	 ref0
	------ 	
	 f
	(1 row)
	
	m_db=# DROP TABLE test_between;
	DROP TABLE
	
	-- MySQL: The behavior of BETWEEN AND is different from that of (>= AND <=).
	mysql> CREATE TABLE test_between(c0 YEAR);
	Query OK, 0 rows affected (0.03 sec)
	
	mysql> INSERT INTO test_between(c0) VALUES('1963');
	Query OK, 1 row affected (0.00 sec)
	
	mysql> SELECT (''<>'Lt') BETWEEN '' AND c0 AS ref0 FROM test_between;
	+------+
	| ref0 |
	+------+
	| 1 |
	+------+
	1 row in set (0.00 sec)
	
	mysql> SELECT (''<>'Lt') >= '' AND (''<>'Lt') <= c0 AS ref0 FROM test_between;
	+------+
	| ref0 |
	+------+
	| 0 |
	+------+
	1 row in set (0.00 sec)
	
	mysql> DROP TABLE test_between;
	Query OK, 0 rows affected (0.01 sec)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operators	Differences Compared with MySQL
IN	MySQL: The left operand of IN can only be an expression of a bitwise or arithmetic operation, or expression consisting of parentheses. GaussDB: The left operand of IN can be any expression.
!	MySQL: The operand of ! can only be an expression consisting of unary operators (excluding NOT) or parentheses. GaussDB: The operand of ! can be any expression.
#	MySQL supports the comment tag (#), but GaussDB does not.
BINARY	Expressions (including some functions and operators) supported by GaussDB are different from those supported by MySQL. For GaussDB-specific expressions such as "~" and "IS DISTINCT FROM", due to the higher priority of the BINARY keyword, when BINARY expr is used, BINARY is combined with the left parameters of "~" and "IS DISTINCT FROM" first. As a result, an error is reported.
Negation (-)	If the number of consecutive negation times exceeds 1, GaussDB identifies the negations as comments. As a result, it returns results different from MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Operators	Differences Compared with MySQL

XOR, |, & , < , > , <=, >=, =, and !=	The execution mechanism of MySQL is as follows: After the left operand is executed, the system checks whether the result is empty and then determines whether to execute the right operand.
	As for the execution mechanism of GaussDB, after the left and right operands are executed, the system checks whether the result is empty.
	If the result of the left operand is empty and an error is reported during the execution of the right operand, MySQL does not report an error but directly returns an error. GaussDB reports an error during the execution.
	Behavior in MySQL:
	`mysql> SELECT version();`
	`+-----------------+`
	`| version() |`
	`+-----------------+`
	`| 5.7.44-debug-log |`
	`+-----------------+`
	`1 row in set (0.00 sec)`
	
	`mysql> dROP TABLE IF EXISTS data_type_table;`
	`Query OK, 0 rows affected (0.02 sec)`
	
	`mysql> CREATE TABLE data_type_table (`
	`-> MyBool BOOL,`
	`-> MyBinary BINARY(10),`
	`-> MyYear YEAR`
	`-> );`
	`Query OK, 0 rows affected (0.02 sec)`
	
	`mysql> INSERT INTO data_type_table VALUES (TRUE, 0x1234567890, '2021');`
	`Query OK, 1 row affected (0.00 sec)`
	
	`mysql> SELECT (MyBool % MyBinary) | (MyBool - MyYear) FROM data_type_table;`
	`+----------------------------------------+`
	`| (MyBool % MyBinary) | (MyBool - MyYear) |`
	`+----------------------------------------+`
	`| NULL |`
	`+----------------------------------------+`
	`1 row in set, 2 warnings (0.00 sec)`
	Behavior in GaussDB:
	`m_db=# DROP TABLE IF EXISTS data_type_table;`
	`DROP TABLE`
	`m_db=# CREATE TABLE data_type_table (`
	`m_db(# MyBool BOOL,`
	`m_db(# MyBinary BINARY(10),`
	`m_db(# MyYear YEAR`
	`m_db(# );`
	`CREATE TABLE`
	`m_db=# INSERT INTO data_type_table VALUES (TRUE, 0x1234567890, '2021');`
	`INSERT 0 1`
	`m_db=# SELECT (MyBool % MyBinary) | (MyBool - MyYear) FROM data_type_table;`
	`WARNING: Truncated incorrect double value: '4Vx '`
	`CONTEXT: referenced column: (MyBool % MyBinary) | (MyBool - MyYear)`
	`WARNING: division by zero`
	`CONTEXT: referenced column: (MyBool % MyBinary) | (MyBool - MyYear)`
	`ERROR: Bigint is out of range.`
	`CONTEXT: referenced column: (MyBool % MyBinary) | (MyBool - MyYear)`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-151 Differences in operator combinations

Example of Operator Combination	MySQL	GaussDB	Description
SELECT 1 LIKE 3 & 1;	Not supported	Supported	The right operand of LIKE cannot be an expression consisting of bitwise operators.
SELECT 1 LIKE 1 +1;	Not supported	Supported	The right operand of LIKE cannot be an expression consisting of arithmetic operators.
SELECT 1 LIKE NOT 0;	Not supported	Supported	The right operand of LIKE can only be an expression consisting of unary operators (such as +, -, or ! but except NOT) or parentheses.
SELECT 1 BETWEEN 1 AND 2 BETWEEN 2 AND 3;	Right-to-left combination	Left-to-right combination	You are advised to add parentheses to specify the calculation priority to prevent result deviation caused by sequence differences.
SELECT 2 BETWEEN 1=1 AND 3;	Not supported	Supported	The second operand of BETWEEN cannot be an expression consisting of comparison operators.
SELECT 0 LIKE 0 BETWEEN 1 AND 2;	Not supported	Supported	The first operand of BETWEEN cannot be an expression consisting of pattern matching operators.
SELECT 1 IN (1) BETWEEN 0 AND 3;	Not supported	Supported	The first operand of BETWEEN cannot be an expression consisting of IN operators.
SELECT 1 IN (1) IN (1);	Not supported	Supported	The second left operand of the IN expression cannot be an expression consisting of INs.
SELECT ! NOT 1;	Not supported	Supported	The operand of ! can only be an expression consisting of unary operators (such as +, -, or ! but except NOT) or parentheses.

Index Differences

- Currently, GaussDB supports only UB-tree and B-tree indexes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- For fuzzy match (LIKE operator), the default index created can be used in MySQL, but cannot be used in GaussDB. You need to use the following syntax to specify **opclass** to, for example, **text_pattern_ops**, so that LIKE operators can be used as indexes:
  ```
  CREATE INDEX indexname ON tablename(col [opclass]);
  ```
- In the B-tree/UB-tree index scenario, the original logic of the native GaussDB is retained. That is, index scan supports comparison of types in the same operator family, but does not support other index types currently.
- When GaussDB JDBC is used to connect to the database, the YEAR type of GaussDB cannot use indexes in the PBE scenario that contains bind parameters.
- When some functions are nested in operator expressions, implicit conversion occurs on the return value type. If the return value type of a function is converted to a type different from that of the expression index, index scan cannot be used.
- In the operation scenarios involving index column type and constant type, the conditions that indexes of a WHERE clause are supported in GaussDB are different from those in MySQL. For details, see **Table 3-152**. For example, GaussDB does not support indexes in the following statement:
  ```
  CREATE TABLE t(_int int);
  CREATE INDEX idx ON t(_int) USING BTREE;
  SELECT * FROM t WHERE _int > 2.0;
  ```

  NOTE

  In the operation scenarios involving index column type and constant type in the WHERE clause, you can use the cast function to explicitly convert the constant type to the column type for indexing.

  ```
  SELECT * FROM t WHERE _int > cast(2.0 AS signed);
  ```

Table 3-152 Differences in index support

Index Column Type	Constant Type	Supported by GaussDB	Supported by MySQL
Integer	Integer	Yes	Yes
Floating-point	Floating-point	Yes	Yes
Fixed-point	Fixed-point	Yes	Yes
String	String	Yes	Yes
Binary	Binary	Yes	Yes
Time with date	Time with date	Yes	Yes
TIME	TIME	Yes	Yes
Time with date	Type that can be converted to time type with date (for example, integers such as 20231130)	Yes	Yes

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Index Column Type	Constant Type	Supported by GaussDB	Supported by MySQL
Time with date	TIME	Yes	Yes
TIME	Constants that can be converted to the TIME type (for example, integers such as 203008)	Yes	Yes
Floating-point	Integer	Yes	Yes
Floating-point	Fixed-point	Yes	Yes
Floating-point	String	Yes	Yes
Floating-point	Binary	Yes	Yes
Floating-point	Time with date	Yes	Yes
Floating-point	TIME	Yes	Yes
Fixed-point	Integer	Yes	Yes
String	Time with date	Yes	No
String	TIME	Yes	No
Binary	String	Yes	Yes
Binary	Time with date	Yes	No
Binary	TIME	Yes	No
Integer	Floating-point	No	Yes
Integer	Fixed-point	No	Yes
Integer	String	No	Yes
Integer	Binary	No	Yes
Integer	Time with date	No	Yes
Integer	TIME	No	Yes
Fixed-point	Floating-point	No	Yes
Fixed-point	String	No	Yes
Fixed-point	Binary	No	Yes
Fixed-point	Time with date	No	Yes
Fixed-point	TIME	No	Yes
String	Binary	No	Yes

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Index Column Type	Constant Type	Supported by GaussDB	Supported by MySQL
Time with date	Integer (that cannot be converted to the time type with date)	No	Yes
Time with date	Floating-point (that cannot be converted to the time type with date)	No	Yes
Time with date	Fixed-point (that cannot be converted to the time type with date)	No	Yes
TIME	Integer (that cannot be converted to the TIME type)	No	Yes
TIME	Character string (that cannot be converted to the TIME type)	No	Yes
TIME	Binary (that cannot be converted to the TIME type)	No	Yes
TIME	Time with date	No	Yes
YEAR	YEAR	Yes	Yes
YEAR	Constants that can be converted to the YEAR type (for example, integers such as 2034)	Yes	Yes
BIT	BIT	No	Yes

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-153 Whether index use is supported

Index Column Type	Constant Type	Use Index or Not in GaussDB	Use Index or Not in MySQL
String	Integer	No	No
String	Floating-point	No	No
String	Fixed-point	No	No
Binary	Integer	No	No
Binary	Floating-point	No	No
Binary	Fixed-point	No	No
Time with date	Character string (that cannot be converted to the time type with date)	No	No
Time with date	Binary (that cannot be converted to the time type with date)	No	No
TIME	Floating-point (that cannot be converted to the TIME type)	No	No
TIME	Fixed-point (that cannot be converted to the TIME type)	No	No
BIT	String	No	No

3.3.2.4 Character Sets

GaussDB allows you to specify the following character sets for databases, schemas, tables, or columns.

Table 3-154 Character sets

MySQL	GaussDB
utf8mb4	Supported
utf8	Supported
gbk	Supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB
gb18030	Supported
binary	Supported

NOTE

- GaussDB regards utf8 and utf8mb4 as the same character set. The maximum length of the code is 4 bytes. If the current character set is utf8 and the collation is set to **utf8mb4_bin**, **utf8mb4_general_ci**, **utf8mb4_unicode_ci**, or **utf8mb4_0900_ai_ci** (for example, by running **SELECT _utf8'a' collate utf8mb4_bin**), MySQL reports an error but GaussDB does not. The difference also exists when the character set is **utf8mb4** and the collation is set to **utf8_bin**, **utf8_general_ci**, or **utf8_unicode_ci**.
- Lexical and syntax parsing is performed based on the byte stream. When a multi-byte character contains the same encoding as the symbols such as '\', '\'' and '\\'. The behavior is inconsistent with that of MySQL. You are advised to disable the escape character processing to avoid this problem.
- GaussDB does not strictly verify the encoding logic of invalid characters that do not belong to the current character set. In strict mode, some character functions may allow such invalid characters to be successfully entered. Conversely, MySQL will report an error upon detecting such characters. In loose mode, GaussDB truncates such invalid characters and then inserts the value, which is the same as the behavior of MySQL. The strict and loose modes are determined by the **strict_trans_tables** configuration item of the GUC parameter **sql_mode**.
- The logic for converting between character sets in GaussDB is different from that in MySQL. When the binary character set is used together with other character sets, the transcoding process may cause inconsistent behavior. Therefore, when the binary character set is used, it is recommended that the character set-related parameters such as **client_encoding**, **server_encoding**, and **character_set_connection** be consistent and only characters within the ASCII range be entered.

3.3.2.5 Collation Rules

GaussDB allows you to specify the following collation rules for schemas, tables, or columns.

Table 3-155 Collation rules

MySQL	GaussDB
binary	Supported
gb18030_bin	Supported
gb18030_chinese_ci	Supported
gbk_bin	Supported
gbk_chinese_ci	Supported
utf8_bin	Supported
utf8_general_ci	Supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB
utf8_unicode_ci	Supported
utf8mb4_0900_ai_ci	Supported
utf8mb4_bin	Supported
utf8mb4_general_ci	Supported
utf8mb4_unicode_ci	Supported

NOTE

Differences in collation rules:

- Currently, collation rules can only be specified for the character string type and some binary types. Other types do not support specifying collation rules. You can check the **typcollation** attribute of a type in the pg_type system catalog. If it is not **0**, the type supports the collation. In MySQL, collation can be specified for all types; however, specifying collation rules for types other than character strings and binary types has no practical meaning.
- The current collation rules (except binary) can be specified only when the corresponding character set is the same as the database-level character set. In GaussDB, the character set must be the same as the database character set, and multiple character sets cannot be used together in a table.
- The default collation of the utf8mb4 character set is utf8mb4_general_ci, which is the same as that in MySQL 5.7.

3.3.2.6 Transactions

GaussDB is compatible with MySQL transactions, but there are some differences. This section describes transaction-related differences in GaussDB M-compatible databases.

Default Transaction Isolation Levels

The default isolation level of an M-compatible database is READ COMMITTED, and that of MySQL is REPEATABLE READ.

```
-- View the current transaction isolation level.
m_db=# SHOW transaction_isolation;
```

Sub-transactions

In an M-compatible database, SAVEPOINT is used to create a savepoint (sub-transaction) in the current transaction, and ROLLBACK TO SAVEPOINT is used to roll back to a savepoint (sub-transaction). After the sub-transaction is rolled back, the parent transaction can continue to run, the rollback of a sub-transaction does not affect the transaction status of the parent transaction.

No savepoint (sub-transaction) can be created in MySQL.

Nested Transactions

A nested transaction refers to a new transaction started in a transaction block.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

In an M-compatible database, if a new transaction is started in a normal transaction block, a warning is displayed indicating that an ongoing transaction exists and the start command is ignored. If a new transaction is started in an abnormal transaction block, an error is reported. The transaction can be executed only after **ROLLBACK** or **COMMIT** is executed. If **ROLLBACK** or **COMMIT** is executed, the previous statement is rolled back.

In MySQL, before a new transaction is started in a normal transaction block, the previous one must be committed. If a new transaction is started in an abnormal transaction block, the error is ignored and the previous correct statement is committed.

```
-- In an M-compatible database, if a new transaction is started in a normal transaction block, a warning is generated and the transaction is ignored.
m_db=# DROP TABLE IF EXISTS test_t;
m_db=# CREATE TABLE test_t(a int, b int);
m_db=# BEGIN;
m_db=# INSERT INTO test_t values(1, 2);
m_db=# BEGIN; -- The warning "There is already a transaction in progress" is displayed.
m_db=# SELECT * FROM test_t ORDER BY 1;
m_db=# COMMIT;

-- In an M-compatible database, if a new transaction is started in an abnormal transaction block, an error is reported. The transaction can be executed only after ROLLBACK or COMMIT is executed.
m_db=# BEGIN;
m_db=# ERROR sql; -- Error statement.
m_db=# BEGIN; -- An error is reported.
m_db=# COMMIT; -- It can be executed only after ROLLBACK/COMMIT is executed.
```

Statements Committed Implicitly

An M-compatible database uses GaussDB for storage and inherits the GaussDB transaction mechanism. If a DDL or DCL statement is executed in a transaction, the transaction is not automatically committed.

In MySQL, if DDL, DCL, management-related, or lock-related statements are executed, the transaction is automatically committed.

```
-- In M-compatible database, table creation and GUC parameter setting support rollback.
m_db=# DROP TABLE IF EXISTS test_table_rollback;
m_db=# BEGIN;
m_db=# CREATE TABLE test_table_rollback(a int, b int);
m_db=# \d test_table_rollback;
m_db=# ROLLBACK;
m_db=# \d test_table_rollback; -- This table does not exist.
```

Differences in SET TRANSACTION

In an M-compatible database, if SET TRANSACTION is used to set the isolation level or transaction access mode for multiple times, only the last setting takes effect. Transaction features can be separated by spaces or commas (,).

In MySQL, SET TRANSACTION cannot be used to set the isolation level or transaction access mode for multiple times. Transaction features can only be separated by commas (,).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-156 Differences in SET TRANSACTION

Syntax	Description	Difference
SET TRANSACTION	Sets transactions.	In an M-compatible database, SET TRANSACTION takes effect in session-level transactions. In MySQL, SET TRANSACTION takes effect in the next transaction.
SET SESSION TRANSACTION	Sets session-level transactions.	-
SET GLOBAL TRANSACTION	Sets global session-level transactions. This feature applies to subsequent sessions and has no impact on the current session.	In an M-compatible database, GLOBAL takes effect in global session-level transactions and is applicable only to the current database instance. In MySQL, this feature takes effect in all databases.

```
-- SET TRANSACTION takes effect in session-level transactions.
m_db=# SET TRANSACTION ISOLATION LEVEL READ COMMITTED READ WRITE;
m_db=# SHOW transaction_isolation;
m_db=# SHOW transaction_read_only;
-- In an M-compatible database, if the isolation level or transaction access mode is set for multiple times, only the last setting takes effect.
m_db=# SET SESSION TRANSACTION ISOLATION LEVEL READ COMMITTED, ISOLATION LEVEL REPEATABLE READ, READ WRITE, READ ONLY;
m_db=# SHOW transaction_isolation; -- repeatable read
m_db=# SHOW transaction_read_only; -- on
```

Differences in START TRANSACTION

In an M-compatible database, when START TRANSACTION is used to start a transaction, the isolation level can be set. If the isolation level or transaction access mode is set for multiple times, only the last setting takes effect. In the current version, consistency snapshot cannot be enabled immediately. Transaction features can be separated by spaces or commas (,).

In MySQL, if START TRANSACTION is used to start a transaction, the isolation level cannot be set and the transaction access mode cannot be set for multiple times. Transaction features can only be separated by commas (,).

```
-- Start a transaction and set the isolation level.
m_db=# START TRANSACTION ISOLATION LEVEL READ COMMITTED;
m_db=# COMMIT;
-- Set the access mode for multiple times.
m_db=# START TRANSACTION READ ONLY, READ WRITE;
m_db=# COMMIT;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Transaction-related GUC Parameters

Table 3-157 Differences in transaction-related GUC parameters

GUC Parameter	Description	Difference
autocommit	Sets the automatic transaction commit mode.	-
transaction_isolation	Sets the isolation level of the current transaction in an M-compatible database. Sets the isolation level of a session-level transaction in MySQL.	• In GaussDB, you can only change the isolation level of the current transaction by running the **SET transaction_isolation =** *value* command. To change the session-level isolation level, use **default_transaction_isolation**. In MySQL, you can run the **SET** command to change the isolation level of a session-level transaction. • The supported range is different. MySQL supports the following isolation levels, which are case-insensitive but space-sensitive: – READ COMMITTED – READ UNCOMMITTED – REPEATABLE READ – SERIALIZABLE GaussDB supports the following isolation levels, which are case-sensitive and space-sensitive: – read committed – read uncommitted (It can be set, but **read committed** takes effect.) – repeatable read – serializable (It can be set, but **repeatable read** takes effect.) – default (same as the default isolation level in the session) – If **m_format_dev_version** is set to **'s2'**, the isolation levels of MySQL can be set. • In GaussDB, the value of **transaction_isolation** of a new transaction is initialized to the value of **default_transaction_isolation**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Description	Difference
tx_isolation	Sets the transaction isolation level. **tx_isolation** and **transaction_isolation** are synonyms.	This parameter does not support query or modification in an M-compatible database. If necessary, you are advised to use **transaction_isolation** for query.
default_transaction_isolation	Sets the transaction isolation level.	In an M-compatible database, the **SET** command is used to change the transaction isolation level for a session. MySQL does not support this system parameter.
transaction_read_only	Sets the access mode of a transaction.	• In an M-compatible database, only the access mode of the current transaction can be changed by using the **SET** command. If you want to change the access mode of a session-level transaction, you can use **default_transaction_read_only**. In MySQL, you can run the **SET** command to change the isolation level of a session-level transaction. • In GaussDB, the value of **transaction_read_only** of a new transaction is initialized to the value of **default_transaction_read_only**.
tx_read_only	Sets the access mode of a transaction. **tx_read_only** and **transaction_read_only** are synonyms.	This parameter does not support query or modification in an M-compatible database. If necessary, you are advised to use **transaction_read_only** for query.
default_transaction_read_only	Sets the access mode of a transaction.	In an M-compatible database, the **SET** command is used to change the access mode of a session-level transaction. MySQL does not support this system parameter.

3.3.2.7 SQL

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.2.7.1 SQL Compatibility Overview

GaussDB is compatible with most MySQL syntax, but there are some differences. This section describes the MySQL syntax supported by GaussDB.

- Some keywords can be used as identifiers in MySQL, but cannot or are restricted to be identifiers in M-compatible mode, as listed in **Table 3-158**.

  Table 3-158 Restricted identifiers

Keyword Type	Keyword	Restriction
Reserved (Type or function is allowed.)	COLLATION and COMPACT	They cannot be used as identifiers in other databases except for functions and variables.
Non-reserved (Type or function is not allowed.)	BIT, BOOLEAN, COALESCE, DATE, NATIONAL, NCHAR, NONE, NUMBER, TEXT, TIME, TIMESTAMP, and TIMESTAMPDIFF	They cannot be used as identifiers for functions or variables.
Reserved	ANY, ARRAY, BUCKETS, DO, END, LESS, MODIFY, OFFSET, ONLY, RETURNING, SOME, and USER	They cannot be used as identifiers in any database.

- The GaussDB optimizer is different from the MySQL optimizer. Due to the difference in the execution plans generated by optimizers, the GaussDB behavior may be inconsistent with the MySQL behavior, but it does not affect the service data result.

  For example, in the following scenario involving WHERE comparison, GaussDB calls the cast function when calculating **col1** and using **col1**; therefore, two WARNING records are generated. MySQL calls the cast function when calculating **col1** and then directly uses the calculated value for comparison; therefore, only one WARNING record is generated.

```
-- Behavior in GaussDB:
m_db=# SELECT * FROM (SELECT cast('abc' AS decimal) AS col1) t1 WHERE col1=0;
WARNING:  The DECIMAL value 'abc' is incorrect.
WARNING:  The DECIMAL value 'abc' is incorrect.
CONTEXT:  referenced column: col1
 col1
------
    0
(1 row)

m_db=# EXPLAIN VERBOSE SELECT * FROM (SELECT cast('abc' AS decimal) AS col1) t1 WHERE col1=0;
WARNING:  The DECIMAL value 'abc' is incorrect.
WARNING:  The DECIMAL value 'abc' is incorrect.
CONTEXT:  referenced column: col1
                QUERY PLAN
------------------------------------------
 Result  (cost=0.00..0.01 rows=1 width=0)
   Output: 0::decimal
(2 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Behavior in MySQL:
mysql>  SELECT * FROM (SELECT cast('abc' AS decimal) AS col1) t1 WHERE col1=0;
+------+
| col1 |
+------+
|    0 |
+------+
1 row in set, 1 warning (0.00 sec)

mysql> SHOW warnings;
+---------+------+----------------------------------------+
| Level   | Code | Message                                |
+---------+------+----------------------------------------+
| Warning | 1292 | Truncated incorrect DECIMAL value: 'abc' |
+---------+------+----------------------------------------+
1 row in set (0.00 sec)

mysql> EXPLAIN SELECT * FROM (SELECT cast('abc' AS decimal) AS col1) t1 WHERE col1=0;
+----+-------------+------------+------------+--------+---------------+------+---------+------+------+----------
+----------------+
| id | select_type | table      | partitions | type   | possible_keys | key  | key_len | ref  | rows | filtered |
Extra          |
+----+-------------+------------+------------+--------+---------------+------+---------+------+------+----------
+----------------+
|  1 | PRIMARY     | <derived2> | NULL       | system | NULL          | NULL | NULL    | NULL |    1 |
100.00 | NULL           |
|  2 | DERIVED     | NULL       | NULL       | NULL   | NULL          | NULL | NULL    | NULL | NULL |
NULL | No tables used |
+----+-------------+------------+------------+--------+---------------+------+---------+------+------+----------
+----------------+
2 rows in set, 2 warnings (0.01 sec)

mysql> SHOW warnings;
+---------+------+------------------------------------------------------------------+
| Level   | Code | Message                                                          |
+---------+------+------------------------------------------------------------------+
| Warning | 1292 | Truncated incorrect DECIMAL value: 'abc'                         |
| Note    | 1003 | /* select#1 */ select '0' AS `col1` from dual where ('0' = 0) |
+---------+------+------------------------------------------------------------------+
2 rows in set (0.00 sec)
```

3.3.2.7.2 Keywords

The constraint differences are as follows:

- If a keyword is a reserved one in M-compatible mode but non-reserved in MySQL, it cannot be a table name, column name, column alias, AS column alias, AS table alias, table alias, function name, or variable name in M-compatible mode, but can be any of these names or aliases in MySQL.
- If a keyword is a non-reserved one in M-compatible mode but reserved in MySQL, it can be a table name, column name, column alias, AS column alias, AS table alias, table alias, function name, or variable name in M-compatible mode, but cannot be any of these names or aliases in MySQL.
- If a keyword is a reserved one (function or type) both in M-compatible mode and MySQL, it can be a column alias, AS column alias, function name, or variable name in M-compatible mode, but cannot be any of these names or aliases in MySQL.
- If a keyword is a reserved one (function or type) in M-compatible mode but non-reserved in MySQL, it cannot be a table name, column name, AS table alias, or table alias in M-compatible mode, but can be one of these names or aliases in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If a keyword is a non-reserved one (excluding function and type) in M-compatible mode but reserved in MySQL, it can be a table name, column name, column alias, AS column alias, AS table alias, table alias, function name, or variable name in M-compatible mode, but cannot be any of these names or aliases in MySQL.
- If a keyword is a non-reserved one (excluding function and type) both in M-compatible mode and MySQL, it cannot be a function name in M-compatible mode, but can be a function name in MySQL.

  NOTE

  Among non-reserved keywords, reserved keywords (functions or types), and non-reserved keywords (not functions or types) in M-compatible mode, the following keywords cannot be used as column aliases:

  BETWEEN, BIGINT, BLOB, CHAR, CHARACTER, COMPILE, CROSS, DEC, DECIMAL, DIV, DOUBLE, EXISTS, FLOAT, FLOAT4, FLOAT8, GROUPING, INNER, INOUT, INT, INT1, INT2, INT3, INT4, INT8, INTEGER, JOIN, LEFT, LIKE, LONGBLOB, LONGTEXT, MEDIUMBLOB, MEDIUMINT, MEDIUMTEXT, MOD, NATURAL, NUMERIC, OUT, OUTER, PRECISION, REAL, RIGHT, ROW, ROW_NUMBER, SIGNED, SMALLINT, SOUNDS, TINYBLOB, TINYINT, TINYTEXT, VALUES, VARCHAR, VARYING, and WITHOUT.

  SIGNED and WITHOUT can be used as column aliases in MySQL.

3.3.2.7.3 Identifiers

Differences in identifiers in M-compatible mode are as follows:

- In GaussDB, unquoted identifiers cannot start with a dollar sign ($). In MySQL unquoted identifiers can start with a dollar sign ($).
- GaussDB unquoted identifiers support case-sensitive database objects.
- GaussDB identifiers support extended characters from U+0080 to U+00FF. MySQL identifiers support extended characters from U+0080 to U+FFFF.
- As for unquoted identifier, a table that starts with a digit and ends with an e or E as the identifier cannot be created in GaussDB. For example:

```
-- GaussDB reports an error indicating that this operation is not supported. MySQL supports this operation.
m_db=# CREATE TABLE 23e(c1 int);
ERROR:  syntax error at or near "23"
LINE 1: CREATE TABLE 23e(c1 int);
                     ^
m_db=# CREATE TABLE t1(23E int);
ERROR:  syntax error at or near "23"
LINE 1: CREATE TABLE t1(23E int);
                        ^
```

- As for quoted identifiers, tables whose column names contain only digits or scientific computing cannot be directly used in GaussDB. You need to use them in quotes. This rule also applies to the dot operator (.) scenarios. For example:

```
-- Create a table whose column names contain only numbers or scientific computing.
m_db=# CREATE TABLE t1(`123` int, `1e3` int, `1e` int);
CREATE TABLE

-- Insert data into the table.
m_db=# INSERT INTO t1 VALUES(7, 8, 9);
INSERT 0 1

-- The result is not as expected, but is the same as that in MySQL.
m_db=# SELECT 123 FROM t1;
 ?column?
----------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
     123
(1 row)

-- The result is not as expected, but is the same as that in MySQL.
m_db=# SELECT 1e3 FROM t1;
 ?column?
----------
    1000
(1 row)

-- The result is not as expected and is not the same as that in MySQL.
m_db=# SELECT 1e FROM t1;
 e
---
 1
(1 row)

-- The correct way to use is as follows:
m_db=# SELECT `123` FROM t1;
 123
-----
   7
(1 row)

m_db=# SELECT `1e3` FROM t1;
 1e3
-----
   8
(1 row)

m_db=# SELECT `1e` FROM t1;
 1e
----
  9
(1 row)

-- Dot operator scenarios are not supported by GaussDB but supported by MySQL.
m_db=# SELECT t1.123 FROM t1;
ERROR:  syntax error at or near ".123"
LINE 1: SELECT t1.123 FROM t1;
                ^
m_db=# SELECT t1.1e3 FROM t1;
ERROR:  syntax error at or near "1e3"
LINE 1: SELECT t1.1e3 FROM t1;
                 ^
m_db=# SELECT t1.1e FROM t1;
ERROR:  syntax error at or near "1"
LINE 1: SELECT t1.1e FROM t1;
                 ^
-- The correct way to use in dot operator scenarios is as follows:
m_db=# SELECT t1.`123` FROM t1;
 123
-----
   7
(1 row)

m_db=# SELECT t1.`1e3` FROM t1;
 1e3
-----
   8
(1 row)

m_db=# SELECT t1.`1e` FROM t1;
 1e
----
  9
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
m_db=# DROP TABLE t1;
DROP TABLE
```

- In GaussDB, the partition name is case-sensitive when it is enclosed in double quotation marks (**SQL_MODE** must include **ANSI_QUOTES**) or backquotes, but in MySQL the partition name is case-insensitive.
- The maximum length of a MySQL identifier is 64 characters, while that of a GaussDB identifier is 63 bytes. If the length of an identifier exceeds the limit, MySQL reports an error, while GaussDB truncates the identifier and generates an alarm.

3.3.2.7.4 DDL

Table 3-159 DDL syntax compatibility

Description	Syntax	Difference
Create primary keys, UNIQUE indexes, and foreign keys during table creation and modification.	ALTER TABLE and CREATE TABLE	• In GaussDB, when the table joined with the constraint is Ustore and USING BTREE is specified in the SQL statement, the underlying index is created as UB-tree. • GaussDB: Foreign keys can be used as partition keys. • The index name, constraint name, and key name are unique in a schema in GaussDB and unique in a table in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Support auto-increment columns.	ALTER TABLE and CREATE TABLE	• It is recommended that an auto-increment column in GaussDB be the first column of an index. Otherwise, an alarm is generated during table creation. The auto-increment column in MySQL must be the first column of the index. Otherwise, an error is reported during table creation. In GaussDB, an error occurs when some operations (such as ALTER TABLE EXCHANGE PARTITION) are performed on a table that contains auto-increment columns. • In GaussDB, for the AUTO_INCREMENT [=] value syntax, **value** must be a positive integer ranging from 1 to 2^{127} - 1. In MySQL, this parameter can be set to **0** (actual value: **1**). In GaussDB, this parameter cannot be set to **0**. • In GaussDB, when the value has reached the maximum value of the column data type during auto-increment and auto-increment continues, if **sql_mode** is set to strict, an error occurs. If **sql_mode** is set to loose, auto-increment can proceed but the value remains at the maximum value. The type conversion rules used by the GaussDB auto-increment counter and table column value conversion are the same as those used by expressions. The logic for

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		processing maximum value overflow is proper. In MySQL, when a value in an auto-increment column of the BIGINT UNSIGNED type reaches the maximum and auto-increment continues, an error is reported. In some scenarios, an error is reported as soon as it reaches the maximum value **–2**. However, there is no unified standard. If the MySQL auto-increment column is of the BIGINT or smaller integer type, when the maximum value is reached and auto-increment proceeds, the value is still the maximum value. ● GaussDB does not support the *innodb_autoinc_lock_mode* system variable, but when its GUC parameter **auto_increment_cache** is set to **0**, the behavior of inserting auto-increment columns in batches is similar to that when the MySQL system variable *innodb_autoinc_lock_mode* is set to **1**. ● In GaussDB, when 0s, NULLs, and definite values are imported or batch inserted into auto-increment columns, the auto-increment values inserted after an error occurs in GaussDB may not be the same as those in MySQL. The **auto_increment_cache** parameter is provided to control the number of reserved auto-increment values.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		• In GaussDB, when auto-increment is triggered by parallel import or insertion of auto-increment columns, the cache value reserved for each parallel thread is used only in the thread. If the cache value is not used up, the values of auto-increment columns in the table are discontinuous. The auto-increment value generated by parallel insertion cannot be guaranteed to be the same as that generated in MySQL. • GaussDB does not allow the value of **auto_increment_offset** to be greater than that of **auto_increment_increment**. Otherwise, an error occurs. MySQL allows it and states that **auto_increment_offset** will be ignored. • If a table has a primary key or index, the sequence in which the **ALTER TABLE** command rewrites table data may be different from that in MySQL. GaussDB rewrites table data based on the table data storage sequence, while MySQL rewrites table data based on the primary key or index sequence. As a result, the auto-increment sequence may be different. • When the ALTER TABLE statement is used to add or modify auto-increment columns in GaussDB, the number reserved for values in the first auto-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		increment is the number of rows in the table statistics, which may not be the same as that in MySQL. As a result, the auto-increment values are discontinuous in GaussDB and may be different from those in MySQL. • If the values of the GUC parameters **auto_increment_offset** and **auto_increment_increment** in GaussDB are out of range, an error occurs. MySQL automatically changes the value to a boundary value. • Temporary tables do not support auto-increment columns. • The local auto-increment columns are independent on each DN. The values of the LAST_INSERT_ID function and GeneratedKey cannot be updated when the insertion is successful. • If the **no_auto_value_on_zero** parameter is set for **sql_mode**, the auto-increment column defined in the table is subject to non-NOT NULL constraints. If no value is specified for auto-increment columns of a table during data insertion, **NULL** values are inserted into the columns in GaussDB. In MySQL, auto-increment is triggered and **NULL** values are inserted into the columns.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		• In different execution plans, the auto-increment sequence of local auto-increment columns and the number of reserved auto-increment columns may be different from those in MySQL. For example, "INSERT INTO table VALUES(...),(...),..." is distributed to different DNs. Therefore, in some execution plans, DNs cannot obtain the number of rows to be inserted. • Auto-increment and initial value update of the global auto-increment column in GaussDB are earlier than those in MySQL. Therefore, hole values are more likely to be generated when error occurs during data insertion. For example, if the CHECK constraint is violated during the INSERT process, the auto-increment column in MySQL 8.0 stops working, and the global auto-increment column in GaussDB distributed mode proceeds. • When ALTER TABLE is used to add a global auto-increment column in GaussDB, each DN applies for global auto-increment from the GTM. As a result, its auto-increment sequence is different from that in MySQL, and the hole values are also different. • The UPDATE statement updates the values of auto-increment columns. If a new value is not **0**,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		NULL, or **DEFAULT** and is greater than the maximum value of auto-increment, the initial value of auto-increment will be updated. The behavior is the same as that in MySQL 8.0. ● When the INSERT INTO ON DUPLICATE KEY statement is used to insert multiple rows of data at a time, if the UNIQUE constraint is violated, the generated holes may be different from those in MySQL. ● When the TRUNCATE is performed on a table in GaussDB, the global auto-increment column cannot be reset to automatically increase from the initial value, but you can do this with the AUTOINC_RESETVAL() function. Exercise caution when using this function. If data in the table is not cleared, duplicate values will be generated in the auto-increment column. ● When the INSERT INTO ON DUPLICATE KEY statement updates the value of the global auto-increment column due to primary key conflicts, for the getGeneratedKeys() API of JDBC, GaussDB returns the auto-increment value of the auto-increment column, and MySQL returns the updated value of the auto-increment column. ● When data is inserted into the global auto-increment column, the auto-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		increment time and the time for updating the auto-increment start value in GaussDB are earlier than those in MySQL. If an error is reported during data insertion and GaussDB may have updated the value of **last_insert_id**, the next query result of the last_insert_id() function may be the auto-increment value generated by the last statement for which the error is reported, which is different from that in MySQL.
Prefix indexes	CREATE INDEX, ALTER TABLE, and CREATE TABLE	• In GaussDB, the prefix length cannot exceed 2676. The actual length of the key value is restricted by the internal page. If a column contains multi-byte characters or an index has multiple keys, an error may be reported when the index line length exceeds the threshold. • In GaussDB, the primary key index does not support prefix keys. The prefix length cannot be specified when a primary key is created or added.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Specify character sets and collation rules.	ALTER SCHEMA, ALTER TABLE, CREATE SCHEMA, and CREATE TABLE	• When you specify a database-level character set, except binary character sets, the character set of a new database or schema cannot be different from that specified by **server_encoding** of the database. • When you specify a table-level or column-level character set and collation, MySQL allows you to specify a character set and collation that are different from the database-level character set and collation. In GaussDB, the table-level and column-level character sets and collations support only the binary character sets and collations or can be the same as the database-level character sets and collations.
Add columns before the first column of a table or after a specified column during table modification.	ALTER TABLE	-
Alter the column name/definition.	ALTER TABLE	Currently, the DROP INDEX, DROP KEY, or ORDER BY is not supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Create a partitioned table.	CREATE TABLE PARTITION	• MySQL supports expressions but does not support multiple partition keys in the following scenarios: – The LIST/RANGE partitioning policy is used and the COLUMNS keyword is not specified. – The hash partitioning policy is used. • MySQL does not support expressions but supports multiple partition keys in the following scenarios: – The LIST/RANGE partitioning policy is used and the COLUMNS keyword is specified. – The KEY partitioning policy is used. • In GaussDB, expressions cannot be used as partition keys, and partitions cannot be specified. • GaussDB supports multiple partition keys only when the LIST or RANGE partitioning policy is used. • In GaussDB partitioned tables, generated columns cannot be used as partition keys.
Specify table-level and column-level comments during table creation and modification.	CREATE TABLE and ALTER TABLE	-
Specify index-level comments during index creation.	CREATE INDEX	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Exchange the partition data of an ordinary table and a partitioned table.	ALTER TABLE PARTITION	Differences in ALTER TABLE EXCHANGE PARTITION: ● After **ALTER EXCHANGE PARTITION** is executed, the auto-increment columns are reset in MySQL, but in GaussDB, they are not reset and continue the auto-increment based on their old values. ● If MySQL tables or partitions use tablespaces, data in partitions and ordinary tables cannot be exchanged. If GaussDB tables or partitions use different tablespaces, data in partitions and ordinary tables can still be exchanged. ● MySQL does not verify the default values of columns. Therefore, data in partitions and ordinary tables can be exchanged even if the default values are different. GaussDB verifies the default values. If they are different, data in partitions and ordinary tables cannot be exchanged. ● After the DROP COLUMN operation is performed on a partitioned table or an ordinary table in MySQL, if the table definition is still consistent, data can be exchanged between partitions and ordinary tables. In GaussDB, data can be exchanged between partitions and ordinary tables only when the deleted columns of ordinary tables and partitioned tables are strictly aligned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		• MySQL and GaussDB use different hash algorithms. Therefore, data stored in the same hash partition may be inconsistent. As a result, the exchanged data may also be inconsistent. • MySQL's partitioned tables do not support foreign keys. If an ordinary table contains foreign keys or other tables reference the foreign keys of an ordinary table, data in partitions and ordinary tables cannot be exchanged. GaussDB's partitioned tables support foreign keys. If the FOREIGN KEY constraints of two tables are the same, data in partitions and ordinary tables can be exchanged. If a partitioned table does not contain foreign keys, an ordinary table is referenced by other tables, and the partitioned table is the same as the ordinary table, GaussDB also allows the partitioned table to exchange data with the ordinary table.
Modify the partition key information of a partitioned table.	ALTER TABLE	MySQL allows you to modify the partition key information of a partitioned table, but GaussDB does not.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
CREATE TABLE... LIKE syntax	CREATE TABLE ... LIKE	• In versions earlier than MySQL 8.0.16, CHECK constraints are parsed but their functions are ignored. In this case, CHECK constraints are not replicated. GaussDB supports replication of CHECK constraints. • During table creation, MySQL fixes all PRIMARY KEY constraint names at **PRIMARY KEY**. GaussDB does not support copying PRIMARY KEY constraint names. • When a table is created, MySQL supports replication of UNIQUE constraint names, but GaussDB does not. • When a table is created, MySQL versions earlier than 8.0.16 do not have CHECK constraint information, but GaussDB supports replication of CHECK constraint names. • During table creation, MySQL supports copying index names, but GaussDB does not. • When a table is created across sql_mode, MySQL is controlled by the loose mode and strict mode. The strict mode may become invalid in GaussDB. For example, if the source table has the default value **"0000-00-00"**, a table with the default value **"0000-00-00"** can be created in "no_zero_date" strict mode in GaussDB, which means that the strict mode is invalid. In MySQL, the table creation

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		will fail because it is controlled by the strict mode.
Truncate a partition.	ALTER TABLE [IF EXISTS] table_name truncate_clause;	For truncate_clause, the supported subitems are different: • M-compatible mode: TRUNCATE PARTITION { { ALL | partition_name [, ...] } | FOR (partition_value [, ...]) } [UPDATE GLOBAL INDEX] • MySQL: TRUNCATE PARTITION {partition_names | ALL}
Index name of a primary key	CREATE TABLE table_name (col_definition ,PRIMARY KEY [index_name] [USING method] ({ column_name | (expression) }[ASC | DESC] } [, ...]) index_parameters [USING method| COMMENT 'string'])	The index name created after being specified by a primary key in GaussDB is the index name specified by a user. In MySQL, the index name is **PRIMARY**.
Drop dependent objects.	DROP drop_type name CASCADE;	In GaussDB, **CASCADE** needs to be added to drop dependent objects. In MySQL, **CASCADE** is not required.
The NOT NULL constraint does not allow NULL values to be inserted.	CREATE TABLE t1(id int NOT NULL DEFAULT 8); INSERT INTO t1 VALUES(NULL); INSERT INTO t1 VALUES(1), (NULL),(2);	In MySQL loose mode, NULL is converted and data is successfully inserted. In MySQL strict mode, NULL values cannot be inserted. GaussDB does not support this feature. NULL values cannot be inserted in loose or strict mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
The CHECK constraint takes effect.	CREATE TABLE	• CREATE TABLE that contains the CHECK constraint takes effect in MySQL 8.0. MySQL 5.7 parses the syntax but the syntax does not take effect. GaussDB synchronizes this function of MySQL 8.0, and the GaussDB CHECK constraint can reference other columns, but MySQL cannot. • A maximum of 32767 CHECK constraints can be added to a table in GaussDB.
The **algorithm** and **lock** options of an index do not take effect.	CREATE INDEX ... DROP INDEX ...	Currently, the index options **algorithm_option** and **lock_option** in the CREATE/DROP INDEX statement in M-compatible mode are supported only in syntax. No error is reported during creation, but they do not take effect.
The storage of hash partitions and subpartitions in CREATE TABLE in GaussDB is different from that in MySQL.	CREATE TABLE	In GaussDB, the hash functions used by hash partitioned tables and subpartitioned tables in the CREATE TABLE statement are different from those in MySQL. Therefore, the storage of hash partitioned tables and subpartitioned tables is different from that in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Partitioned table index	CREATE INDEX	• GaussDB partitioned table indexes are classified into local and global indexes. A local index is bound to a specific partition, and a global index corresponds to the entire partitioned table. • For details about how to create local and global indexes and their default rules, see **CREATE INDEX**. For example, if you create a unique index on a non-partition key, the index is created as a global index by default. • MySQL does not have global indexes. In GaussDB, if the partitioned table index is a global index, the global index is not updated by default when operations such as DROP, TRUNCATE, and EXCHANGE are performed on table partitions. As a result, the global index becomes invalid and cannot be selected in subsequent statements. To avoid this problem, you are advised to explicitly specify the UPDATE GLOBAL INDEX clause at the end of the partition syntax or set the global GUC parameter **enable_gpi_auto_update** to **true** (recommended) so that global indexes can be automatically updated during partition operations. You can also set the global GUC parameter **enable_drop_truncate_partition_keep_gpi** to **true** so that global indexes are

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		automatically updated when partitions are dropped or truncated.
If the table is partitioned by key in the CREATE/ALTER TABLE statement, algorithms cannot be specified. Input parameters of some partition definition do not support expressions.	CREATE TABLE and ALTER TABLE	GaussDB: If the table is partitioned by key in the CREATE/ALTER TABLE statement, algorithms cannot be specified. The syntaxes that do not support expressions as input parameters are as follows: • PARTITION BY HASH() • PARTITION BY KEY() • VALUES LESS THAN()
Partitioned tables do not support LINEAR/KEY hash.	CREATE TABLE ... PARTITION ...	GaussDB: Partitioned tables do not support LINEAR/KEY hash.
The CHECK and AUTO_INCREMENT syntaxes cannot be used in the same column.	CREATE TABLE	The column using CHECK does not take effect in MySQL 5.7. When both CHECK and AUTO_INCREMENT are used on the same column, only AUTO_INCREMENT takes effect. However, GaussDB reports an error.
Delete dependent tables.	DROP TABLE	In GaussDB, CASCADE must be added to delete dependent tables. In MySQL, CASCADE is not required.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Options related to table definition.	CREATE TABLE ... and ALTER TABLE ...	• GaussDB does not support the following options: **AVG_ROW_LENGTH**, **CHECKSUM**, **COMPRESSION**, **CONNECTION**, **DATA DIRECTORY**, **INDEX DIRECTORY**, **DELAY_KEY_WRITE**, **ENCRYPTION**, **INSERT_METHOD**, **KEY_BLOCK_SIZE**, **MAX_ROWS**, **MIN_ROWS**, **PACK_KEYS**, **PASSWORD**, **STATS_AUTO_RECALC**, **STATS_PERSISTENT**, and **STATS_SAMPLE_PAGES**. • The following options do not report errors in GaussDB and do not take effect: **ENGINE** and **ROW_FORMAT**.
The GENERATED ALWAYS AS statement cannot reference columns generated by GENERATED ALWAYS AS.	Generated Always AS	In GaussDB, the GENERATED ALWAYS AS statement cannot reference columns generated by GENERATED ALWAYS AS, but it can in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Alter table names.	ALTER TABLE tbl_name RENAME [TO | AS | =] new_tbl_name;	• The ALTER RENAME syntax in GaussDB supports only the function of changing the table name and cannot be coupled with other function operations. • In GaussDB, only the old table name column supports the schema.table_name format, and the new and old table names belong to the same schema. • GaussDB does not support renaming of old and new tables across schemas. However, if you have the permission, you can modify the names of tables in other schemas in the current schema.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Support the CREATE VIEW AS SELECT syntax.	CREATE VIEW table_name AS query;	• For the following types, the query using the CREATE VIEW view_name AS query syntax cannot contain calculation operations (such as function call and calculation using operators): – BINARY[(n)] – BOOLEAN/BOOL – VARBINARY(n) – CHAR[(n)] – VARCHAR(n) – TIME[(p)] – DATETIME[(p)] – TIMESTAMP[(p)] – BIT[(n)] – NUMERIC[(p[,s])] – DECIMAL[(p[,s])] – DEC[(p[,s])] – FIXED[(p[,s])] – FLOAT4[(p, s)] – FLOAT8[(p,s)] – FLOAT[(p)] – REAL[(p, s)] – FLOAT[(p, s)] – DOUBLE[(p,s)] – DOUBLE PRECISION[(p,s)] – TEXT – TINYTEXT – MEDIUMTEXT – LONGTEXT – BLOB – TINYBLOB – MEDIUMBLOB – LONGBLOB • In the simple query scenario, an error message is displayed for the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		preceding calculation operations in M-compatible mode. For example:
		m_db=# CREATE TABLE TEST (salary int(10));
		CREATE TABLE
		
		m_db=# INSERT INTO TEST VALUES(8000);
		INSERT 0 1
		
		m_db=# CREATE VIEW view1 AS SELECT salary/10 as te FROM TEST;
		ERROR: Unsupported type numeric used with expression in CREATE VIEW statement.
		
		m_db=# CREATE VIEW view2 AS SELECT sec_to_time(salary) as te FROM TEST;
		ERROR: Unsupported type time used with expression in CREATE VIEW statement.
		• In non-simple query scenarios such as composite query and subquery, the calculation operations of the preceding types in M-compatible mode are different from those in MySQL. In M-compatible mode, the data type column precision attribute of the created table is not retained.

Range of index names that can be duplicated	CREATE TABLE, CREATE INDEX	In MySQL, an index name is unique in a table. Different tables can have the same index name. In M-compatible mode, the index name must be unique in the same schema. In M-compatible mode, the same rules apply to constraints and keys that automatically create indexes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
View dependency differences	CREATE VIEW and ALTER TABLE	In MySQL, view storage records only the table name, column name, and database name of the target table, but does not record the unique identifier of the target table. GaussDB parses the SQL statement used for creating a view and stores the unique identifier of the target table. Therefore, the differences are as follows: • In MySQL, you can modify the data type of a column on which a view depends because the view is unaware of the modification of the target table. In GaussDB, such modification is forbidden and the attempt will fail. • In MySQL, you can rename a column on which a view depends because the view is unaware of the modification of the target table, but the view cannot be queried after the operation. In GaussDB, each column precisely stores the unique identifier of the corresponding table and column. Therefore, the column name in the table can be modified successfully without changing the column name in the view. In addition, the view can be queried after the operation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Foreign key differences	CREATE TABLE	• In GaussDB, FOREIGN KEY constraints are insensitive to types. If the data types of the columns in the main and child tables are implicitly converted, foreign keys can be created. In MySQL, FOREIGN KEY constraints are sensitive to types. If the column types of the two tables are different, foreign keys cannot be created. • MySQL does not allow you to modify the data type or name of a table column where the foreign key of the column is located by running **MODIFY COLUMN** or **CHANGE COLUMN**, but GaussDB supports such operation.
Differences in index ascending and descending orders	CREATE INDEX	In MySQL 5.7, **ASC | DESC** is parsed but ignored, and the default behavior is **ASC**. In MySQL 8.0 and GaussDB, **ASC | DESC** is parsed and takes effect.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

Setting default values of columns	CREATE TABLE and ALTER TABLE	• For MySQL 5.7, only the default value without parentheses is supported. MySQL 8.0 and GaussDB support default values in parentheses.
		-- GaussDB
		m_db=# DROP TABLE IF EXISTS t1, t2;
		DROP TABLE
		m_db=# CREATE TABLE t1(a DATETIME DEFAULT NOW());
		CREATE TABLE
		m_db=# CREATE TABLE t2(a DATETIME DEFAULT (NOW()));
		CREATE TABLE
		
		-- MySQL 5.7
		mysql> DROP TABLE IF EXISTS t1, t2;
		Query OK, 0 rows affected (0.04 sec)
		
		mysql> CREATE TABLE t1(a DATETIME DEFAULT NOW());
		Query OK, 0 rows affected (0.04 sec)
		
		mysql> CREATE TABLE t2(a DATETIME DEFAULT (NOW()));
		ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '(NOW()))' at line 1
		
		-- MySQL 8.0
		mysql> DROP TABLE IF EXISTS t1, t2;
		Query OK, 0 rows affected (0.17 sec)
		
		mysql> CREATE TABLE t1(a DATETIME DEFAULT NOW());
		Query OK, 0 rows affected (0.19 sec)
		
		mysql> CREATE TABLE t2(a DATETIME DEFAULT (NOW()));
		Query OK, 0 rows affected (0.20 sec)
		• In MySQL, when specifying default values for BLOB, TEXT, and JSON data types, you must add parentheses to the default values. In GaussDB, you do not need to add parentheses when specifying default values

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		for the preceding data types.
		● When the default value is specified, GaussDB does not check whether the default value overflows. When the default value without parentheses is specified in MySQL, MySQL checks whether the default value overflows. When the default value with parentheses is specified, MySQL does not check whether the default value overflows.
		● In GaussDB, time constants starting with DATE, TIME, or TIMESTAMP can be used to specify default values for columns. In MySQL, when time constants starting with DATE, TIME, or TIMESTAMP are used to specify default values for columns, parentheses must be added to the default values.
		-- GaussDB
		m_db=# DROP TABLE IF EXISTS t1, t2;
		DROP TABLE
		m_db=# CREATE TABLE t1(a TIMESTAMP DEFAULT TIMESTAMP '2000-01-01 00:00:00');
		CREATE TABLE
		m_db=# CREATE TABLE t2(a TIMESTAMP DEFAULT (TIMESTAMP '2000-01-01 00:00:00'));
		CREATE TABLE
		
		-- MySQL 5.7
		mysql> DROP TABLE IF EXISTS t1, t2;
		Query OK, 0 rows affected (0.02 sec)
		
		mysql> CREATE TABLE t1(a TIMESTAMP DEFAULT TIMESTAMP '2000-01-01 00:00:00');
		ERROR 1067 (42000): Invalid default value for 'a'
		mysql> CREATE TABLE t2(a

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		TIMESTAMP DEFAULT (TIMESTAMP '2000-01-01 00:00:00'));
		ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '(TIMESTAMP '2000-01-01 00:00:00'))' at line 1
		
		-- MySQL 8.0
		mysql> DROP TABLE IF EXISTS t1, t2;
		Query OK, 0 rows affected (0.14 sec)
		
		mysql> CREATE TABLE t1(a TIMESTAMP DEFAULT TIMESTAMP '2000-01-01 00:00:00');
		ERROR 1067 (42000): Invalid default value for 'a'
		mysql> CREATE TABLE t2(a TIMESTAMP DEFAULT (TIMESTAMP '2000-01-01 00:00:00'));
		Query OK, 0 rows affected (0.19 sec)

3.3.2.7.5 DML

Table 3-160 DML syntax compatibility

Description	Syntax	Difference
DELETE supports deleting data from a specified partition (or subpartition).	DELETE	-
SELECT INTO syntax	SELECT	• In GaussDB, you can use SELECT INTO to create a table based on the query result. MySQL does not support this function. • In GaussDB, the SELECT INTO syntax does not support the query result that is obtained after the set operation of multiple queries is performed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

REPLACE INTO syntax	REPLACE	Difference between the initial values of the time type. For example:
		• MySQL is not affected by the strict or loose mode. You can insert time 0 into a table.
		`mysql> CREATE TABLE test(f1 TIMESTAMP NOT NULL, f2 DATETIME NOT NULL, f3 DATE NOT NULL);`
		`Query OK, 1 row affected (0.00 sec)`
		
		`mysql> REPLACE INTO test VALUES(f1, f2, f3);`
		`Query OK, 1 row affected (0.00 sec)`
		
		`mysql> SELECT * FROM test;`
		`+---------------------+---------------------+------------+`
		`| f1 | f2 | f3 |`
		`+---------------------+---------------------+------------+`
		`| 0000-00-00 00:00:00 | 0000-00-00 00:00:00 | 0000-00-00 |`
		`+---------------------+---------------------+------------+`
		`1 row in set (0.00 sec)`
		• The time 0 can be successfully inserted only when GaussDB is in loose mode.
		`gaussdb=# SET sql_mode = '';`
		`SET`
		`gaussdb=# CREATE TABLE test(f1 TIMESTAMP NOT NULL, f2 DATETIME NOT NULL, f3 DATE NOT NULL);`
		`CREATE TABLE`
		`gaussdb=# REPLACE INTO test VALUES(f1, f2, f3);`
		`REPLACE 0 1`
		`gaussdb=# SELECT * FROM test;`
		`f1 | f2 | f3`
		`---------------------+---------------------+------------`
		`0000-00-00 00:00:00 | 0000-00-00 00:00:00 | 0000-00-00`
		`(1 row)`
		In strict mode, the error "Incorrect Date/Time/Datetime/Timestamp/Year value" is reported.

SELECT supports multi-partition query.	SELECT	-
UPDATE supports multi-partition update.	UPDATE	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

Import data by using LOAD DATA.	LOAD DATA	When LOAD DATA is used to import data, GaussDB differs from MySQL in the following aspects:
		• The execution result of the LOAD DATA syntax is the same as that in MySQL strict mode. The loose mode is not adapted currently. For example, if the source data to be imported contains invalid characters, GaussDB reports an error, while MySQL reports an error in strict mode, or truncates the invalid characters and then import the data in loose mode.
		-- GaussDB
		m_db=# SET sql_mode='';
		SET
		m_db=# DROP TABLE IF EXISTS t1;
		DROP TABLE
		m_db=# CREATE TABLE t1(a text) CHARSET gbk;
		CREATE TABLE
		m_db=# SELECT unhex('1AFF1F') INTO OUTFILE '/tmp/3.data';
		COPY 1
		m_db=# LOAD DATA INFILE '/tmp/3.data' INTO TABLE t1 CHARACTER SET gbk;
		ERROR: invalid byte sequence for encoding "GBK": 0xff 0x1f
		CONTEXT: COPY t1, line 1.
		m_db=# DROP TABLE t1;
		DROP TABLE
		
		-- MySQL
		mysql> SET sql_mode='';
		Query OK, 0 rows affected (0.00 sec)
		mysql> DROP TABLE IF EXISTS t1;
		Query OK, 0 rows affected (0.02 sec)
		mysql> CREATE TABLE t1(a text) CHARSET gbk;
		Query OK, 0 rows affected (0.03 sec)
		mysql> SELECT unhex('1AFF1F') INTO OUTFILE '/tmp/3.data';
		Query OK, 1 row affected (0.00 sec)
		mysql> LOAD DATA INFILE '/tmp/3.data' INTO TABLE t1 CHARACTER SET gbk;
		Query OK, 1 row affected, 1 warning (0.01 sec)
		Records: 1 Deleted: 0 Skipped: 0 Warnings: 1
		mysql> SHOW warnings;
		+---------+------
		+----------
		----------+

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		<pre>| Level | Code | Message | +---------+------ +---------- ---------- ----------+ | Warning | 1366 | Incorrect string value: '\xFF\x1F' for column 'a' at row 1 | +---------+------ +---------- ---------- ----------+ 1 row in set (0.00 sec) mysql> SELECT hex(a) FROM t1; +--------+ | hex(a) | +--------+ | 1A | +--------+ 1 row in set (0.00 sec) mysql> DROP TABLE t1; Query OK, 0 rows affected (0.02 sec)</pre> • The **IGNORE** and **LOCAL** parameters are used only to ignore the conflicting rows when the imported data conflicts with the data in the table and to automatically fill default values for other columns when the number of columns in the file is less than that in the table. Other functions are not supported currently. • If the keyword **LOCAL** is specified and the file path is a relative path, the file is searched from the binary directory. If the keyword **LOCAL** is not specified and the file path is a relative path, the file is searched from the data directory. • LOAD DATA can only be used to import files from the server. • The **[(col_name_or_user_var [, col_name_or_user_var]...)]** parameter cannot be used to specify a column repeatedly. • The newline character specified by **[FIELDS TERMINATED BY 'string']**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		cannot be the same as the separator specified by **[LINES TERMINATED BY'string']**. ● If the data written to a table by running **LOAD DATA** cannot be converted to the data type of the table, an error is reported. ● The LOAD DATA SET expression does not support the calculation of a specified column name. ● LOAD DATA applies only to tables but not views. ● The default newline character of the file in Windows is different from that in Linux. LOAD DATA cannot identify this scenario and reports an error. You are advised to check the newline character at the end of lines in the file to be imported. ● If the number of columns in the source data is greater than that in the target table, GaussDB displays an alarm, while MySQL displays an alarm in the following scenarios and reports an error in other scenarios: – The strict mode is not enabled for **sql_mode**. – The **IGNORE** option is specified. – The **LOCAL** option is specified and the **REPLACE** option is not specified. ● When the source data file ends with a column delimiter, MySQL processes the file as if there is no new column after the delimiter, while GaussDB processes the file as if there is an empty

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		string column after the delimiter.
		-- Source data in **1.csv**
		1,2,
		
		-- GaussDB
		m_db=# DROP TABLE IF EXISTS t1;
		NOTICE: table "t1" does not exist, skipping
		DROP TABLE
		m_db=# CREATE TABLE t1(a char(30), b char(30), c char(30));
		CREATE TABLE
		m_db=# LOAD DATA infile '/tmp/1.csv' INTO TABLE t1 fields terminated by ',';
		COPY 1
		m_db=# SELECT c IS NULL, length(c) FROM t1;
		?column? | length
		----------+--------
		f | 0
		(1 row)
		
		-- MySQL
		mysql> DROP TABLE IF EXISTS t1;
		Query OK, 0 rows affected (0.06 sec)
		
		mysql> CREATE TABLE t1(a char(30), b char(30), c char(30));
		Query OK, 0 rows affected (0.04 sec)
		
		mysql> LOAD DATA infile '/tmp/1.csv' INTO TABLE t1 fields terminated by ',';
		ERROR 1261 (01000): Row 1 doesn't contain data for all columns

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
LIMIT differences	DELETE, SELECT, and UPDATE	The LIMIT clauses of each statement in GaussDB are different from those in MySQL. The maximum parameter value of LIMIT (of the BIG INT type) in GaussDB is **9223372036854775807**. If the actual value exceeds the number, an error is reported. In MySQL, the maximum value of LIMIT (of the unsigned LONGLONG type) is **18446744073709551615**. If the actual value exceeds the number, an error is reported. You can set a small value in LIMIT, which is rounded off during execution. The value cannot be a decimal in MySQL. In GaussDB, limit 0 is not allowed in the DELETE statement. In MySQL, limit 0 is allowed in the DELETE statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Difference in using backslashes (\)	INSERT	The usage of backslashes (\) can be determined by parameters in GaussDB and MySQL, but their default usages are different. In MySQL, the **NO_BACKSLASH_ESCAPES** parameter is used to determine whether backslashes (\) in character strings and identifiers are parsed as common characters or escape characters. By default, backslashes (\) are parsed as escape characters in character strings and identifiers. If **SET sql_mode='NO_BACKSLASH_ESCAPES';** is used, the backslashes (\) cannot be parsed as escape characters in strings and identifiers. In GaussDB, the **standard_conforming_strings** parameter is used to determine whether backslashes (\) in character strings and identifiers are parsed as common characters or escape characters. The default value is **on**, indicating that backslashes (\) are parsed as common text in common character string texts according to the SQL standard. If **SET standard_conforming_strings=off;** is used, backslashes (\) can be parsed as escape characters in character strings and identifiers.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
If the inserted value is less than the number of columns, MySQL reports an error while GaussDB supplements null values.	INSERT	In GaussDB, if the column list is not specified and the inserted value is less than the number of columns, values are assigned based on the column sequence when the table is created by default. If a column has a NOT NULL constraint, an error is reported. If no NOT NULL constraint exists and a default value is specified, the default value is added to the column. If no default value is specified, **null** is added.
The columns sorted in ORDER BY must be included in the columns of the result set.	SELECT	In GaussDB, when used with the GROUP BY clause, the columns to be sorted in ORDER BY must be included in the columns of the result set retrieved by the SELECT statement. When used with the DISTINCT keyword, the columns to be sorted in ORDER BY must be included in the columns of the result set retrieved by the SELECT statement.
Duplicate column names are allowed in the SELECT result.	SELECT	-
NATURAL JOIN in GaussDB is different from that in MySQL.	SELECT	In GaussDB, NATURAL [[LEFT | RIGHT] OUTER] JOIN allows you not to specify **LEFT | RIGHT**. If **LEFT | RIGHT** is not specified, **NATURAL OUTER JOIN** is **NATURAL JOIN**. You can use JOIN consecutively.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

Support of UPDATE or DELETE for ORDER BY and LIMIT	UPDATE/DELETE	When the ORDER BY and LIMIT clauses are used together with the UPDATE/DELETE clause, if the tuple to be updated or deleted is concurrently modified by another session, the system obtains the latest value of the sorted tuple from the session and performs matching again. If the condition is still met, the tuple is updated. During this process, MySQL obtains the sorting result again for update. However, GaussDB updates or deletes only the tuples that have been sorted by the session and involve concurrent update, which may cause data inconsistency.
		• Behavior in GaussDB:
		m_db=# CREATE TABLE t1(a INT, b INT);
		CREATE TABLE
		m_db=# CREATE TABLE t2(a INT, b INT);
		CREATE TABLE
		m_db=# INSERT INTO t1 VALUES(1,1),(2,2);
		INSERT 0 2
		-- Session one.
		m_db=# START TRANSACTION;
		START TRANSACTION
		-- The session one is updated first, but the transaction is not committed.
		m_db=# UPDATE t1 SET a=3 WHERE a=1;
		UPDATE 1
		-- Session two.
		m_db=# START TRANSACTION;
		START TRANSACTION
		-- The session two is updated after the session one is updated. The update is successful after the session one transaction ends.
		m_db=# UPDATE t1 SET b=4 ORDER BY a LIMIT 1;
		UPDATE 1
		-- The session one commits the transaction.
		m_db=# COMMIT;
		COMMIT
		-- Query data after the session two is updated successfully.
		m_db=# SELECT * FROM t1;
		a | b
		---+---
		3 | 4
		2 | 2
		(2 rows)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		• Behavior in MySQL:
		-- Session one.
		mysql> START TRANSACTION;
		Query OK, 0 rows affected (0.00 sec)
		-- The session one is updated first, but the transaction is not committed.
		mysql> UPDATE t1 SET a=3 WHERE a=1;
		Query OK, 1 row affected (0.00 sec)
		Rows matched: 1 Changed: 1 Warnings: 0
		-- Session two.
		mysql> START TRANSACTION;
		Query OK, 0 rows affected (0.00 sec)
		mysql> UPDATE t1 SET b=4 ORDER BY a LIMIT 1;
		-- The session two is updated after the session one is updated. The update is successful after the session one transaction ends.
		Query OK, 1 row affected (5.12 sec)
		Rows matched: 1 Changed: 1 Warnings: 0
		-- The session one commits the transaction.
		mysql> COMMIT;
		Query OK, 0 rows affected (0.00 sec)
		-- Query data after the session two is updated successfully.
		mysql> SELECT * FROM t1;
		+------+------+
		| a | b |
		+------+------+
		| 3 | 1 |
		| 2 | 4 |
		+------+------+
		2 rows in set (0.00 sec)

If the foreign key data type is timestamp or datetime, an error is reported for attempts to perform UPDATE or DELETE on a foreign table.	UPDATE/DELETE	If the foreign key data type is timestamp or datetime, an error is reported for attempts to perform UPDATE or DELETE on a foreign table, but such operations are allowed in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

Compatibility in terms of natural join and using	SELECT	• In GaussDB, join sequence is strictly from left to right. MySQL may adjust the sequence.
		• In GaussDB and MySQL, columns involving **join** in the left or right table cannot be ambiguous during natural join or using. (Generally, ambiguity is caused by duplicate names of columns in the left or right temporary table.) The join sequence differs in two databases, which may lead to different behaviors.
		– Behavior in GaussDB:
		m_regression=# CREATE TABLE t1(a int,b int);
		CREATE TABLE
		m_regression=# CREATE TABLE t2(a int,b int);
		CREATE TABLE
		m_regression=# CREATE TABLE t3(a int,b int);
		CREATE TABLE
		m_regression=# SELECT * FROM t1 JOIN t2;
		a | b | a | b
		---+---+---+---
		(0 rows)
		m_regression=# SELECT * FROM t1 JOIN t2 natural join t3; -- Failed. Duplicate contents exist in columns **a** and **b** of the temporary table obtained by **t1 join t2**. Therefore, there is ambiguity in natural join.
		ERROR: common column name "a" appears more than once in left table
		– Behavior in MySQL:
		mysql> SELECT * FROM t1 JOIN t2 NATURAL JOIN t3;
		Empty set (0.00 sec)
		mysql> SELECT * FROM (t1 join t2) NATURAL JOIN t3;
		ERROR 1052 (23000): Column 'a' in from clause is ambiguous

The WITH clause is compatible with MySQL 8.0.	SELECT, INSERT, UPDATE, and DELETE	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Compatibility in terms of join	SELECT	Commas (,) cannot be used as a way of join in GaussDB, but can be used in MySQL. GaussDB does not support use index for join.
If the column expression in the SELECT statement is a function expression or arithmetic expression, the column name in the query result is **?column?**.	SELECT	In GaussDB, if the column expression in the SELECT statement is a function expression or arithmetic expression, the column name in the query result is **?column?**. In MySQL, the name is the corresponding expression.
SELECT export file (into outfile)	SELECT ... INTO OUTFILE ...	In the file exported by using the SELECT INTO OUTFILE syntax, the display precision of values of the FLOAT, DOUBLE, and REAL types in GaussDB is different from that in MySQL. The syntax does not affect the import using COPY the values after import.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

Specify schema names and table names by using SELECT/UPDATE/ INSERT/REPLACE.	SELECT/UPDATE/ INSERT/REPLACE	• When the SELECT statement is used to the projection column, MySQL supports the three-segment format of *schema name.table alias.column name*, but GaussDB does not.
		m_db=# CREATE SCHEMA test;
		CREATE SCHEMA
		m_db=# CREATE TABLE test.t1(a int);
		CREATE TABLE
		m_db=# SELECT test.alias1.a FROM t1 alias1;
		ERROR: invalid reference to FROM-clause entry for table "alias1"
		LINE 1: SELECT test.alias1.a FROM t1 alias1;
		^
		HINT: There is an entry for table "alias1", but it cannot be referenced from this part of the query.
		CONTEXT: referenced column: a
		• The three-segment format for UPDATE/REPLACE SET is *database.table.column* in MySQL, and is *table.column.field* in GaussDB, where *field* indicates the attribute in the specified composite type.
		• For INSERT ... SET, MySQL supports *column*, *table.column*, and *database.table.column*. GaussDB supports only *column* and does not support *table.column* and *database.table.column*.
		• For INSERT... SET, you can reference column names and expressions that contain column names on the right of the SET clause in MySQL. GaussDB does not support this operation.
		– Behavior in GaussDB:
		m_db=# CREATE TABLE t2 (a int default 3, b int default 5);
		CREATE TABLE
		
		m_db=# INSERT INTO t2 SET a = b + 1;
		ERROR: Column "b" does not exist.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		LINE 1: INSERT INTO t2 SET a = b + 1;
		^
		HINT: There is a column named "b" in table "t2", but it cannot be referenced from this part of the query.
		
		m_db=# INSERT INTO t2 SET a = b + 1, b = 0;
		ERROR: Column "b" does not exist.
		LINE 1: INSERT INTO t2 SET a = b + 1, b = 0;
		^
		HINT: There is a column named "b" in table "t2", but it cannot be referenced from this part of the query.
		
		m_db=# INSERT INTO t2 SET b = 0, a = b + 1;
		ERROR: Column "b" does not exist.
		LINE 1: INSERT INTO t2 SET b = 0, a = b + 1;
		^
		HINT: There is a column named "b" in table "t2", but it cannot be referenced from this part of the query.
		
		m_db=# INSERT INTO t2 SET a = a + 1;
		ERROR: Column "a" does not exist.
		LINE 1: INSERT INTO t2 SET a = a + 1;
		^
		HINT: There is a column named "a" in table "t2", but it cannot be referenced from this part of the query.
		
		m_db=# DROP TABLE t2;
		DROP TABLE
		– Behavior in MySQL:
		mysql> CREATE TABLE t2 (a int default 3, b int default 5);
		Query OK, 0 rows affected (0.07 sec)
		
		mysql> INSERT INTO t2 SET a = b + 1;
		Query OK, 1 row affected (0.02 sec)
		
		mysql> SELECT * FROM t2;
		+------+------+
		| a | b |
		+------+------+
		| 6 | 5 |
		+------+------+
		1 row in set (0.00 sec)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		mysql> INSERT INTO t2 SET a = b + 1, b = 0;
		Query OK, 1 row affected (0.00 sec)
		
		mysql> SELECT * FROM t2;
		+------+------+
		| a | b |
		+------+------+
		| 6 | 5 |
		| 6 | 0 |
		+------+------+
		2 rows in set (0.00 sec)
		
		mysql> INSERT INTO t2 SET b = 0, a = b + 1;
		Query OK, 1 row affected (0.00 sec)
		
		mysql> SELECT * FROM t2;
		+------+------+
		| a | b |
		+------+------+
		| 6 | 5 |
		| 6 | 0 |
		| 1 | 0 |
		+------+------+
		3 rows in set (0.00 sec)
		
		mysql> INSERT INTO t2 SET a = a + 1;
		Query OK, 1 row affected (0.02 sec)
		
		mysql> SELECT * FROM t2;
		+------+------+
		| a | b |
		+------+------+
		| 6 | 5 |
		| 6 | 0 |
		| 1 | 0 |
		| 4 | 5 |
		+------+------+
		4 rows in set (0.00 sec)
		
		mysql> DROP TABLE t2;
		Query OK, 4 rows affected (0.40 sec)

The execution sequence of UPDATE SET is different from that of MySQL.	UPDATE ... SET	In MySQL, UPDATE SET is performed in sequence. The results of UPDATE at the front affect subsequent results of UPDATE, and the same column can be updated for multiple times. In GaussDB, all related data is obtained first, and then UPDATE is performed on the data at a time. The same column cannot be updated for multiple times.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
IGNORE feature	UPDATE/DELETE/ INSERT	The execution process in MySQL is different from that in GaussDB. Therefore, the number and information of generated warnings may be different.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

SHOW COLUMNS syntax	SHOW	• User permission verification is different from that of MySQL.
		– In GaussDB, you need the USAGE permission on the schema of a specified table and table-level or column-level permissions on the specified table. Only information about columns with the SELECT, INSERT, UPDATE, REFERENCES, and COMMENT permissions is displayed.
		– In MySQL, you need table-level or column-level permissions on a specified table. Only information about columns with the SELECT, INSERT, UPDATE, REFERENCES, and COMMENT permissions is displayed.
		• When the LIKE and WHERE clauses involve string comparison operations, the collation is different from that in MySQL.
		– **utf8_general_ci** is used in MySQL.
		– The **collation_connection** of the current client is used as the collation in GaussDB.
		In GaussDB, you are advised not to select columns other than the returned fields in the WHERE clause. Otherwise, unexpected errors may occur.
		`-- Expected error`
		`m_db=# SHOW FULL COLUMNS FROM t02 WHERE `b`='pri';`
		`ERROR: Column "b" does not exist.`
		`LINE 1: SHOW FULL COLUMNS FROM t02 WHERE `b`='pri';`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		^ -- Unexpected error m_db=# SHOW FULL COLUMNS FROM t02 WHERE `c`='pri'; ERROR: input of anonymous composite types is not implemented LINE 1: SHOW FULL COLUMNS FROM t02 WHERE `c`='pri'; ^
SHOW CREATE DATABASE syntax	SHOW	User permission verification is different from that of MySQL. ● In GaussDB, you need the USAGE permission on a specified schema. ● In MySQL, you need database-level permissions (except GRANT OPTION and USAGE), table-level permissions (except GRANT OPTION), or column-level permissions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
SHOW CREATE TABLE syntax	SHOW	• User permission verification is different from that of MySQL. – In GaussDB, you need the USAGE permission on the schema where a specified table is located and table-level permissions on the specified table. – Table-level permissions (except GRANT OPTION) of the specified table are required in MySQL. • The returned statements for table creation are different from those in MySQL. – In GaussDB, indexes are returned as CREATE INDEX statements. In MySQL, indexes are returned as CREATE TABLE statements. In GaussDB, the range of optional parameters supported by the CREATE INDEX syntax is different from that supported by the CREATE TABLE syntax. Therefore, some indexes cannot be created in CREATE TABLE statements. – In GaussDB, the **ENGINE** and **ROW_FORMAT** options of CREATE TABLE are adapted only for the syntax but do not take effect. Therefore, they are not displayed in the returned statements for table creation. • These statements are compatible with MySQL only after the compatibility parameter **m_format_dev_version** is set to **'s2'**. The compatibility parameter takes effect by

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		changing the positions of column comments, table comments, **ON COMMIT** option for global temporary tables, PRIMARY KEY and UNIQUE constraints (where the **USING INDEX TABLESPACE** option is no longer displayed), and index comments.
SHOW CREATE VIEW syntax	SHOW	• User permission verification is different from that of MySQL. – In GaussDB, you need the USAGE permission on the schema where a specified view is located and table-level permissions on the specified view. – In MySQL, you need the table-level SELECT and table-level SHOW VIEW permissions on the specified view. • The returned statements for view creation are different from those in MySQL. If a view is created in the format of **SELECT * FROM** *tbl_name*, * is not expanded in GaussDB but expanded in MySQL. • The *character_set_client* and *collation_connection* columns in the returned result are different from those in MySQL. – The session values of system variables *character_set_client* and *collation_connection* are displayed during view creation in MySQL – Related metadata is not recorded in GaussDB and **NULL** is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
SHOW PROCESSLIST syntax	SHOW	In GaussDB, the column content and case in the query result of this command are the same as those in the information_schema.processlist view. In MySQL, the column content and case may be different. • In GaussDB, common users can access only their own thread information. Users with the SYSADMIN permission can access thread information of all users. • In MySQL, common users can access only their own thread information. Users with the PROCESS permission can access thread information of all users.
SHOW [STORAGE] ENGINES	SHOW	In GaussDB, the column content and case of the query result of this command are the same as those in the information_schema.engines view. In MySQL, they may be different from those in the view. The query results of this command are different in MySQL and GaussDB because the databases have different storage engines.
SHOW [SESSION] STATUS	SHOW	In GaussDB, the column content and case of the query result of this command are the same as those in the information_schema.session_status view. In MySQL, they may be different from those in the view. Currently, GaussDB supports only **Threads_connected** and **Uptime**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
SHOW [GLOBAL] STATUS	SHOW	In GaussDB, the column content and case of the query result of this command are the same as those in the information_schema.global_status view. In MySQL, they may be different from those in the view. Currently, GaussDB supports only **Threads_connected** and **Uptime**.
SHOW INDEX	SHOW	• User permission verification is different from that of MySQL. – In GaussDB, you need the USAGE permission on a specified schema and table-level or column-level permissions on a specified table. – In MySQL, you need table-level (except GRANT OPTION) or column-level permission on the specified table. • Temporary tables in GaussDB are stored in independent temporary schemas. When using the FROM or IN db_name condition to display the index information of a specified temporary table, you must specify **db_name** as the schema where the temporary table is located. Otherwise, the system displays a message indicating that the temporary table does not exist. This is different from MySQL in some cases.
SHOW SESSION VARIABLES	SHOW	In GaussDB, the column content and case of the query result are the same as those in the information_schema.session_variables view. In MySQL, they may be different from those in the view.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
SHOW GLOBAL VARIABLES	SHOW	In GaussDB, the column content and case of the query result are the same as those in the information_schema.global_variables view. In MySQL, they may be different from those in the view.
SHOW CHARACTER SET	SHOW	In GaussDB, the column content and case of the query result are the same as those in the information_schema.character_sets view. In MySQL, they may be different from those in the view.
SHOW COLLATION	SHOW	In GaussDB, the column content and case of the query result are the same as those in the information_schema.collations view. In MySQL, they may be different from those in the view.
EXCEPT Syntax	SELECT	-
SELECT supports the STRAIGHT_JOIN syntax.	SELECT	The execution plans generated in the multi-table JOIN scenarios in GaussDB may be different from those in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
SHOW TABLES	SHOW	• The LIKE behavior is different. For details, see "LIKE" in **Operators**. • The WHERE expression behavior is different. For details, see "WHERE" in GaussDB. • In GaussDB, permissions on tables and databases must be assigned to users separately. The database to be queried must be available to users on the SHOW SCHEMAS. Users must have permissions on both tables and databases. MySQL can be accessed as long as you have table permissions. • In GaussDB, the verification logic preferentially checks whether a schema exists and then checks whether the current user has the permission on the schema, which is different from that in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
SHOW TABLE STATUS	SHOW	• In GaussDB, the syntax displays data depending on the tables view under information_schema. In MySQL, the tables view specifies tables. • In GaussDB, permissions on tables and databases must be assigned to users separately. The database to be queried must be available to users on the SHOW SCHEMAS. Users must have permissions on both tables and databases. MySQL can be accessed as long as you have table permissions. • In GaussDB, the verification logic preferentially checks whether a schema exists and then checks whether the current user has the permission on the schema, which is different from that in MySQL.
HAVING syntax	SELECT	In GaussDB, HAVING can only reference columns in the GROUP BY clause or columns used in aggregate functions. In addition, MySQL allows HAVING to reference SELECT columns in the list and columns in external subqueries.

SELECT followed by a row expression	SELECT	In MySQL, SELECT cannot be followed by a row expression, but in GaussDB, SELECT can be followed by a row expression.
		Behavior in MySQL:
		`mysql> SELECT row(1,2);`
		`ERROR 1241 (21000): Operand should contain 1 column(s)`
		Behavior in GaussDB:
		`m_db=# SELECT row(1,2);`
		` row(1,2)`
		`----------`
		` (1,2)`
		`(1 row)`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

SELECT FOR SHARE/FOR UPDATE/ LOCK IN SHARE MODE	SELECT	• The FOR SHARE/FOR UPDATE/LOCK IN SHARE MODE and UNION/EXCEPT/ DISTINCT/GROUP BY/ HAVING clauses cannot be used together in GaussDB. They can be used together in MySQL 5.7 (except in the FOR SHARE/EXCEPT syntax) and MySQL 8.0.
		• When a locking clause is used together with the LEFT/ RIGHT [OUTER] JOIN clause, the LEFT JOIN cannot be used to lock the right table, and the RIGHT JOIN clause cannot be used to lock the left table. In MySQL, tables on both sides of JOIN can be locked at the same time.
		• When a lock clause is used together with the ORDER BY clause, if GaussDB detects that the tuple to be locked is concurrently modified by another session, it obtains the latest value of the sorted tuple from the session. During this process, MySQL obtains the sorting result again.
		• Behavior in GaussDB:
		gaussdb=# CREATE TABLE t1(a INT,b INT);
		CREATE TABLE
		gaussdb=# INSERT INTO t1 VALUES(1,1),(2,2);
		INSERT 0 2
		-- Session one.
		gaussdb=# START TRANSACTION;
		START TRANSACTION
		-- The session one is updated first, but the transaction is not committed.
		gaussdb=# UPDATE t1 SET a=3 WHERE a=1;
		UPDATE 1
		-- The session two starts a transaction.
		gaussdb=# START TRANSACTION;
		START TRANSACTION
		-- The session two is queried after the session one is updated. The query is successful after the session one transaction ends.
		gaussdb=# SELECT * FROM t1 ORDER

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		BY a FOR UPDATE;
		commit a | b
		---+---
		3 | 1
		2 | 2
		(2 rows)
		-- The session one commits the transaction.
		gaussdb=# COMMIT;
		COMMIT
		• Behavior in MySQL:
		mysql> CREATE TABLE t1(a INT,b INT);
		Query OK, 0 rows affected (0.00 sec)
		mysql> INSERT INTO t1 VALUES(1,1),(2,2);
		Query OK, 2 rows affected (0.00 sec)
		Records: 2 Duplicates: 0 Warnings: 0
		-- Session one.
		mysql> START TRANSACTION;
		Query OK, 0 rows affected (0.00 sec)
		-- The session one is updated first, but the transaction is not committed.
		mysql> UPDATE t1 SET a=3 WHERE a=1;
		Query OK, 1 row affected (0.00 sec)
		Rows matched: 1 Changed: 1 Warnings: 0
		-- The session two starts a transaction.
		mysql> START TRANSACTION;
		Query OK, 0 rows affected (0.00 sec)
		-- The session two is queried after the session one is updated. The query is successful after the session one transaction ends.
		mysql> SELECT * FROM t1 ORDER BY a FOR UPDATE;
		+------+------+
		| a | b |
		+------+------+
		| 3 | 1 |
		| 2 | 4 |
		+------+------+
		2 rows in set (0.00 sec)
		-- The session one commits the transaction.
		mysql> COMMIT;
		Query OK, 0 rows affected (0.00 sec)
		• In MySQL, multiple lock clauses cannot be specified for the same table, while GaussDB supports this operation and the strongest lock will take effect.
		-- GaussDB
		m_db=# DROP TABLE IF EXISTS t1;
		DROP TABLE
		
		m_db=# CREATE TABLE t1(a INT, b INT);
		CREATE TABLE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		m_db=# INSERT INTO t1 VALUES(1,2);
		INSERT 0 1
		
		m_db=# SELECT * FROM t1 FOR UPDATE OF t1 LOCK IN SHARE MODE;
		a | b
		---+---
		1 | 2
		(1 row)
		
		m_db=# DROP TABLE t1;
		DROP TABLE
		
		-- MySQL
		mysql> DROP TABLE IF EXISTS t1;
		Query OK, 0 rows affected (0.05 sec)
		
		mysql> CREATE TABLE t1(a INT, b INT);
		Query OK, 0 rows affected (0.09 sec)
		
		mysql> INSERT INTO t1 VALUES(1,2);
		Query OK, 1 row affected (0.01 sec)
		
		mysql> SELECT * FROM t1 FOR UPDATE OF t1 LOCK IN SHARE MODE;
		ERROR 3569 (HY000): Table t1 appears in multiple locking clauses.
		
		mysql> DROP TABLE t1;
		Query OK, 0 rows affected (0.05 sec)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

SELECT syntax	SELECT	• In GaussDB, the HAVING clause must be used together with the GROUP BY clause or aggregate functions. In MySQL, users can specify only the HAVING clause in a query statement.
		• In GaussDB, HAVING can only reference columns in the GROUP BY clause or columns used in aggregate functions. In addition, MySQL allows HAVING to reference SELECT columns in the list and columns in external subqueries.
		• In GaussDB and MySQL 5.7, GROUP BY WITH ROLLUP cannot be used together with the DISTINCT or ORDER BY clause. However, MySQL 8.0 supports this operation.
		• In GaussDB, when an empty table is queried by using the specified WITH ROLLUP statement, the query result is an empty row. In contrast, the query result in MySQL is empty.
		• In GaussDB, a table alias with the column name can be specified by using the FROM clause. In MySQL 5.7, a table alias with the column name cannot be specified. In MySQL 8.0, it is allowed only in a subquery.
		`-- GaussDB`
		`m_db=# DROP TABLE IF EXISTS t1;`
		`DROP TABLE`
		`m_db=# CREATE TABLE t1(a INT, b INT);`
		`CREATE TABLE`
		`m_db=# INSERT INTO t1 VALUES(1,2);`
		`INSERT 0 1`
		`m_db=# SELECT * FROM t1 t2(a, b);`
		` a | b`
		`---+---`
		` 1 | 2`
		`(1 row)`
		
		`m_db=# SELECT * FROM (SELECT *`

Description	Syntax	Difference

		FROM t1) t2(a, b);
		 a | b
		---+---
		 1 | 2
		(1 row)
		
		-- MySQL 5.7
		mysql> DROP TABLE IF EXISTS t1;
		Query OK, 0 rows affected, 1 warning (0.00 sec)
		
		mysql> CREATE TABLE t1(a INT, b INT);
		Query OK, 0 rows affected (0.03 sec)
		
		mysql> INSERT INTO t1 VALUES(1,2);
		Query OK, 1 row affected (0.01 sec)
		
		mysql> SELECT * FROM t1 t2(a, b);
		ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '(a, b)' at line 1
		mysql> SELECT * FROM (SELECT * FROM t1) t2(a, b);
		ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '(a, b)' at line 1
		
		-- MySQL 8.0
		mysql> DROP TABLE IF EXISTS t1;
		Query OK, 0 rows affected (0.10 sec)
		
		mysql> CREATE TABLE t1(a INT, b INT);
		Query OK, 0 rows affected (0.18 sec)
		
		mysql> INSERT INTO t1 VALUES(1,2);
		Query OK, 1 row affected (0.03 sec)
		
		mysql> SELECT * FROM t1 t2(a, b);
		ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '(a, b)' at line 1
		mysql> SELECT * FROM (SELECT * FROM t1) t2(a, b);
		+------+------+
		| a | b |
		+------+------+
		| 1 | 2 |
		+------+------+
		1 row in set (0.00 sec)
		• If a query statement does not contain the FROM clause, GaussDB supports the WHERE clause, which is the same as that in MySQL

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		8.0. MySQL 5.7 does not support the WHERE clause. -- GaussDB m_db=# SELECT 1 WHERE true; 1 --- 1 (1 row) -- MySQL 5.7 mysql> SELECT 1 WHERE true; ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'where true' at line 1 -- MySQL 8.0 mysql> SELECT 1 WHERE true; +---+ | 1 | +---+ | 1 | +---+ 1 row in set (0.00 sec)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

Differences in assigning values to user variables in SELECT projection columns	SELECT @var_name := expr	• In MySQL, user-defined variable names can be escaped using escape characters or double quotation marks, but this feature is not supported in GaussDB.
		Variable names enclosed in single quotation marks cannot contain other single quotation marks. For example, @'', @''', and @'\'' are not supported. During parsing, the single quotation marks (') cannot be matched or an error will be reported. For example:
		-- An error is reported during parsing.
		m_db=# SET @''' = 1;
		ERROR: syntax error at or near "@"
		LINE 1: SET @''' = 1;
		
		-- The single quotation marks (') cannot be matched during parsing.
		m_db=# SET @'\'' = 1;
		m_db'#
		Variable names enclosed in double quotation marks cannot contain double quotation marks ("). For example, @"", @"""" and @"\"" are not supported. The double quotation marks cannot be matched or an error will be reported during parsing. For example:
		-- An error is reported during parsing.
		m_db=# SET @"""" = 1;
		ERROR: syntax error at or near "@"
		LINE 1: SET @"""" = 1;
		
		-- The double quotation marks (") cannot be matched during parsing.
		m_db=# SET @"\"" = 1;
		m_db"#
		The variable name enclosed by backquotes cannot contain backquotes. For example, @````, @`````, and @`\`` are not supported. During parsing, the backquotes (`) cannot be matched or an error will be reported. For example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		-- An error is reported during parsing. m_db=# SET @```` = 1; ERROR: syntax error at or near "@" LINE 1: SET @```` = 1; -- The backquotes (`) cannot be matched during parsing. m_db=# SET @`\`` = 1; m_db`# • When an SQL statement simultaneously assigns values to and reads user variables, the evaluation order of the variables is undefined, which may lead to differences in the final results compared to MySQL. The following is an example of assigning a value to a user variable and using the user variable as the parameter of the MAX function: -- GaussDB m_db=# SELECT @a := 2; @a ---- 2 (1 row) m_db=# SELECT MAX(@a), @a := 3; max | @a -----+---- 2 | 3 (1 row) m_db=# SELECT @a; @a ---- 3 (1 row) -- MySQL 5.7 mysql> SELECT @a := 2; +------+ | @a | +------+ | 2 | +------+ 1 row in set (0.00 sec) mysql> SELECT max(@a), @a := 3; +---------+---------+ | max(@a) | @a := 3 | +---------+---------+ | 3 | 3 | +---------+---------+ 1 row in set (0.00 sec) mysql> SELECT @a; +------+

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		<pre>| @a |
		+------+
		| 3 |
		+------+
		1 row in set (0.00 sec)</pre>
		• When a user variable is set to a subquery or CASE WHEN expression and the final result is a subquery expression, the value of the user-defined variable is the same as the result queried directly in GaussDB. However, the intermediate calculation result is used in MySQL. Therefore, the user variable values in GaussDB and MySQL may be different.
		<pre>CREATE TABLE t1 (a double(6,3));
		INSERT INTO t1 VALUES(1.1);
		
		-- In GaussDB, the value of the user variable is the same as that queried directly.
		m_db=# SELECT @a := (SELECT a FROM t1);
		@a
		------- 	
		1.100
		(1 row)
		
		m_db=# SELECT @a;
		@a
		------- 	
		1.100
		(1 row)
		
		-- In MySQL 5.7, the value of a user variable is inconsistent with that queried directly.
		mysql> SELECT @a := (SELECT a FROM t1);
		+--------------------------+
		| @a := (SELECT a FROM t1) |
		+--------------------------+
		| 1.100 |
		+--------------------------+
		1 row in set (0.00 sec)
		
		mysql> SELECT @a;
		+------+
		| @a |
		+------+
		| 1.1 |
		+------+
		1 row in set (0.00 sec)</pre>
		• When the ORDER BY clause references a column related to a user variable expression,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		the result of the user variable in GaussDB may differ from that in MySQL, depending on the execution plan.
		CREATE TABLE t1 (a int);
		INSERT INTO t1 VALUES (1),(3),(2);
		
		-- GaussDB
		m_db=# SELECT @a := a FROM t1
		ORDER BY a;
		@a
		---- 	
		1
		2
		3
		(3 rows)
		
		m_db=# SELECT @a;
		@a
		---- 	
		2
		(1 row)
		
		-- MySQL 5.7
		mysql> SELECT @a := a FROM t1
		ORDER BY a;
		+---------+
		| @a := a |
		+---------+
		| 1 |
		| 2 |
		| 3 |
		+---------+
		3 rows in set (0.00 sec)
		
		mysql> SELECT @a;
		+------+
		| @a |
		+------+
		| 3 |
		+------+
		1 row in set (0.00 sec)
		• *@var_name* **:= expr** serves as an input parameter for the IF/IFNULL/COALESCE function. When this parameter is not selected, MySQL does not assign a value to the user variable, whereas GaussDB assigns a value to the user variable, resulting in inconsistent implementation. The following uses the IF function as an example:
		-- In GaussDB, a value is assigned to *@var* regardless of whether the *@var***:=val** parameter is selected.
		m_db=# SELECT IF (1, @a:=1, 2);

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		if
		---- 	
		1
		(1 row)
		
		m_db=# SELECT @a;
		@a
		---- 	
		1
		(1 row)
		
		m_db=# SELECT IF (0, @a:=10, 2);
		if
		---- 	
		2
		(1 row)
		
		m_db=# SELECT @a;
		@a
		---- 	
		10
		(1 row)
		
		-- In MySQL 5.7, when the *@var*:=**val** parameter is selected, a value is assigned to *@var*. Otherwise, no value is assigned to *@var*.
		mysql> SELECT IF (1, @a:=1, 2);
		+------------------+
		| if (1, @a:=1, 2) |
		+------------------+
		| 1 |
		+------------------+
		1 row in set, 1 warning (0.00 sec)
		
		mysql> SELECT @a;
		+------+
		| @a |
		+------+
		| 1 |
		+------+
		1 row in set (0.00 sec)
		
		mysql> SELECT IF (0, @a:=10, 2);
		+-------------------+
		| if (0, @a:=10, 2) |
		+-------------------+
		| 2 |
		+-------------------+
		1 row in set, 1 warning (0.00 sec)
		
		mysql> SELECT @a;
		+------+
		| @a |
		+------+
		| 1 |
		+------+
		1 row in set (0.00 sec)
		• When a FROM subquery involves user variable assignment operations, GaussDB directly passes the subquery result to the upper layer for access, whereas

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		MySQL materializes the entire FROM subquery result into a temporary table for upper-layer access and query, resulting in differing implementations.
		CREATE TABLE t1(a int);
		INSERT INTO t1 VALUES (1),(2),(3);
		
		-- GaussDB
		m_db=# SELECT @a := a, a FROM t1;
		@a | a
		----+---
		1 | 1
		2 | 2
		3 | 3
		(3 rows)
		
		m_db=# SELECT @a, a FROM (SELECT @a := a, a FROM t1)t;
		@a | a
		----+---
		1 | 1
		2 | 2
		3 | 3
		(3 rows)
		
		m_db=# SELECT @a;
		@a
		---- 	
		3
		(1 row)
		
		-- MySQL 5.7
		mysql> SELECT @a := a, a FROM t1;
		+---------+------+
		| @a := a | a |
		+---------+------+
		| 1 | 1 |
		| 2 | 2 |
		| 3 | 3 |
		+---------+------+
		3 rows in set (0.00 sec)
		
		mysql> SELECT @a, a FROM (SELECT @a := a, a FROM t1)t;
		+------+------+
		| @a | a |
		+------+------+
		| 3 | 1 |
		| 3 | 2 |
		| 3 | 3 |
		+------+------+
		3 rows in set (0.00 sec)
		
		mysql> SELECT @a;
		+------+
		| @a |
		+------+
		| 3 |
		+------+
		1 row in set (0.00 sec)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
INSERT ... ON DUPLICATE KEY UPDATE syntax	INSERT	• The format of *table name.column name* is not supported by VALUES() in the ON DUPLICATE KEY UPDATE clause in GaussDB, but is supported in MySQL. • In GaussDB, constraint columns cannot be modified by using ON DUPLICATE KEY UPDATE, but this operation is allowed in MySQL. • In the INSERT... query ON DUPLICATE KEY UPDATE statement, if query is a subquery, the ON DUPLICATE KEY UPDATE clause cannot reference column names in the subquery in GaussDB, while MySQL supports this behavior. • In MySQL, when you use the ON DUPLICATE KEY UPDATE clause to update multiple columns, the result of the previous UPDATE statement affects the subsequent results. In addition, you can update the same column for multiple times. In GaussDB, the result of the previous UPDATE operation does not affect the subsequent results. In addition, the same column cannot be updated for multiple times. • When the UPDATE operation is performed on inserted data that violates the UNIQUE constraint, the number of affected rows returned by GaussDB is different from that returned by MySQL. When a data record is updated, GaussDB returns **1** and MySQL returns **2**. If such update does not change the value of an existing row, GaussDB

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		returns **1** and MySQL returns **0**.
		• In scenarios where a table column is of the VARCHAR type and is implicitly converted to a numeric data type, if the length of the string that fails to be converted is 2 characters shorter than the original string, MySQL does not generate an alarm or error; if the length of the string that fails to be converted is not 2 characters shorter than the original string, MySQL generates an alarm or error. However, GaussDB always generates an alarm or error when the string fails to be converted.
		-- GaussDB
		m_db=# CREATE TABLE t1(a int PRIMARY KEY, b varchar(10));
		NOTICE: CREATE TABLE / PRIMARY KEY will create implicit index "t1_pkey" for table "t1"
		CREATE TABLE
		
		m_db=# INSERT INTO t1 VALUES(1, 'A');
		INSERT 0 1
		
		m_db=# INSERT INTO t1 VALUES(1, 'X') ON DUPLICATE KEY UPDATE b=values(a)+values(b);
		ERROR: The double value 'X' is incorrect.
		CONTEXT: referenced column: b
		
		m_db=# INSERT INTO t1 VALUES(1, 'YY') ON DUPLICATE KEY UPDATE b=values(a)+values(b);
		ERROR: The double value 'YY' is incorrect.
		CONTEXT: referenced column: b
		
		m_db=# INSERT INTO t1 VALUES(1, 'ZZZ') ON DUPLICATE KEY UPDATE b=values(a)+values(b);
		ERROR: The double value 'ZZZ' is incorrect.
		CONTEXT: referenced column: b
		
		m_db=# DROP TABLE t1;
		DROP TABLE
		
		-- MySQL
		mysql> CREATE TABLE t1(a INT

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		PRIMARY KEY, b VARCHAR(10));
		Query OK, 0 rows affected (0.00 sec)
		
		mysql> TRUNCATE TABLE t1;
		Query OK, 0 rows affected (0.01 sec)
		
		mysql> INSERT INTO t1 VALUES(1, 'A');
		Query OK, 1 row affected (0.00 sec)
		
		mysql> INSERT INTO t1 VALUES(1, 'X') ON DUPLICATE KEY UPDATE b=values(a)+values(b);
		ERROR 1292 (22007): Truncated incorrect DOUBLE value: 'X'
		
		mysql> INSERT INTO t1 VALUES(1, 'YY') ON DUPLICATE KEY UPDATE b=values(a)+values(b);
		Query OK, 2 rows affected, 2 warnings (0.00 sec)
		
		mysql> INSERT INTO t1 VALUES(1, 'ZZZ') ON DUPLICATE KEY UPDATE b=values(a)+values(b);
		ERROR 1292 (22007): Truncated incorrect DOUBLE value: 'ZZZ'
		
		mysql> DROP TABLE t1;
		Query OK, 0 rows affected (0.00 sec)

UNION/EXCEPT statement	SELECT	In GaussDB, when the combined data in the first two queries is in the form of character string literals, if a third query exists, the return type of the set operation is directly determined by the data type of the third query. -- The type in the third query is used for conversion. The data in the second query fails to be converted. m_db=# SELECT '123' AS col UNION SELECT 'test' UNION SELECT 9999 ORDER BY col; WARNING: The integer syntax of the input is invalid: "test". CONTEXT: referenced column: col col ------ 0 123 9999 (3 rows)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.2.7.6 DCL

Table 3-161 DCL syntax compatibility

Description	Syntax	Difference
Set names with COLLATE specified.	SET [SESSION | LOCAL] NAMES {'charset_name' [COLLATE 'collation_name'] | DEFAULT};	GaussDB does not allow **charset_name** to be different from the database character set. For details, see "SQL Reference > SQL Syntax > SQL Statements > S > SET" in *M Compatibility Developer Guide*.

Switch the current mode with USE.	USE schema_name	When the USE statement is used to specify a schema and the user does not have permissions on the schema, MySQL reports an error while GaussDB specifies the current schema as null.
		-- MySQL
		mysql> USE test;
		ERROR 1044 (42000): Access denied for user 'u1'@'%' to database 'test'
		
		-- GaussDB
		m_db=> USE test;
		SET
		m_db=> SELECT database();
		ERROR: function returned NULL
		CONTEXT: referenced column: database

3.3.2.7.7 Other Statements

Table 3-162 Compatibility of other syntaxes

Description	Syntax	Difference
Transaction-related syntax	Default database isolation level	The default isolation level of an M-compatible database is READ COMMITTED, and that of MySQL is REPEATABLE READ. Only the READ COMMITTED and REPEATABLE READ isolation levels take effect in M-compatible databases.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Transaction-related syntax	Transaction nesting	In M-compatible mode, nested transactions are not automatically committed, but in MySQL, they are automatically committed.
Transaction-related syntax	Autocommit	In M-compatible mode, GaussDB is used for storage and the GaussDB transaction mechanism is inherited. If DDL or DCL is executed in a transaction, the transaction is not automatically committed. In MySQL, if DDL, DCL, management-related, or lock-related statements are executed, the transaction is automatically committed.
Transaction-related syntax	Rollback is required after an error is reported.	If an error is reported for a transaction in an M-compatible database, rollback needs to be performed. There is no such restriction in MySQL.
Transaction-related syntax	Lock mechanism	The M-compatible lock mechanism can be used only in transaction blocks. There is no such restriction in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Lock mechanism	Lock mechanism	• After the read lock is obtained, write operations cannot be performed on the current session in MySQL, but write operations can be performed on the current session in an M-compatible database. • After MySQL locks a table, an error is reported when other tables are read. M-compatible does not have such restriction. • In MySQL, if the lock of the same table is obtained in the same session, the previous lock is automatically released and the transaction is committed. M-compatible databases do not have this mechanism. • M-compatible databases allow LOCK TABLE to be used only inside a transaction block, and have no UNLOCK TABLE command. Locks are always released at the end of transactions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
PBE	PBE	• In an M-compatible database, if a PREPARE statement with the same name is repeatedly created, an error is reported, indicating that the statement already exists. You need to delete the existing statement first. In MySQL, the old statement will be overwritten. • M-compatible databases and MySQL report errors in different phases, such as parsing and execution, during SQL statement execution. PREPARE statements process prepared statements till the parsing phrase. Therefore, in abnormal scenarios in PBE, an M-compatible database may be different from that in MySQL in terms of whether the error is reported in the PREPARE or EXECUTE phase.

3.3.2.7.8 Users and Permissions

Overview

In M-compatible mode, the behaviors and syntaxes related to user and permission control inherit the GaussDB mechanism but are not synchronized with those in MySQL.

User and permission behaviors are the same as those in GaussDB. For details, see "Database Security > Users and Permissions" in *Developer Guide*

Some syntaxes for users and permissions are tailored in GaussDB. For details about the syntaxes, see "SQL Reference > SQL Syntax > SQL Statements" in *M Compatibility Developer Guide*. For details about the syntax differences between an M-compatible database and GaussDB, see **Table 3-163**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-163 Syntax differences between an M-compatible database and GaussDB

Syntax	Overview	Difference
CREATE ROLE	Creates a role.	In M-compatible mode, options involving the following keywords cannot be specified: **ENCRYPTED**, **UNENCRYPTED**, **RESOURCE POOL**, **PERM SPACE**, **TEMP SPACE**, and **SPILL SPACE**. When a user is created, a schema with the same name as the user is automatically created in an M-compatible database, but it is not created in MySQL.
CREATE USER	Creates a user.	
CREATE GROUP	Creates a user group. CREATE GROUP is the alias of CREATE ROLE and is not recommended.	
ALTER ROLE	Modifies role attributes.	
ALTER USER	Modifies user attributes.	
ALTER GROUP	Modifies the attributes of a user group.	-
DROP ROLE	Drops a role.	-
DROP USER	Drops a user.	-
DROP GROUP	Drops a user group.	-
DROP OWNED	Drops the database objects owned by a database role.	-
REASSIGN OWNED	Changes the owner of a database object.	This syntax is not supported in an M-compatible database.
GRANT	Grants permissions to roles and users.	In an M-compatible database, permissions on objects such as functions, stored procedures, tablespaces, and database links cannot be granted or revoked.
REVOKE	Revokes permissions from one or more roles.	
ALTER DEFAULT PRIVILEGES	Sets the permissions that will be granted to objects created in the future. (It does not affect permissions granted to existing objects.)	In M-compatible mode, you cannot change functions, types, and permissions on client master keys and column encryption keys of encrypted databases.

Differences

- Syntax format differences

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

For details about the permission granting syntaxes in M-compatible mode, see "SQL Reference > SQL Syntax > G > GRANT" in *M Compatibility Developer Guide*. The permission granting syntax in MySQL is as follows:

```
-- Global, database-level, table-level, and stored procedure–level permission granting syntax
GRANT
    priv_type [(column_list)]
      [, priv_type [(column_list)]] ...
    ON [object_type] priv_level
    TO user [auth_option] [, user [auth_option]] ...
    [REQUIRE {NONE | tls_option [[AND] tls_option] ...}]
    [WITH {GRANT OPTION | resource_option} ...]

-- Syntax for granting permissions to a user proxy
GRANT PROXY ON user
    TO user [, user] ...
    [WITH GRANT OPTION]

object_type: {
    TABLE
  | FUNCTION
  | PROCEDURE
}

priv_level: {
    *
  | *.*
  | db_name.*
  | db_name.tbl_name
  | tbl_name
  | db_name.routine_name
}

user:
    'user_name'@'host_name'

auth_option: {
    IDENTIFIED BY 'auth_string'
  | IDENTIFIED WITH auth_plugin
  | IDENTIFIED WITH auth_plugin BY 'auth_string'
  | IDENTIFIED WITH auth_plugin AS 'auth_string'
  | IDENTIFIED BY PASSWORD 'auth_string'
}

tls_option: {
    SSL
  | X509
  | CIPHER 'cipher'
  | ISSUER 'issuer'
  | SUBJECT 'subject'
}

resource_option: {
  | MAX_QUERIES_PER_HOUR count
  | MAX_UPDATES_PER_HOUR count
  | MAX_CONNECTIONS_PER_HOUR count
  | MAX_USER_CONNECTIONS count
}
```

- Differences in types of permissions granted

  In MySQL, the following types of permissions can be granted.

Table 3-164 Types of permissions that can be granted in MySQL

Permission Type	Definition and Permission Level
ALL [PRIVILEGES]	Grants all permissions of a specified access level, except **GRANT OPTION** and **PROXY**.
ALTER	Enables **ALTER TABLE**. Level: global, database, and table.
ALTER ROUTINE	Allows you to modify or delete stored procedures. Level: global, database, and routine.
CREATE	Enables database and table creation. Level: global, database, and table.
CREATE ROUTINE	Enables stored procedure creation. Level: global and database.
CREATE TABLESPACE	Allows you to create, modify, or delete tablespaces or log file groups. Level: global.
CREATE TEMPORARY TABLES	Enables **CREATE TEMPORARY TABLE**. Level: global and database.
CREATE USER	Enables **CREATE USER**, **DROP USER**, **RENAME USER**, and **REVOKE ALL PRIVILEGES**. Level: global.
CREATE VIEW	Allows you to create or modify views. Level: global, database, and table.
DELETE	Enables **DELETE**. Level: global, database, and table.
DROP	Allows you to delete databases, tables, or views. Level: global, database, and table.
EVENT	Enables scheduled tasks. Level: global and database.
EXECUTE	Allows you to execute stored procedures. Level: global, database, and stored procedure.
FILE	Allows you to enable the server to read or write files. Level: global.
GRANT OPTION	Allows you to grant permissions to or remove permissions from other accounts. Level: global, database, table, stored procedure, and proxy.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Permission Type	Definition and Permission Level
INDEX	Allows you to create or delete indexes. Level: global, database, and table.
INSERT	Enables **INSERT**. Level: global, database, table, and column.
LOCK TABLES	LOCK TABLES can be enabled on tables with the SELECT permission. Level: global and database.
PROCESS	Allows users to view all running threads through **SHOW PROCESSLIST**. Level: global.
PROXY	Enables a user proxy. Level: from user to user.
REFERENCES	Enables foreign key creation. Level: global, database, table, and column.
RELOAD	Enables **FLUSH**. Level: global.
REPLICATION CLIENT	Allows you to query the location of the source server or replica server. Level: global.
REPLICATION SLAVE	Allows replicas to read binary logs from the source. Level: global.
SELECT	Enables **SELECT**. Level: global, database, table, and column.
SHOW DATABASES	Enables **SHOW DATABASES** to display all databases. Level: global.
SHOW VIEW	Enables **SHOW CREATE VIEW**. Level: global, database, and table.
SHUTDOWN	Enables **mysqladmin shutdown**. Level: global.
SUPER	Enables other management operations, such as **CHANGE MASTER TO**, **KILL**, **PURGE BINARY LOGS**, and **SET GLOBAL**, as well as the **mysqladmin debug** command. Level: global.
TRIGGER	Enables TRIGGER. Level: global, database, and table.
UPDATE	Enables **UPDATE**. Level: global, database, table, and column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Permission Type	Definition and Permission Level
USAGE	Equivalent to "no privilege".

M-compatible databases support the following permissions by level:

Table 3-165 Types of permissions that can be granted in M-compatible databases

Object	Permissions That Can Be Granted
Database	CREATE, CONNECT, TEMPORARY, TEMP, ALTER, DROP, and COMMENT
Schema	CREATE, USAGE, ALTER, DROP, and COMMENT
Table and view	SELECT, INSERT, UPDATE, DELETE, TRUNCATE, REFERENCES, TRIGGER, ALTER, DROP, COMMENT, INDEX, and VACUUM
Column	SELECT, INSERT, UPDATE, REFERENCES, and COMMENT
Sequence	SELECT, USAGE, UPDATE, ALTER, DROP, and COMMENT

- In MySQL, *'*.*'* represents a global-level authorization object; in GaussDB, *'{DATABASE} db_name'* represents a database-level authorization object. The database level in GaussDB corresponds to the global level in MySQL.
- In MySQL, *'schema_name.*'* represents a database/schema-level authorization object; in GaussDB, *'{SCHEMA} schema_name'* represents a schema-level authorization object. The schema level in GaussDB corresponds to the database/schema level in MySQL.
- In MySQL, a username consists of two parts: *username*@*hostname*; in an M-compatible database, a username consists of only the username itself.
- MySQL allows you to modify user verification, secure connection, and resource parameter attributes (including **auth_option**, **tls_option**, and **resource option**) using the GRANT syntax. In M-compatible databases, these cannot be modified using the GRANT syntax; instead, you need to use CREATE USER and ALTER USER to set user attributes.
- MySQL supports permission granting with a user proxy. GRANT PROXY ON is used to manage permissions of users in a unified manner. MySQL 5.7 does not provide the role mechanism, but MySQL 8.0 and M-compatible databases provide the role mechanism. If a role can manage and control the permissions of users in a unified manner, it can replace GRANT PROXY ON.
- M-compatible databases have a concept called "public". All users have the public permission and can query some system catalogs and system views. Users can grant or revoke public permissions. In MySQL, newly created users

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

have only the global usage permission, which is almost low to none. They have only the permission to connect to the database and query the information_schema database.

- In M-compatible databases, the owner of an object has all permissions on the object by default. For security purposes, the owner can discard some permissions. However, ALTER, DROP, COMMENT, INDEX, VACUUM, and re-grantable permissions on the object are implicitly inherent, unrevocable permissions of the owner. In MySQL, there is no notion of "owner". Even if a user creates a table, they cannot perform operations such as IUD on the table without being granted the corresponding permissions.
- In MySQL, all users have the USAGE permission, which indicates no permission. When **REVOKE** or **GRANT USAGE** is executed, no modification is made. In M-compatible databases, the USAGE permission has the following meanings:
  - For schemas, USAGE allows access to objects contained in the schema. Without this permission, it is still possible to see the object names.
  - For sequences, USAGE allows use of the nextval function.
- In M-compatible databases, administrator roles can be set for users, including the system administrator (SYSADMIN), security administrator (CREATEROLE), audit administrator (AUDITADMIN), monitor administrator (MONADMIN), O&M administrator (OPRADMIN), and security policy administrator (POLADMIN). By default, SYSADMIN have the highest permission in the system. After separation of duties is enabled, system administrators no longer possess the CREATEROLE or AUDITADMIN attribute. That is, system administrators can neither create roles or users, nor view or maintain database audit logs. In MySQL, administrator roles cannot be set for users, and there is no design for separation of duties.
- In M-compatible databases, the ANY permission can be granted to users, allowing them to have the corresponding permissions in non-system schemas, including CREATE ANY TABLE, SELECT ANY TABLE, TRUNCATE ANY TABLE, and CREATE ANY INDEX. In MySQL, the ANY permission cannot be granted.
- MySQL provides SHOW GRANTS to query user permissions. In M-compatible databases, you can run gsql client meta-commands such as **'\l+'**, **'\dn+'**, or **'\dp'** to query permission information, or query related columns in system catalogs such as pg_namespace, pg_class, and pg_attribute for permission information.
- When a database, table, or column is deleted, the related permission granting information is still retained in system catalogs in MySQL. If an object with the same name is created again, the user still has the original permissions. However, M-compatible databases will not retain such permission granting information. If an object with the same name is created again, permissions need to be granted again.
- When granting database-level permissions, MySQL supports fuzzy match of database names using underscores (_) and percent signs (%). However, M-compatible databases do not support fuzzy match of object names using special characters such as underscores (_) or percent signs (%), which are identified as common characters.
- In MySQL, if a non-existing user is specified in the GRANT statement, a user account is created by default (this feature has been removed from MySQL

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.0). In M-compatible databases, permissions cannot be granted to users who have not been created.

3.3.2.7.9 System Catalogs and System Views

Table 3-166 Differences between M-compatible databases and GaussDB in terms of system catalogs or views

System Catalog or System View	Column	Difference
information_schema.columns	generation_expression	The output of this column varies due to different string concatenation logics of expressions in M-compatible mode and MySQL.
information_schema.columns	data_type	The output result of this column in M-compatible mode, having not been modified due to the data type format_type involved, is different from that in MySQL.
information_schema.columns	column_type	The output result of this column in M-compatible mode, having not been modified due to the data type format_type involved, is different from that in MySQL.
information_schema.tables	engine	In M-compatible mode: • ENGINE is aligned with data of information_schema.engines. • In some system catalogs, ENGINE is left empty. • If the default table is an ASTORE table and **STORAGE_TYPE** is not specified, **ENGINE** is empty.
information_schema.tables	version	This column is not supported in M-compatible mode.
information_schema.tables	row_format	This column is not supported in M-compatible mode.
information_schema.tables	avg_row_length	In M-compatible mode, the result of dividing the size of the data files by the number of all tuples (including live tuples and dead tuples) is used. If there is no tuple in the table, the value is **null**.
information_schema.tables	max_data_length	This column is not supported in M-compatible mode.
information_schema.tables	data_free	In M-compatible mode, it indicates the result of (Number of dead tuples/Total number of tuples) x Data file size. If there is no tuple in the table, the value is **null**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

System Catalog or System View	Column	Difference
information_schema.tables	check_time	This column is not supported in M-compatible mode.
information_schema.tables	create_time	The behavior of this column in M-compatible mode is different from that in MySQL. When a view is created in MySQL, this column is set to **null**. In M-compatible mode, the actual table creation time is displayed. The value is **null** if it is a table or view provided by the database.
information_schema.tables	update_time	The value is **null** if it is a table or view provided by the M-compatible database.
information_schema.statistics	collation	The value can only be **A** or **D** but not NULL in M-compatible mode.
information_schema.statistics	packed	This column is not supported in M-compatible mode.
information_schema.statistics	sub_part	This column is not supported in M-compatible mode.
information_schema.statistics	comment	This column is not supported in M-compatible mode.
information_schema.partitions	subpartition_name	In M-compatible mode, if the specified object is not a subpartition, the value is **null**.
information_schema.partitions	subpartition_ordinal_position	In M-compatible mode, if the specified object is not a subpartition, the value is **null**.
information_schema.partitions	partition_method	In M-compatible mode: Partitioning policy. If it is not a partition, the value is **null**. ● **'r'**: range partition. ● **'i'**: interval partition. ● **'l'**: list partition. ● **'h'**: hash partition

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

System Catalog or System View	Column	Difference
information_schema.partitions	subpartition_method	In M-compatible mode: Subpartitioning policy. If it is not a subpartition, the value is **null**. ● **'r'**: range partition. ● **'i'**: interval partition. ● **'l'**: list partition. ● **'h'**: hash partition
information_schema.partitions	partition_description	In M-compatible mode, there are partitions and subpartitions.
information_schema.partitions	partition_expression	This column is not supported in M-compatible mode.
information_schema.partitions	subpartition_expression	This column is not supported in M-compatible mode.
information_schema.partitions	data_length	This column is not supported in M-compatible mode.
information_schema.partitions	max_data_length	This column is not supported in M-compatible mode.
information_schema.partitions	index_length	This column is not supported in M-compatible mode.
information_schema.partitions	data_free	This column is not supported in M-compatible mode.
information_schema.partitions	create_time	This column is not supported in M-compatible mode.
information_schema.partitions	update_time	This column is not supported in M-compatible mode.
information_schema.partitions	check_time	This column is not supported in M-compatible mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

System Catalog or System View	Column	Difference
information_schema.partitions	checksum	This column is not supported in M-compatible mode.
information_schema.partitions	partition_comment	This column is not supported in M-compatible mode.
information_schema.partitions	nodegroup	This column is not supported in M-compatible mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The precision range cannot be specified for the command output of the integer type in a view. For example, the bigint(1) type in MySQL corresponds to the bigint type in M-compatible mode, and the bigint(21) unsigned type in MySQL corresponds to the bigint unsigned type in M-compatible mode.
- The int type in MySQL corresponds to the integer type in M-compatible mode.
- M-compatible mode does not support columns of the set and enum types that are supported in MySQL. This version does not support or display **Column_priv** column in the m_schema.columns_priv view, **Table_priv,Column_priv** column in the m_schema.tables_priv view, **Routine_type,Proc_priv** column in the m_schema.procs_priv view, the **type,language,sql_data_access,is_deterministic,security_type,sql_mode** column in the m_schema.proc view, or the **type** column in the m_schema.func view.
- **table_rows**, **avg_row_length**, **data_length**, **data_free**, **index_length**, and **cardinality** in information_schema.tables and **cardinality** in information_schema.statistics are obtained based on statistics. Therefore, run **ANALYZE** to update statistics before viewing them. (If data is updated in the database, you are advised to delay running **ANALYZE**.)
- The index columns contained in information_schema.statistics must be complete table columns in the created indexes. If the index columns are expressions, they are not in this view.
- **table_row** in information_schema.partitions is obtained based on statistics. Before viewing the value, run **ANALYZE** to update the statistics. (If data is updated in the database, you are advised to delay running **ANALYZE**.)
- The format of the **grantee** column supported in MySQL is '*user_name*'**@**'*host_name*'. In an M-compatible database, it is the name of the user or role to which the permission is granted.
- For the **host** column supported in the M-compatible database, the **hostname** of the current node is returned.
- In MySQL, you need the permission before viewing m_schema.tables_priv, information_schema.user_privileges, information_schema.schema_privileges, information_schema.table_privileges, information_schema.column_privileges, m_schema.columns_priv, m_schema.func, and m_schema.procs_priv. In an M-compatible database, you can view them with the default permission. For example, for table **t1**, you need the corresponding permission in MySQL so that you can view the corresponding permission information in the permission view. In an M-compatible database, you can view the permission information related to table **t1** in the view.
- A system view in m_schema is a system catalog in MySQL.
- The collations of **VIEW_DEFINITION** in information_schema.views and **ROUTINE_DEFINITION** in information_schema.routines are not controlled.
- For the view columns of the character type listed in "Schemas" in *M Compatibility Developer Guide*, the character set is utf8mb4, and the collation is utf8mb4_bin or utf8mb4_general_ci, and the collation priority is the priority of columns of data types that support collation described in "SQL Reference > Character Set and Collations > Rules for Combining Character Sets and Collations" in *M Compatibility Developer Guide*. These features are different from those in MySQL.

3.3.3 MYSQL-compatible Mode

3.3.3.1 Data Types

3.3.3.1.1 Numeric Data Types

Integer Types

Unless otherwise specified, the precision, scale, and number of digits cannot be set to the floating-point values in MySQL-compatible mode by default. You are advised to use valid integer values.

Differences in terms of the integer types:

- Input format:
  - MySQL

    For characters such as "asbd", "12dd", and "12 12", the system truncates them or returns 0 and reports a WARNING. Data fails to be inserted into a table in strict mode.
  - GaussDB
    - For integer types (TINYINT, SMALLINT, MEDIUMINT, INT, INTEGER, and BIGINT), if the invalid part of a character string is truncated, for example, "12@3" or "1E09" is converted into a scientific notation of an integer, no message is displayed. Data is successfully inserted into a table.
    - If the whole integer is truncated (for example, "@123") or the character string is empty, 0 is returned and data is successfully inserted into a table.
- Operators:
  - +, -, and *

    GaussDB: When INT, INTEGER, SMALLINT, or BIGINT is used for calculation, a value of the original type is returned and is not changed to a larger type. If the return value exceeds the range, an error is reported.

    MySQL: The value can be changed to BIGINT for calculation.
  - |, &, ^, and ~

    GaussDB: The value is calculated in the bits occupied by the type. In GaussDB, ^ indicates the exponentiation operation. If the XOR operator is required, replace it with #.

    MYSQL: The value is changed to a larger type for calculation.
  - <, <=, <=>, <>, =, >, and >=

    GaussDB: When an integer is compared with a string, the string is converted to an integer before the comparison. If a string contains non-digit characters such as decimal points and letters, the non-integer part is directly truncated for comparison. If the string is completely truncated (for example, "@123") or is empty, the string type is converted to 0 before being compared with the integer type.

    MYSQL: When an integer is compared with a string, they are converted to floating-point numbers for comparison.
- Explicit type conversion of negative numbers:

  GaussDB: The result is **0** in loose mode and an error is reported in strict mode.

  MySQL: The most significant bit is replaced with a numeric bit based on the corresponding binary value, for example, (-1)::uint4 = 4294967295.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Other differences:

  The precision of INT[(M)] controls formatted output in MySQL. GaussDB supports only the syntax but does not support the function.

- Aggregate function:
  - variance: indicates the sample variance in GaussDB and the population variance in MySQL.
  - stddev: indicates the sample standard deviation in GaussDB and the overall standard deviation in MySQL.
- Display width:
  - If **ZEROFILL** is not specified when the width information is specified for an integer column, the width information is not displayed in the table definition.
  - When the INSERT statement is used to insert a column of the character type, GaussDB pads 0s before inserting the column.
  - The JOIN USING statement involves type derivation. In MySQL, the first table column is used by default. In GaussDB, if the result is of the signed type, the width information is invalid. Otherwise, the width of the first table column is used.
  - For GREATEST/LEAST, IFNULL/IF, and CASE WHEN/DECODE, MySQL does not pad 0s. In GaussDB, 0s are padded when the type and width information is consistent.
  - MySQL supports this function when it is used as the input or output parameter or return value of a function or stored procedure. GaussDB neither reports syntax errors nor supports this function.

For details about the differences in GaussDB and MySQL in terms of the integer types, see **Table 3-167**.

Table 3-167 Integer types

MySQL	GaussDB	Difference
BOOL	Supported, but with differences.	MySQL: The BOOL/BOOLEAN type is actually mapped to the TINYINT type. GaussDB: BOOL is supported.
BOOLEAN	Supported, but with differences.	• Valid literal values for the "true" state include: **TRUE**, **'t'**, **'true'**, **'y'**, **'yes'**, **'1'**, **'TRUE'**, **true**, **'on'**, and all non-zero values. • Valid literal values for the "false" state include: **FALSE**, **'f'**, **'false'**, **'n'**, **'no'**, **'0'**, **0**, **'FALSE'**, **false**, and **'off'**. **TRUE** and **FALSE** are standard expressions, compatible with SQL statements.
TINYINT[(M)] [UNSIGNED]	Supported, but with differences.	For details, see **Differences in terms of the integer types**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
SMALLINT[(M)] [UNSIGNED]	Supported, but with differences.	For details, see **Differences in terms of the integer types**.
MEDIUMINT[(M)] [UNSIGNED]	Supported, but with differences.	MySQL requires 3 bytes to store MEDIUMINT data. • The signed range is –8388608 to +8388607. • The unsigned range is 0 to +16777215. GaussDB maps data to the INT type and requires 4 bytes for storage. • The signed range is –2147483648 to +2147483647. • The unsigned range is 0 to +4294967295. For details about other differences, see **Differences in terms of the integer types**.
INT[(M)] [UNSIGNED]	Supported, but with differences.	For details, see **Differences in terms of the integer types**.
INTEGER[(M)] [UNSIGNED]	Supported, but with differences.	For details, see **Differences in terms of the integer types**.
BIGINT[(M)] [UNSIGNED]	Supported, but with differences.	For details, see **Differences in terms of the integer types**.

NOTE

If data is different due to different compatibility, the column has an index, and the compatibility environment when the index is used is different from that when the index is built, the index may become invalid. In this case, you need to reindex it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Arbitrary precision types

Table 3-168 Arbitrary precision types

MySQL	GaussDB	Difference
DECIMAL[(M[,D])]	Supported, but with differences.	• Operator: In GaussDB, "^" indicates the exponentiation operation. If the XOR operator is required, replace it with "#". In MySQL, "^" indicates the XOR operation. • Value range: The precision **M** and scale **D** support only integers and do not support floating-point values. • Input format: No error is reported when all input parameters of a character string (for example, "@123") are truncated. An error is reported only when it is partially truncated, for example, "12@3".
NUMERIC[(M[,D])]	Supported, but with differences.	
DEC[(M[,D])]	Supported, but with differences.	
FIXED[(M[,D])]	Not supported.	-

Floating-point types

Table 3-169 Floating-point types

MySQL	GaussDB	Difference
FLOAT[(M,D)]	Supported, but with differences.	• Partitioned table: The FLOAT data type does not support partitioned tables with the key partitioning policy. • Operator: In GaussDB, "^" indicates the exponentiation operation. If the XOR operator is required, replace it with "#". In MySQL, "^" indicates the XOR operation. • Value range: The precision **M** and scale **D** support only integers and do not support floating-point values. • Output format: An ERROR message is reported for invalid input parameters. No WARNING is reported in loose mode (that is, **sql_mode** is set to **''**).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
FLOAT(p)	Supported, but with differences.	• Partitioned table: The FLOAT data type does not support partitioned tables with the key partitioning policy. • Operator: The ^ operator is used for the numeric types, which is different from that in MySQL. In GaussDB, the ^ operator is used for exponential calculation. • Value range: When the precision **p** is defined, only valid integer data types are supported. • Output format: An ERROR message is reported for invalid input parameters. No WARNING is reported in loose mode (that is, **sql_mode** is set to **''**).
DOUBLE[(M, D)]	Supported, but with differences.	• Partitioned table: The DOUBLE data type does not support partitioned tables with the key partitioning policy. • Operator: In GaussDB, "^" indicates the exponentiation operation. If the XOR operator is required, replace it with "#". In MySQL, "^" indicates the XOR operation. • Value range: The precision **M** and scale **D** support only integers and do not support floating-point values. • Output format: An ERROR message is reported for invalid input parameters. No WARNING is reported in loose mode (that is, **sql_mode** is set to **''**).
DOUBLE PRECISION[(M,D)]	Supported, but with differences.	• Operator: In GaussDB, "^" indicates the exponentiation operation. If the XOR operator is required, replace it with "#". In MySQL, "^" indicates the XOR operation. • Value range: The precision **M** and scale **D** support only integers and do not support floating-point values. • Output format: An ERROR message is reported for invalid input parameters. No WARNING is reported in loose mode (that is, **sql_mode** is set to **''**).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
REAL[(M,D)]	Supported, but with differences.	• Partitioned table: The REAL data type does not support partitioned tables with the key partitioning policy. • Operator: In GaussDB, "^" indicates the exponentiation operation. If the XOR operator is required, replace it with "#". In MySQL, "^" indicates the XOR operation. • Value range: The precision **M** and scale **D** support only integers and do not support floating-point values. • Output format: An ERROR message is reported for invalid input parameters. No WARNING is reported in loose mode (that is, **sql_mode** is set to **''**).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Sequential Integers

Table 3-170 Sequential integers

MySQL	GaussDB	Difference
SERIAL	Supported, but with differences.	For details about SERIAL in GaussDB, see **Numeric Types**. The differences in specifications are as follows: `CREATE TABLE test(f1 serial, f2 CHAR(20));` • Difference in type definition: MySQL maps serial to BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE, and GaussDB maps serial to INTEGER NOT NULL DEFAULT nextval('test_f1_seq'::regclass). For example: `-- Definition of MySQL SERIAL:` `mysql> SHOW CREATE TABLE test\G` `*************************** 1. row ***************************` `Table: test` `Create Table: CREATE TABLE `test` (` `  `f1` bigint(20) unsigned NOT NULL AUTO_INCREMENT,` `  `f2` char(20) DEFAULT NULL,` `  UNIQUE KEY `f1` (`f1`)` `) ENGINE=InnoDB DEFAULT CHARSET=utf8` `1 row in set (0.00 sec)` `-- Definition of GaussDB SERIAL` `gaussdb=# \d+ test` `Table "public.test"` ` Column	Type	Modifiers	Storage	Stats target	Description` `--------+---------- +---------- +---------- +--------------+-------------` ` f1	integer	not null default nextval('test_f1_seq'::regclass)	plain		` ` f2	character(20)		extended		` `Has OIDs: no` `Options: orientation=row, compression=no, storage_type=USTORE` • Differences in using INSERT to insert default values of the SERIAL type. For example: `-- Inserting default values of the SERIAL type in MySQL` `mysql> INSERT INTO test VALUES(DEFAULT, 'aaaa');` `Query OK, 1 row affected (0.00 sec)`  `mysql> INSERT INTO test VALUES(10, 'aaaa');` `Query OK, 1 row affected (0.00 sec)`  `mysql> INSERT INTO test VALUES(DEFAULT, 'aaaa');` `Query OK, 1 row affected (0.00 sec)`  `mysql> SELECT * FROM test;` `+----+------+` `| f1	f2 |` `+----+------+` `| 1	aaaa |` `| 10	aaaa |` `| 11	aaaa |` `+----+------+` `3 rows in set (0.00 sec)`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference

		-- Inserting default values of the SERIAL type in GaussDB
		gaussdb=# INSERT INTO test VALUES(DEFAULT, 'aaaa');
		INSERT 0 1
		gaussdb=# INSERT INTO test VALUES(10, 'aaaa');
		INSERT 0 1
		gaussdb=# INSERT INTO test VALUES(DEFAULT, 'aaaa');
		INSERT 0 1
		gaussdb=# SELECT * FROM test;
		f1 | f2
		----+----------------------
		1 | aaaa
		2 | aaaa
		10 | aaaa
		(3 rows)
		• Differences in referencing columns of the serial type in REPLACE scenarios. For details about how to reference columns in GaussDB, see **REPLACE**. For example:
		-- Inserting values of the referencing columns of the SERIAL type in MySQL
		mysql> REPLACE INTO test VALUES(f1, 'aaaa');
		Query OK, 1 row affected (0.00 sec)
		
		mysql> REPLACE INTO test VALUES(f1, 'bbbb');
		Query OK, 1 row affected (0.00 sec)
		
		mysql> SELECT * FROM test;
		+----+------+
		| f1 | f2 |
		+----+------+
		| 1 | aaaa |
		| 2 | bbbb |
		+----+------+
		2 rows in set (0.00 sec)
		
		-- Inserting values of the referencing columns of the SERIAL type in GaussDB
		gaussdb=# REPLACE INTO test VALUES(f1, 'aaaa');
		REPLACE 0 1
		gaussdb=# REPLACE INTO test VALUES(f1, 'bbbb');
		REPLACE 0 1
		gaussdb=# SELECT * FROM test;
		f1 | f2
		----+----------------------
		0 | aaaa
		0 | bbbb
		(2 rows)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.3.1.2 Date and Time Data Types

Table 3-171 Date and Time Data Types

MySQL	GaussDB	Difference
DATE	Supported, with differences	GaussDB supports the date data type. Compared with MySQL, GaussDB has the following differences in specifications: ● Input format – GaussDB supports only the character type and does not support the numeric type. For example, the format can be '2020-01-01' or '20200101', but cannot be 20200101. MySQL supports conversion from numeric input to the date type. – Separator: GaussDB does not support the plus sign (+) or colon (:) as the separator between the year, month, and day. Other symbols are supported. MySQL supports all symbols as separators. Sometimes, the mixed use of separators is not supported, which is different from MySQL, such as '2020-01>01' and '2020/01+01'. You are advised to use hyphens (-) or slashes (/) as separators. – No separator: You are advised to use the complete format, for example, 'YYYYMMDD' or 'YYMMDD'. The parsing rules of incomplete formats (including the ultra-long format) are different from those of MySQL. An error may be reported or the parsing result may be inconsistent with that of MySQL. Therefore, the incomplete format is not recommended. ● Output format If the **sql_mode** parameter of GaussDB does not contain **'strict_trans_tables'** (the strict mode is used unless otherwise defined as the loose mode), the year, month, and day can be set to **0**. However, the value is converted to a valid value in the sequence of year, month, and day. For example, date '0000-00-10' is converted to 0002-12-10 BC. If the input is invalid or exceeds the range, a warning message is reported and the value **0000-00-00** is returned. MySQL outputs the date value as it is, even if the year, month, and day are set to **0**. ● Value range

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference

		The value range of GaussDB is 4713-01-01 BC to 5874897-12-31 AD. BC dates are supported. In loose mode, if the value exceeds the range, **0000-00-00** is returned. In strict mode, an error is reported. In MySQL, the value range is 0000-00-00 to 9999-12-31. In loose mode, if the value exceeds the range, the performance varies in different scenarios. An error may be reported (for example, in the SELECT statement) or the value **0000-00-00** may be returned (for example, in the INSERT statement). As a result, when the date type is used as the input parameter of the function, the results returned by the function are different.
		● Operator
		– GaussDB supports only the comparison operators =, !=, <, <=, >, and >= between date types and returns **true** or **false**. For the addition operation between the date and interval types, the return result is of the date type. For the subtraction operation between the date and interval types, the return result is of the date type. For the subtraction operation between date types, the return result is of the interval type.
		– When the MySQL date type and other numeric types are calculated, the date type is converted to the numeric type, and then the calculation is performed based on the numeric type. The result is also of the numeric type. It is different from GaussDB. For example:
		`-- MySQL: date+numeric. Convert the date type to 20200101 and add it to 1. The result is 20200102.`
		`mysql> SELECT date'2020-01-01' + 1;`
		`+----------------------+`
		`| date'2020-01-01' + 1 |`
		`+----------------------+`
		`| 20200102 |`
		`+----------------------+`
		`1 row in set (0.00 sec)`
		
		`-- GaussDB: date+numeric. Convert the numeric type to the interval type (1 day), and then add them up to obtain a new date.`
		`gaussdb=# SELECT date'2020-01-01' + 1;`
		`?column?`
		`------------`
		`2020-01-02`
		`(1 row)`
		● Type conversion

MySQL	GaussDB	Difference
		Compared with MySQL, GaussDB supports conversion between the date type and char(n), nchar(n), datetime, or timestamp type, but does not support conversion between the date type and binary, decimal, JSON, integer, unsigned integer, or time type. The principles for determining common types in scenarios such as collections and complex expressions are different from those in MySQL. For details, see **Data Type Conversion**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
DATETIME[(fsp)]	Supported, with differences.	GaussDB supports the datetime data type. Compared with MySQL, GaussDB has the following differences in specifications: ● Input format – GaussDB supports only the character type and does not support the numeric type. For example, '2020-01-01 10:20:30.123456' or '20200101102030.123456' is supported, but 20200101102030.123456 is not supported. MySQL supports conversion from numeric input to the datetime type. – Separator: GaussDB does not support the plus sign (+) or colon (:) as the separator between the year, month, and day. Other symbols are supported. Only colons (:) can be used as separators between hours, minutes, and seconds. Sometimes, the mixed use of separators is not supported, which is different from MySQL. Therefore, it is not recommended. MySQL supports all symbols as separators. – No separator: In GaussDB, the complete format 'YYYYMMDDhhmiss.ffffff' is recommended. The parsing rules of incomplete formats (including the ultra-long format) may be different from those of MySQL. An error may be reported or the parsing result may be inconsistent with that of MySQL. Therefore, the incomplete format is not recommended. ● Output format: – The format is 'YYYY-MM-DD hh:mi:ss.ffffff', which is the same as that of MySQL and is not affected by the **DateStyle** parameter. However, for the precision part, if the last several digits are 0, they are not displayed in GaussDB but displayed in MySQL. – If the **sql_mode** parameter of GaussDB does not contain **'strict_trans_tables'** (the strict mode is used unless otherwise defined as the loose mode), the year, month, and day can be set to **0**. However, the value is converted to a valid value in the sequence of year, month, and day. For example, datetime '0000-00-10 00:00:00'

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
		is converted to 0002-12-10 00:00:00 BC. If the input is invalid or exceeds the range, a warning message is reported and the value **0000-00-00 00:00:00** is returned. MySQL outputs the datetime value as it is, even if the year, month, and day are set to **0**. ● Value range 4713-11-24 00:00:00.000000 BC to 294277-01-09 04:00:54.775806 AD. If the value is **294277-01-09 04:00:54.775807 AD**, **infinity** is returned. If the value exceeds the range, GaussDB reports an error in strict mode. Whether MySQL reports an error depends on the application scenario. Generally, no error is reported in the query scenario. However, an error is reported when the DML or SQL statement is executed to change the value of a table attribute. In loose mode, GaussDB returns **0000-00-00 00:00:00**. MySQL may report an error, return **0000-00-00 00:00:00**, or return **null** based on the application scenario. As a result, the execution result of the function that uses the datetime type as the input parameter is different from that of MySQL. ● Precision The value ranges from 0 to 6. For a table column, the default value is **0**, which is the same as that in MySQL. In the datetime[(p)]'str' expression, GaussDB parses **(p)** as the precision. The default value is **6**, indicating that **'str'** is formatted to the datetime type based on the precision specified by **p**. MySQL does not support the datetime[(p)]'str' expression. ● Operator – GaussDB supports only the comparison operators =, !=, <, <=, >, and >= between datetime types and returns **true** or **false**. For the addition operation between the datetime and interval types, the return result is of the datetime type. For the subtraction operation between the datetime and interval types, the return result is of the datetime type. For the subtraction operation between datetime types, the return result is of the interval type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference

		– When the MySQL datetime type and other numeric types are calculated, the datetime type is converted to the numeric type, and then the calculation is performed based on the numeric type. The result is also of the numeric type. It is different from GaussDB. For example:
		-- MySQL: datetime+numeric. Convert the datetime type to 20201010123456 and add it to 1. The result is 20201010123457.
		mysql> SELECT cast('2020-10-10 12:34:56.123456' AS datetime) + 1;
		+--+
		| cast('2020-10-10 12:34:56.123456' AS datetime) + 1 |
		+--+
		| 20201010123457 |
		+--+
		1 row in set (0.00 sec)
		
		-- GaussDB: datetime+numeric. Convert the numeric type to the interval type (1 day), and then add them up to obtain the new datetime.
		gaussdb=# SELECT cast('2020-10-10 12:34:56.123456' AS datetime) + 1;
		?column?
		--------------------- 	
		2020-10-11 12:34:56
		(1 row)
		If the calculation result of the datetime type and numeric type is used as the input parameter of a function, the result of the function may be different from that of MySQL.
		• Type conversion
		Compared with MySQL, GaussDB supports only the conversion between the datetime type and the char(n), varchar(n), or timestamp type, and the conversion from datetime to date or time (only value assignment and explicit conversion). Conversion between binary, decimal, json, integer, and unsigned integer types is not supported. The principles for determining common types in scenarios such as collections and complex expressions are different from those in MySQL. For details, see **Data Type Conversion**.
		• Time zone
		In GaussDB, the datetime value can carry the time zone information (time zone offset or time zone name), for example, '2020-01-01 12:34:56.123456 +01:00' or '2020-01-01 2:34:56.123456 CST'. GaussDB converts the time to the time of the current server time

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
		zone. MySQL 5.7 does not support this function. MySQL 8.0 and later versions support this function. • The table columns of the datetime data type in GaussDB are actually converted to the timestamp(p) without time zone. When you query the table information or use a tool to export the table definition, the data type of columns is timestamp(p) without time zone instead of datetime. For MySQL, datetime(p) is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
TIMESTAMP[(fsp)]	Supported, with differences	GaussDB supports the timestamp data type and differs from MySQL in terms of the following specifications: ● Input format: – It supports only the character type and does not support the numeric type. For example, '2020-01-01 10:20:30.123456' or '20200101102030.123456' is supported, but 20200101102030.123456 is not supported. MySQL supports conversion from numeric input to the timestamp type. – Separator: It does not support the plus sign (+) or colon (:) as the separator between the year, month, and day. Other symbols are supported. Only colons (:) can be used as separators between hours, minutes, and seconds. Sometimes, the mixed use of separators is not supported, which is different from MySQL. Therefore, it is not recommended. MySQL supports all symbols as separators. – No separator: The complete format 'YYYYMMDDhhmiss.ffffff' is recommended. The parsing rules of incomplete formats (including the ultra-long format) may be different from those of MySQL. An error may be reported or the parsing result may be inconsistent with that of MySQL. Therefore, the incomplete format is not recommended. ● Output format: – The format is 'YYYY-MM-DD hh:mi:ss.ffffff', which is the same as that of MySQL and is not affected by the **DateStyle** parameter. However, for the precision part, if the last several digits are 0, they are not displayed in GaussDB but displayed in MySQL. – If the **sql_mode** parameter of GaussDB does not contain **'strict_trans_tables'** (the strict mode is used unless otherwise defined as the loose mode), the year, month, and day can be set to **0**. However, the value is converted to a valid value in the sequence of year, month, and day. For example, timestamp '0000-00-10

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
		00:00:00' is converted to 0002-12-10 00:00:00 BC. If the input is invalid or exceeds the range, a warning message is reported and the value **0000-00-00 00:00:00** is returned. MySQL outputs the timestamp value as it is, even if the year, month, and day are set to **0**. ● Value range: 4713-11-24 00:00:00.000000 BC to 294277-01-09 04:00:54.775806 AD. If the value is **294277-01-09 04:00:54.775807 AD**, **infinity** is returned. If the value exceeds the range, GaussDB reports an error in strict mode. Whether MySQL reports an error depends on the application scenario. Generally, no error is reported in the query scenario. However, an error is reported when the DML or SQL statement is executed to change the value of a table attribute. In loose mode, GaussDB returns **0000-00-00 00:00:00**. MySQL may report an error, return **0000-00-00 00:00:00**, or return **null** based on the application scenario. As a result, the execution result of the function that uses the timestamp type as the input parameter is different from that of MySQL. ● Precision: The value ranges from 0 to 6. For a table column, the default value is **0**, which is the same as that in MySQL. In the timestamp[(p)] 'str' expression: – GaussDB parses **(p)** as the precision. The default value is **6**, indicating that **'str'** is formatted to the timestamp type based on the precision specified by **p**. – The meaning of timestamp 'str' in MySQL is the same as that in GaussDB. The default precision is **6**. However, timestamp(p) 'str' is parsed as a function call. **p** is used as the input parameter of the timestamp function. The result returns a value of the timestamp type, and **'str'** is used as the alias of the projection column. ● Operators: – GaussDB supports only the comparison operators =, !=, <, <=, >, and >= between timestamp types and returns **true** or **false**. For the addition operation between

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference

		the timestamp and interval types, the return result is of the timestamp type. For the subtraction operation between the timestamp and interval types, the return result is of the timestamp type. For the subtraction operation between timestamp types, the return result is of the interval type.
		– When the MySQL timestamp type and other numeric types are calculated, the timestamp type is converted to the numeric type, and then the calculation is performed based on the numeric type. The result is also of the numeric type. It is different from GaussDB. For example:
		`-- MySQL: timestamp+numeric. Convert the timestamp type to 20201010123456.123456 and add it to 1. The result is 20201010123457.123456.`
		`mysql> SELECT timestamp '2020-10-10 12:34:56.123456' + 1;`
		`+-------------------------------------------+`
		`| timestamp '2020-10-10 12:34:56.123456' + 1 |`
		`+-------------------------------------------+`
		`| 20201010123457.123456 |`
		`+-------------------------------------------+`
		`1 row in set (0.00 sec)`
		
		`-- GaussDB: timestamp+numeric. Convert the numeric type to the interval type (1 day), and then add them up to obtain a new timestamp.`
		`gaussdb=# SELECT timestamp '2020-10-10 12:34:56.123456' + 1;`
		`?column?`
		`---------------------------`
		`2020-10-11 12:34:56.123456`
		`(1 row)`
		If the calculation result of the timestamp type and numeric type is used as the input parameter of a function, the result of the function may be different from that of MySQL.
		• Type conversion:
		Compared with MySQL, GaussDB supports only the conversion between the timestamp type and the char(n), varchar(n), or datetime type, and the conversion from timestamp to date or time (only value assignment and explicit conversion). Conversion between binary, decimal, json, integer, and unsigned integer types is not supported. The principles for determining common types in scenarios such as collections and complex expressions are different from those in MySQL. For details, see **Data Type Conversion**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
		• Time zone In GaussDB, the timestamp value can carry the time zone information (time zone offset or time zone name), for example, '2020-01-01 12:34:56.123456 +01:00' or '2020-01-01 2:34:56.123456 CST'. GaussDB converts the time to the time of the current server time zone. If the time zone of the server is changed, the timestamp value is converted to the timestamp of the new time zone. MySQL 5.7 does not support this function. MySQL 8.0 and later versions support this function. • The table columns of the timestamp data type in GaussDB are actually converted to the timestamp(p) with time zone. When you query the table information or use a tool to export the table definition, the data type of columns is timestamp(p) with time zone instead of timestamp. For MySQL, timestamp(p) is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
TIME[(fsp)]	Supported, with differences	GaussDB supports the time data type and differs from MySQL in terms of the following specifications: ● Input format: – It supports only the character type and does not support the numeric type. For example, '1 10:20:30' or '102030' is supported, but 102030 is not supported. MySQL supports conversion from numeric input to the time type. – Separator: GaussDB supports only colons (:) as separators between hours, minutes, and seconds. MySQL supports all symbols as separators. – No separator: The complete format 'hhmiss.ffffff' is recommended. The parsing rules of incomplete formats (including the ultra-long format) may be different from those of MySQL. An error may be reported or the parsing result may be inconsistent with that of MySQL. Therefore, the incomplete format is not recommended. – When a negative value is entered for minute, second, or precision, GaussDB may ignore the first part of the negative value, which is parsed as 0. For example, '00:00:-10' is parsed as '00:00:00'. An error may also be reported. For example, if '00:00:-10000' is parsed, an error will be reported. The result depends on the range of the input value. However, MySQL reports an error in both cases. ● Output format: The format is hh:mi:ss.ffffff, which is the same as that of MySQL. However, for the precision part, if the last several digits are 0, they are not displayed in GaussDB but displayed in MySQL. ● Value range: –838:59:59.000000 to 838:59:59.000000, which is the same as that of MySQL. For values that exceed the range, when GaussDB performs DML operations such as SELECT, INSERT, and UPDATE in loose mode, it returns the nearest boundary values such as **-838:59:59** or **838:59:59**. In MySQL, an error is reported during query, or the nearest

MySQL	GaussDB	Difference

		boundary value is returned after a DML operation. As a result, when the time type is used as the input parameter of the function, the results returned by the function are different.
		● Precision:
		The value ranges from 0 to 6. For a table column, the default value is **0**, which is the same as that in MySQL. In the time(p) 'str' expression, GaussDB parses **(p)** as the precision. The default value is **6**, indicating that **'str'** is formatted to the time type based on the precision specified by **p**. MySQL parses it as a time function, **p** is an input parameter, and **'str'** is the alias of the projection column.
		● Operators:
		– GaussDB supports only the comparison operators =, !=, <, <=, >, and >= between time types and returns **true** or **false**. For the addition operation between the time and interval types, the return result is of the time type. For the subtraction operation between the time and interval types, the return result is of the time type. For the subtraction operation between time types, the return result is of the interval type.
		– When the MySQL time type and other numeric types are calculated, the time type is converted to the numeric type, and then the calculation is performed based on the numeric type. The result is also of the numeric type. It is different from GaussDB. For example:
		`-- MySQL: time+numeric. Convert the time type to 123456 and add it to 1. The result is 123457.`
		`mysql> SELECT time '12:34:56' + 1;`
		`+---------------------+`
		`| time '12:34:56' + 1 |`
		`+---------------------+`
		`| 123457 |`
		`+---------------------+`
		`1 row in set (0.00 sec)`
		
		`-- GaussDB: time+numeric. Convert the numeric type to the interval type (1 day), and then add them up to obtain the new time. Because 24 hours are added, the obtained time is still 12:34:56.`
		`gaussdb=# SELECT time '12:34:56' + 1;`
		` ?column?`
		`----------`

MySQL	GaussDB	Difference
		12:34:56 (1 row) If the calculation result of the time type and numeric type is used as the input parameter of a function, the result of the function may be different from that of MySQL. • Type conversion: Compared with MySQL, GaussDB supports only the conversion between the time type and the char(n) or nchar(n) type, and the conversion from datetime or timestamp to the time type. The conversion between the binary, decimal, date, json, integer, and unsigned integer types is not supported. The principles for determining common types in scenarios such as collections and complex expressions are different from those in MySQL. For details, see **Data Type Conversion**.
YEAR[(4)]	Supported, with differences	GaussDB supports the year data type and differs from MySQL in terms of the following specifications: • Operators: – GaussDB supports only the comparison operators =, !=, <, <=, >, and >= between YEAR types and returns **true** or **false**. – GaussDB supports only the arithmetic operators + and - between the YEAR and int4 types and returns integer values. MySQL returns unsigned integer values. • Type conversion: Compared with MySQL, GaussDB supports only the conversion between the year type and int4 type, and supports only the conversion from the int4, varchar, numeric, date, time, timestamp, or timestamptz type to the year type. The principles for determining common types in scenarios such as collections and complex expressions are different from those in MySQL. For details, see **Data Type Conversion**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
INTERVAL	Supported, with differences	GaussDB supports the INTERVAL data type, but INTERVAL is an expression in MySQL. The differences are as follows: ● The date input of the character string type cannot be used as an operation, for example, SELECT '2023-01-01' + interval 1 day. ● In the INTERVAL expr unit syntax, **expr** cannot be a negative integer or floating-point number, for example, SELECT date'2023-01-01' + interval -1 day. ● In the INTERVAL expr unit syntax, **expr** cannot be the input of an operation expression, for example, SELECT date'2023-01-01' + interval 4/2 day. ● When the INTERVAL expression is used for calculation, the return value is of the datetime type. For MySQL, the return value is of the datetime or date type. The calculation logic is the same as that of GaussDB but different from that of MySQL. ● In the INTERVAL expr unit syntax, the value range of **expr** varies with the unit. The maximum value range is [–2147483648, 2147483647]. If the value exceeds the range, an error is reported in strict mode, and a warning is reported in loose mode and **0** is returned. ● In the INTERVAL expr unit syntax, if the number of columns specified by **expr** is greater than the expected number of columns in unit, an error is reported in strict mode, and a warning is reported in loose mode and **0** is returned. For example, if the value of **unit** is **DAY_HOUR**, the expected number of columns is 2. If the value of **expr** is **'1-2-3'**, the expected number of columns is 3.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.3.1.3 String Data Types

Table 3-172 String data types

MySQL	GaussDB	Difference
CHAR[(M)]	Supported, but with differences.	• Input format – The length of parameters and return values of GaussDB user-defined functions cannot be verified. The length of stored procedure parameters cannot be verified. In addition, correct spaces cannot be supplemented when **PAD_CHAR_TO_FULL_LENGTH** is enabled. However, MySQL supports these functions. – GaussDB does not support escape characters or double quotation marks (""). MySQL supports these inputs. • Syntax The CAST(expr as char) syntax of GaussDB cannot convert the input string to the corresponding type based on the string length. It can only be converted to the varchar type. CAST('' as char) and CAST('' as char(0)) cannot convert an empty string to the char(0) type. MySQL supports conversion to the corresponding type by length. • Operator – After performing addition, subtraction, multiplication, division, or modulo operations on a string (that can be converted to a floating-point value) and an integer value, GaussDB returns an integer, while MySQL returns a floating-point value. – If a value is divided by 0, GaussDB reports an error, and MySQL returns **null**. – "~": returns a negative number in GaussDB and an 8-byte unsigned integer in MySQL. – "^": indicates a power in GaussDB and a bitwise XOR in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
VARCHAR(M)	Supported, but with differences.	• Input format – The length of parameters and return values of GaussDB user-defined functions cannot be verified. The length of stored procedure parameters cannot be verified. However, MySQL supports these functions. – The length of temporary variables in GaussDB user-defined functions and stored procedures can be verified, and an error or truncation alarm is reported in strict or loose mode. However, MySQL does not support these functions. – GaussDB does not support escape characters or double quotation marks (""). MySQL supports these inputs. • Operator – After performing addition, subtraction, multiplication, division, or modulo operations on a string (that can be converted to a floating-point value) and an integer value, GaussDB returns an integer, while MySQL returns a floating-point value. – If a value is divided by 0, GaussDB reports an error, and MySQL returns **null**. – "~": returns a negative number in GaussDB and an 8-byte unsigned integer in MySQL. – "^": indicates a power in GaussDB and a bitwise XOR in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
TINYTEXT	Supported, but with differences.	• Input format – In GaussDB, the length of this type cannot exceed 1 GB. If the length exceeds 1 GB, an error is reported. In MySQL, the length of this type cannot exceed 255 bytes. Otherwise, an error is reported in strict mode, and data is truncated and an alarm is generated in loose mode. – GaussDB does not support escape characters or double quotation marks (""). MySQL supports these inputs. • Operator – After performing addition, subtraction, multiplication, division, or modulo operations on a string (that can be converted to a floating-point value) and an integer value, GaussDB returns an integer, while MySQL returns a floating-point value. – If a value is divided by 0, GaussDB reports an error, and MySQL returns **null**. – "~": returns a negative number in GaussDB and an 8-byte unsigned integer in MySQL. – "^": indicates a power in GaussDB and a bitwise XOR in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
TEXT	Supported, but with differences.	• Input format – In GaussDB, the length of this type cannot exceed 1 GB. If the length exceeds 1 GB, an error is reported. In MySQL, the length of this type cannot exceed 65535 bytes. Otherwise, an error is reported in strict mode, and data is truncated and an alarm is generated in loose mode. – GaussDB does not support escape characters or double quotation marks (""). MySQL supports these inputs. • Operator – After performing addition, subtraction, multiplication, division, or modulo operations on a string (that can be converted to a floating-point value) and an integer value, GaussDB returns an integer, while MySQL returns a floating-point value. – If a value is divided by 0, GaussDB reports an error, and MySQL returns **null**. – "~": returns a negative number in GaussDB and an 8-byte unsigned integer in MySQL. – "^": indicates a power in GaussDB and a bitwise XOR in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
MEDIUMTEXT	Supported, but with differences.	• Input format – In GaussDB, the length of this type cannot exceed 1 GB. If the length exceeds 1 GB, an error is reported. In MySQL, the length of this type cannot exceed 16777215 bytes. Otherwise, an error is reported in strict mode, and data is truncated and an alarm is generated in loose mode. – GaussDB does not support escape characters or double quotation marks (""). MySQL supports these inputs. • Operator – After performing addition, subtraction, multiplication, division, or modulo operations on a string (that can be converted to a floating-point value) and an integer value, GaussDB returns an integer, while MySQL returns a floating-point value. – If a value is divided by 0, GaussDB reports an error, and MySQL returns **null**. – "~": returns a negative number in GaussDB and an 8-byte unsigned integer in MySQL. – "^": indicates a power in GaussDB and a bitwise XOR in MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
LONGTEXT	Supported, but with differences.	• Input format – GaussDB supports a maximum of 1 GB, and MySQL supports a maximum of 4 GB minus 1 byte. – GaussDB does not support escape characters or double quotation marks (""). MySQL supports these inputs. • Operator – After performing addition, subtraction, multiplication, division, or modulo operations on a string (that can be converted to a floating-point value) and an integer value, GaussDB returns an integer, while MySQL returns a floating-point value. – If a value is divided by 0, GaussDB reports an error, and MySQL returns **null**. – "~": returns a negative number in GaussDB and an 8-byte unsigned integer in MySQL. – "^": indicates a power in GaussDB and a bitwise XOR in MySQL.
ENUM('value1','value2',...)	Not supported.	-
SET('value1','value2',...)	Not supported.	-

3.3.3.1.4 Binary Data Types

Table 3-173 Binary data types

MySQL	GaussDB	Difference
BINARY[(M)]	Not supported.	-
VARBINARY(M)	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
TINYBLOB	Supported, with differences.	• Value range: In GaussDB, this type is mapped from the BYTEA type. Its length cannot exceed 1 GB. Otherwise, an error is reported. In MySQL, the length of this type cannot exceed 255 bytes. Otherwise, an error is reported in strict mode, and data is truncated and an alarm is generated in loose mode. • Input format: Escape characters and double quotation marks ("") are not supported. • Output format: For the '\0' character, the query result is displayed as "\000". If the getBytes API of the JDBC driver is used, the result is the '\0' character. • Operator: Arithmetic operators (+ - * / %) are not supported. Common logical operators OR, AND, NOT (|| && !) are not supported. Common bitwise operators (~ & | ^) are not supported.
BLOB	Supported, with differences.	• Value range: In GaussDB, this type is mapped from the BYTEA type. Its length cannot exceed 1 GB. Otherwise, an error is reported. In MySQL, the length of this type cannot exceed 65535 bytes. Otherwise, an error is reported in strict mode, and data is truncated and an alarm is generated in loose mode. • Input format: Escape characters and double quotation marks ("") are not supported. • Output format: For the '\0' character, the query result is displayed as "\000". If the getBytes API of the JDBC driver is used, the result is the '\0' character. • Operator: Arithmetic operators (+ - * / %) are not supported. Common logical operators OR, AND, NOT (|| && !) are not supported. Common bitwise operators (~ & | ^) are not supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
MEDIUMBLOB	Supported, with differences.	• Value range: In GaussDB, this type is mapped from the BYTEA type. Its length cannot exceed 1 GB. Otherwise, an error is reported. In MySQL, the length of this type cannot exceed 16777215 bytes. Otherwise, an error is reported in strict mode, and data is truncated and an alarm is generated in loose mode. • Input format: Escape characters and double quotation marks ("") are not supported. • Output format: For the '\0' character, the query result is displayed as "\000". If the getBytes API of the JDBC driver is used, the result is the '\0' character. • Operator: Arithmetic operators (+ - * / %) are not supported. Common logical operators OR, AND, NOT (|| && !) are not supported. Common bitwise operators (~ & | ^) are not supported.
LONGBLOB	Supported, with differences.	• Value range: In GaussDB, this type is mapped from the BYTEA type. Its length cannot exceed 1 GB. For details, see the centralized and distributed specifications of the BYTEA data type. • Input format: Escape characters and double quotation marks ("") are not supported. • Output format: For the '\0' character, the query result is displayed as "\000". If the getBytes API of the JDBC driver is used, the result is the '\0' character. • Operator: Arithmetic operators (+ - * / %) are not supported. Common logical operators OR, AND, NOT (|| && !) are not supported. Common bitwise operators (~ & | ^) are not supported.
BIT[(M)]	Not supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.3.1.5 JSON Data Type

Table 3-174 JSON Data Type

MySQL	GaussDB	Difference
JSON	Supported, with differences	• The JSON types in GaussDB in MYSQL-compatible mode are the same as the native JSON type of GaussDB but greatly different from that of MySQL. Therefore, the JSON types are not listed one by one. • For details about the JSON type behavior of GaussDB in MySQL-compatible mode, see **JSON/JSONB Types**.

3.3.3.1.6 Attributes Supported by Data Types

Table 3-175 Attributes Supported by Data Types

MySQL	GaussDB
NULL	Supported
NOT NULL	Supported
DEFAULT	Supported
ON UPDATE	Supported
PRIMARY KEY	Supported
CHARACTER SET name	Supported
COLLATE name	Supported

3.3.3.1.7 Data Type Conversion

Conversion between different data types is supported. Data type conversion is involved in the following scenarios:

- The data types of operands of operators (such as comparison and arithmetic operators) are inconsistent. It is commonly used for comparison operations in query or join conditions.
- The data types of arguments and parameters are inconsistent when a function is called.
- The data types of target columns to be updated by DML statements (including INSERT, UPDATE, MERGE, and REPLACE) and the defined column types are inconsistent.
- Using CAST(expr AS datatype) can explicitly convert an expression to a data type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- After the target data type of the final projection column is determined by set operations (UNION and EXCEPT), the type of the projection column in each SELECT statement is inconsistent with the target data type.
- In other expression calculation scenarios, the target data type used for comparison or final result is determined based on the data type of different expressions.
  - DECODE
  - CASE WHEN
  - lexpr [NOT] IN (expr_list)
  - BETWEEN AND
  - JOIN USING(a,b)
  - GREATEST and LEAST
  - NVL and COALESCE

GaussDB and MySQL have different rules for data type conversion and target data types. The following examples show the differences between the two processing modes:

- Example 1

```
-- MySQL: The execution result of IN is 0, indicating false. According to the rule, '1970-01-01' is compared with the expressions in the list in sequence. The results are all 0s. Therefore, the final result is 0.
mysql> SELECT '1970-01-01' IN ('1970-01-02', 1, '1970-01-02');
+-------------------------------------------------+
| '1970-01-01' IN ('1970-01-02', 1, '1970-01-02') |
+-------------------------------------------------+
|                                               0 |
+-------------------------------------------------+

-- GaussDB: The execution result of IN is true, which is opposite to the MySQL result. The common type selected based on the rule is int. Therefore, the left expression '1970-01-01' is converted to the int type and compared with the value after the expression in the list is converted to the int type.
-- When '1970-01-01' and '1970-01-02' are converted to the int type, the values are 1970. (In MySQL-compatible mode, invalid characters and the following content are ignored during conversion, and the previous part is converted to the int type.) The comparison result is equal. Therefore, the returned result is true.
gaussdb=# SELECT '1970-01-01' IN ('1970-01-02', 1::int, '1970-01-02') AS result;
 result
--------
 t
(1 row)
```

- Example 2

```
-- In MySQL, when an integer is compared with a string, they are converted to floating-point numbers for comparison.
mysql> SELECT '2.1' = 2, '2.1' > 2, '2.1' < 2, '.123' = 0;
+-----------+-----------+-----------+------------+
| '2.1' = 2 | '2.1' > 2 | '2.1' < 2 | '.123' = 0 |
+-----------+-----------+-----------+------------+
|         0 |         1 |         0 |          0 |
+-----------+-----------+-----------+------------+
1 row in set (0.00 sec)

-- In GaussDB, when an integer is compared with a string, the string is converted to an integer before the comparison. For instance, when '2.1' is converted to an integer, the non-integer part is truncated, resulting in 2; when '.123' is converted to an integer, it is directly converted to 0.
gaussdb=# SELECT '2.1' = 2, '2.1' > 2, '2.1' < 2, '.123' = 0;
 ?column? | ?column? | ?column? | ?column?
----------+----------+----------+----------
 t        | f        | f        | t
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Differences in data type conversion rules:

- The GaussDB clearly defines the conversion rules between different data types.
  - Whether to support conversion: Conversion is supported only when the conversion path of two types is defined in the pg_cast system catalog.
  - Conversion scenarios: conversion in any scenario, conversion only in CAST expressions, and conversion only during value assignment. In scenarios that are not supported, data type conversion cannot be performed even if the conversion path is defined.
- MySQL supports conversion between any two data types.

Due to the preceding differences, when MySQL-based applications are migrated to GaussDB, an error may be reported because the SQL statement does not support the conversion between different data types. In the scenario where conversion is supported, different conversion rules result in different execution results of SQL statements.

You are advised to use the same data type in SQL statements for comparison or value assignment to avoid unexpected results or performance loss caused by data type conversion.

Differences in target data type selection rules:

In some scenarios, the data type to be compared or returned can be determined only after the types of multiple expressions are considered. For example, in the UNION operation, projection columns at the same position in different SELECT statements are of different data types. The final data type of the query result needs to be determined based on the data type of the projection columns in each SELECT statement.

GaussDB and MySQL have different rules for determining the target data types.

- GaussDB rules:
  - If the operand types of operators are inconsistent, the operand types are not converted to the target type before calculation. Instead, operators of two data types are directly registered, and two types of processing rules are defined during operator processing. In this mode, implicit type conversion does not exist, but the customized processing rule implies the conversion operation.
  - Rules for determining the target data type in the set operation and expression scenarios:
    - If all types are the same, it is the target type.
    - If the two data types are different, check whether the data types are of the same type, such as the numeric type, character type, and date and time type. If they do not belong to the same type, the target type cannot be determined. In this case, an error is reported during SQL statement execution.
    - For data types with the same **category** attribute (defined in the pg_type system catalog), the data type with the **preferred** attribute (defined in the pg_type system catalog) is selected as the target type. If operand 1 can be converted to operand 2 (no conversion

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

path), but operand 2 cannot be converted to operand 1 or the priority of the numeric type is lower than that of operand 2, then operand 2 is selected as the target type.

- If three or more data types are involved, the rule for determining the target type is as follows: common_type(type1,type2,type3) = common_type(common_type(type1,type2),type3). Perform iterative processing in sequence to obtain the final result.
- For IN and NOT IN expressions, if the target type cannot be determined based on the preceding rules, each expression in **lexpr** and **expr_list** is compared one by one based on the equivalent operator (=).
- Precision determination: The precision of the finally selected expression is used as the final result.

- MySQL rules:
  - If the operand types of operators are inconsistent, determine the target type based on the following rules. Then, convert the inconsistent operand types to the target type and then process the operands.
    - If both parameters are of the string type, they are compared based on the string type.
    - If both parameters are of the integer type, they are compared based on the integer type.
    - If a hexadecimal value is not compared with a numeric value, they are compared based on the binary string.
    - If one parameter is of the datetime/timestamp type, and the other parameter is a constant, the constant is converted to the timestamp type for comparison.
    - If one parameter is of the decimal type, the data type used for comparison depends on the other parameter. If the other type is decimal or integer, the decimal type is used. If the other type is not decimal, the real type is used.
    - In other scenarios, the data type is converted to the real type for comparison.
  - Rules for determining the target data type in the set operation and expression scenarios:
    - Establish a target type matrix between any two types. Given two types, the target type can be determined by using the matrix.
    - If three or more data types are involved, the rule for determining the target type is as follows: common_type(type1,type2,type3) = common_type(common_type(type1,type2),type3). Perform iterative processing in sequence to obtain the final result.
    - If the target type is integer and each expression type contains signed and unsigned integers, the type is promoted to an integer type with higher precision. The result is unsigned only when all expressions are unsigned. Otherwise, the result is signed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The highest precision in the expression is used as the final result.

According to the preceding rules, GaussDB and MySQL differ greatly in data type conversion rules and types cannot be directly compared. In the preceding scenario, the execution result of SQL statements may be different from that in MySQL. In the current version, you are advised to use the same type for all expressions or use CAST to convert the type to the required type in advance to avoid differences.

3.3.3.2 System Functions

3.3.3.2.1 System Function Compatibility Overview

GaussDB is compatible with most MySQL system functions, but there are some differences.

Unless otherwise specified, the function behavior in MYSQL-compatible mode is the native GaussDB behavior by default.

NOTE

For most GaussDB system functions compatible with MySQL, the return value precision (the number of trailing zeros in the result) is inconsistent with that of MySQL. This is because the precision of some data types is lost in some scenarios and the precision cannot be correctly obtained. As a result, some functions are not fully adapted.

3.3.3.2.2 Flow Control Functions

Table 3-176 Flow control functions

MySQL	GaussDB	Difference
IF()	Supported, with differences.	• The **expr1** input parameter supports only the Boolean type. If an input parameter of the non-BOOL type cannot be converted to the BOOL type, an error is reported. • If the types of **expr2** and **expr3** are different and no implicit conversion function exists between the two types, an error is reported. • If the two input parameters are of the same type, the input parameter type is returned. • If the **expr2** and **expr3** input parameters are of the NUMERIC, STRING, or TIME type respectively, GaussDB outputs the TEXT type, while MySQL outputs the VARCHAR type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
IFNULL()	Supported, with differences.	• If the types of **expr1** and **expr2** are different and no implicit conversion function exists between the two types, an error is reported. • If the two input parameters are of the same type, the input parameter type is returned. • If the **expr1** and **expr2** input parameters are of the NUMERIC, STRING, or TIME type respectively, GaussDB outputs the TEXT type, while MySQL outputs the VARCHAR type. • If one input parameter is of the FLOAT4 type and the other is of any type in the numeric category, GaussDB returns the DOUBLE type. In MySQL, if one input parameter is of FLOAT4 type and the other is of the TINYINT, UNSIGNED TINYINT, SMALLINT, UNSIGNED SMALLINT, MEDIUMINT, UNSIGNED MEDIUMINT, or BOOL type, the FLOAT4 type is returned. If the first is of FLOAT4 type and the second is of BIGINT or UNSIGNED BIGINT type, the FLOAT type is returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference

NULLIF()	Supported, with differences.	• The NULLIF() type derivation in GaussDB complies with the following logic:
		– If the data types of two parameters are different and the two input parameter types have an equality comparison operator, the left value type corresponding to the equality comparison operator is returned. Otherwise, the two input parameter types are forcibly compatible.
		– If an equality comparison operator exists after forcible type compatibility, the left value type of the equality comparison operator after forcible type compatibility is returned.
		– If the corresponding equality operator cannot be found after forcible type compatibility, an error is reported.
		-- The two input parameter types have an equality comparison operator.
		gaussdb=# SELECT pg_typeof(nullif(1::int2, 2::int8));
		pg_typeof
		----------- 	
		smallint
		(1 row)
		-- The two input parameter types do not have the equality comparison operator, but the equality comparison operator can be found after forcible type compatibility.
		gaussdb=# SELECT pg_typeof(nullif(1::int1, 2::int2));
		pg_typeof
		----------- 	
		bigint
		(1 row)
		
		-- The two input parameter types do not have the equality comparison operator, and the equality comparison operator does not exist after forcible type compatibility.
		gaussdb=# SELECT nullif(1::bit, '1'::MONEY);
		ERROR: operator does not exist: bit = money
		LINE 1: SELECT nullif(1::bit, '1'::MONEY);
		^
		HINT: No operator matches the given name and argument type(s). You might need to add explicit type casts.
		CONTEXT: referenced column: nullif
		• The MySQL output type is related only to the type of the first input parameter.
		– If the type of the first input parameter is TINYINT, SMALLINT, MEDIUMINT, INT, or BOOL, the output is of the INT type.
		– If the type of the first input parameter is BIGINT, the output is of the BIGINT type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
		– When the type of the first input parameter is UNSIGNED TINYINT, UNSIGNED SMALLINT, UNSIGNED MEDIUMINT, UNSIGNED INT, or BIT, the output is of the UNSIGNED INT type. – If the type of the first input parameter is UNSIGNED BIGINT, the output is of the UNSIGNED BIGINT type. – If the type of the first input parameter is of the FLOAT, DOUBLE, or REAL type, the output is of the DOUBLE type. – If the type of the first input parameter DECIMAL or NUMERIC, the output is of the DECIMAL type. – If the type of the first input parameter is DATE, TIME, DATETIME, TIMESTAMP, CHAR, VARCHAR, TINYTEXT, ENUM, or SET, the output is of the VARCHAR type. – If the type of the first input parameter is TEXT, MEDIUMTEXT, or LONGTEXT, the output is of the LONGTEXT type. – If the type of the first input parameter is TINYBLOB, the output is of the VARBINARY type. – If the type of the first input parameter is MEDIUMBLOB or LONGBLOB, the output is of the LONGBLOB type. – If the type of the first input parameter is BLOB, the output is of the BLOB type.
ISNULL()	Supported, with differences.	In GaussDB, the return value is **t** or **f** of the BOOLEAN type. In MySQL, the return value is **1** or **0** of the INT type.

3.3.3.2.3 Date and Time Functions

The date and time functions in GaussDB in MYSQL-compatible mode, with the same behavior as MySQL, are described as follows:

- Functions may use time expressions as their input parameters.

  Time expressions mainly include text, datetime, date, and time. Besides, all types that can be implicitly converted to time expressions can be input parameters. For example, a number can be implicitly converted to text and then used as a time expression.

  However, different functions take effect in different ways. For example, the datediff function calculates only the difference between dates. Therefore, time expressions are parsed as the date type. The timestampdiff function parses

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

time expressions as date, time, or datetime based on the **unit** parameter before calculating the time difference.

- The input parameters of functions may contain an invalid date.

  Generally, the supported date and datetime ranges are the same as those of MySQL. The value of date ranges from '0000-01-01' to '9999-12-31', and the value of datetime ranges from '0000-01-01 00:00:00' to '9999-12-31 23:59:59'. Although GaussDB supports larger date and datetime ranges than MySQL, dates out of range are still considered invalid.

  In most cases, time functions report an alarm and return NULL if the input date is invalid, unless the invalid date can be converted by CAST.

- Separators for input parameters of functions:

  For a time function, all non-digit characters are regarded as separators when input parameters are processed. The standard format is recommended: Use hyphens (-) to separate year, month, and day, use colons (:) to separate hour, minute, and second, and use a period (.) before milliseconds.

  Error-prone scenario: When **SELECT timestampdiff(hour, '2020-03-01 00:00:00', '2020-02-28 00:00:00+08');** is executed in a MYSQL-compatible database, the time function does not automatically calculate the time zone. Therefore, +08 is not identified as the time zone. Instead, + is used as the separator for calculation as seconds.

Most function scenarios of GaussDB date and time functions are the same as those of MySQL, but there are still differences. Some differences are as follows:

- If an input parameter of a function is NULL, the function returns NULL, and no warning or error is reported. These functions include:

  from_days, date_format, str_to_date, datediff, timestampdiff, date_add, subtime, month, time_to_sec, to_days, to_seconds, dayname, monthname, convert_tz, sec_to_time, addtime, adddate, date_sub, timediff, last_day, weekday, from_unixtime, unix_timestamp, subdate, day, year, weekofyear, dayofmonth, dayofyear, week, yearweek, dayofweek, time_format, hour, minute, second, microsecond, quarter, get_format, extract, makedate, period_add, timestampadd, period_diff, utc_time, utc_timestamp, maketime, and curtime.

  Example:

  ```
  gaussdb=# SELECT day(null);
   day
  -----

  (1 row)
  ```

- Some functions with pure numeric input parameters are different from those of MySQL. Numeric input parameters without quotation marks are converted into text input parameters for processing.

  Example:

  ```
  gaussdb=# SELECT day(19231221.123141);
  WARNING:  Incorrect datetime value: "19231221.123141"
  CONTEXT:  referenced column: day
   day
  -----

  (1 row)
  ```

- Time and date calculation functions are adddate, subdate, date_add, and date_sub. If the calculation result is a date, the supported range is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

[0000-01-01,9999-12-31]. If the calculation result is a date and time, the supported range is [0000-01-01 00:00:00.000000,9999-12-31 23:59:59.999999]. If the calculation result exceeds the supported range, an ERROR is reported in strict mode, or a WARNING is reported in loose mode. If the date result after calculation is within the range [0000-01-01,0001-01-01], GaussDB returns the result normally. MySQL returns '0000-00-00'.

Example:

```
gaussdb=# SELECT subdate('0000-01-01', interval 1 hour);
ERROR:  Datetime function: datetime field overflow
CONTEXT:  referenced column: subdate

gaussdb=# SELECT subdate('0001-01-01', interval 1 day);
  subdate
-------------
 0000-12-31

(1 row)
```

- If the input parameter of the date or datetime type of the date and time function contains month 0 or day 0, the value is invalid. In strict mode, an error is reported. In loose mode, if the input is a character string or number, a warning is reported. If the input is of the date or datetime type, the system processes the input as December of the previous year or the last day of the previous month.

  If the type of the CAST function is converted to date or datetime, an error is reported in strict mode. In loose mode, no warning is reported. Instead, the system processes the input as December of the previous year or the last day of the previous month. Pay attention to this difference. MySQL outputs the value as it is, even if the year, month, and day are set to **0**.

  Example:

```
gaussdb=# SELECT adddate('2023-01-00', 1);-- Strict mode
ERROR:  Incorrect datetime value: "2023-01-00"
CONTEXT:  referenced column: adddate

gaussdb=# SELECT adddate('2023-01-00', 1); -- Loose mode
WARNING:  Incorrect datetime value: "2023-01-00"
CONTEXT:  referenced column: adddate
 adddate
---------

(1 row)

gaussdb=# SELECT adddate(date'2023-00-00', 1); -- Loose mode
  adddate
------------
 2022-12-01
(1 row)

gaussdb=# SELECT cast('2023/00/00' as date); -- Loose mode
    date
------------
 2022-11-30
(1 row)

gaussdb=# SELECT cast('0000-00-00' as datetime); -- Loose mode
      timestamp
---------------------
 0000-00-00 00:00:00
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If the input parameter of the function is of the numeric data type, no error is reported in the case of invalid input, and the input parameter is processed as 0.

  Example:

```
gaussdb=# SELECT from_unixtime('aa');
    from_unixtime
---------------------
 1970-01-01 08:00:00
(1 row)
```

- A maximum of six decimal places are allowed. Decimal places with all 0s are not allowed.

  Example:

```
gaussdb=# SELECT from_unixtime('1234567899.00000');
    from_unixtime
---------------------
 2009-02-14 07:31:39
(1 row)
```

- If the time function parameter is a character string, the result is correct only when the year, month, and day are separated by a hyphen (-) and the hour, minute, and second are separated by a colon (:).

  Example:

```
gaussdb=# SELECT adddate('20-12-12',interval 1 day);
  adddate
------------
 2020-12-13
(1 row)
```

- If the return value of a function is of the varchar type in MySQL, the return value of the function is of the text type in GaussDB.

```
-- Return value of a function in GaussDB.
gaussdb=# SELECT pg_typeof(adddate('2023-01-01', 1));
 pg_typeof
-----------
 text
(1 row)

-- Return value of a function in MySQL.
mysql> CREATE VIEW v1 AS SELECT adddate('2023-01-01', 1);
Query OK, 0 rows affected (0.00 sec)

mysql> DESC v1;
+--------------------------+-------------+------+-----+---------+-------+
| Field                    | Type        | Null | Key | Default | Extra |
+--------------------------+-------------+------+-----+---------+-------+
| adddate('2023-01-01', 1) | varchar(29) | YES  |     | NULL    |       |
+--------------------------+-------------+------+-----+---------+-------+
1 row in set (0.00 sec)
```

Table 3-177 Date and time functions

MySQL	GaussDB	Difference
ADDDATE()	Supported, with differences	The performance of this function is different from that of MySQL due to interval expression differences. For details, see **INTERVAL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
ADDTIME()	Supported, with differences	• MySQL returns NULL if the second input parameter is a string in the DATETIME format. GaussDB can calculate the value. • The value range of an input parameter is ['0001-01-01 00:00:00', 9999-12-31 23:59:59.999999]. • If the first parameter of the ADDTIME function in MySQL is a dynamic parameter (for example, in a prepared statement), the return type is TIME. Otherwise, the parse type of the function is derived from the parse type of the first parameter. The return value rules of the ADDTIME function in GaussDB are as follows: – The first input parameter is of the date type, the second input parameter is of the date type, and the return value is of the time type. – The first input parameter is of the date type, the second input parameter is of the text type, and the return value is of the text type. – The first input parameter is of the date type, the second input parameter is of the datetime type, and the return value is of the time type. – The first input parameter is of the date type, the second input parameter is of the time type, and the return value is of the time type. – The first input parameter is of the text type, the second input parameter is of the date type, and the return value is of the text type. – The first input parameter is of the text type, the second input parameter is of the text type, and the return value is of the text type. – The first input parameter is of the text type, the second input parameter is of the datetime type, and the return value is of the text type. – The first input parameter is of the text type, the second input parameter is of the time type, and the return value is of the text type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
		– The first input parameter is of the datetime type, the second input parameter is of the date type, and the return value is of the datetime type. – The first input parameter is of the datetime type, the second input parameter is of the text type, and the return value is of the text type. – The first input parameter is of the datetime type, the second input parameter is of the datetime type, and the return value is of the datetime type. – The first input parameter is of the datetime type, the second input parameter is of the time type, and the return value is of the datetime type. – The first input parameter is of the time type, the second input parameter is of the date type, and the return value is of the time type. – The first input parameter is of the time type, the second input parameter is of the text type, and the return value is of the text type. – The first input parameter is of the time type, the second input parameter is of the datetime type, and the return value is of the time type. – The first input parameter is of the time type, the second input parameter is of the time type, and the return value is of the time type.
CONVERT_TZ()	Supported	-
CURDATE()	Supported	-
CURRENT_DATE(), CURRENT_DATE	Supported	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
CURRENT_TIME(), CURRENT_TIME	Supported, with differences	The time value (after the decimal point) output by precision is rounded off in GaussDB and directly truncated in MySQL. The trailing 0s of the time value (after the decimal point) output by precision are not displayed in GaussDB but displayed in MySQL. GaussDB supports only an integer value within the range of [0,6] as the precision of the returned time. For other values, an error is reported. In MySQL, a precision value within [0,6] is valid, but an input integer value is divided by 256 to get a remainder. For example, if the input integer value is **257**, the time value with precision 1 is returned.
CURRENT_TIMESTAMP(), CURRENT_TIMESTAMP	Supported, with differences	The time value (after the decimal point) output by precision is rounded off in GaussDB and directly truncated in MySQL. The trailing 0s of the time value (after the decimal point) output by precision are not displayed in GaussDB but displayed in MySQL. GaussDB supports only an integer value within the range of [0,6] as the precision of the returned time. If the input integer value is greater than 6, an alarm is generated and the time value is output based on the precision 6. In MySQL, a precision value within [0,6] is valid, but an input integer value is divided by 256 to get a remainder. For example, if the input integer value is **257**, the time value with precision 1 is returned.
CURTIME()	Supported, with differences	In GaussDB, if a character string or a non-integer value is entered, the value is implicitly converted into an integer and then the precision is verified. If the value is beyond the [0,6] range, an error is reported. If the value is within the range, the time value is output normally. In MySQL, an error is reported. The time value (after the decimal point) output by precision is rounded off in GaussDB and directly truncated in MySQL. The trailing 0s of the time value (after the decimal point) output by precision are not displayed in GaussDB but displayed in MySQL. GaussDB supports only an integer value within the range of [0,6] as the precision of the returned time. For other values, an error is reported. In MySQL, a precision value within [0,6] is valid, but an input integer value is divided by 256 to get a remainder. For example, if the input integer value is **257**, the time value with precision 1 is returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
YEARWEEK()	Supported	-
DATE_ADD()	Supported, with differences	The performance of this function is different from that of MySQL due to interval expression differences. For details, see **INTERVAL**.
DATE_FORMAT()	Supported	-
DATE_SUB()	Supported, with differences	The performance of this function is different from that of MySQL due to interval expression differences. For details, see **INTERVAL**.
DATEDIFF()	Supported	-
DAY()	Supported	-
DAYNAME()	Supported	-
DAYOFMONTH()	Supported	-
DAYOFWEEK()	Supported	-
DAYOFYEAR()	Supported	-
EXTRACT()	Supported	-
FROM_DAYS()	Supported	-
FROM_UNIXTIME()	Supported	-
GET_FORMAT()	Supported	-
HOUR()	Supported	-
LAST_DAY	Supported	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
LOCALTIME(), LOCALTIME	Supported, with differences	The time value (after the decimal point) output by precision is rounded off in GaussDB and directly truncated in MySQL. The trailing 0s of the time value (after the decimal point) output by precision are not displayed in GaussDB but displayed in MySQL. GaussDB supports only an integer value within the range of [0,6] as the precision of the returned time. For other integer values, an error is reported. In MySQL, a precision value within [0,6] is valid, but an input integer value is divided by 256 to get a remainder. For example, if the input integer value is **257**, the time value with precision 1 is returned.
LOCALTIMESTAMP, LOCALTIMESTAMP()	Supported, with differences	The time value (after the decimal point) output by precision is rounded off in GaussDB and directly truncated in MySQL. The trailing 0s of the time value (after the decimal point) output by precision are not displayed in GaussDB but displayed in MySQL. GaussDB supports only an integer value within the range of [0,6] as the precision of the returned time. If the input integer value is greater than 6, an alarm is generated and the time value is output based on the precision 6. In MySQL, a precision value within [0,6] is valid, but an input integer value is divided by 256 to get a remainder. For example, if the input integer value is **257**, the time value with precision 1 is returned.
MAKEDATE()	Supported	-
MAKETIME()	Supported, with differences	When the input parameter is NULL, GaussDB does not support self-nesting of the maketime function, but MySQL supports.
MICROSECOND()	Supported	-
MINUTE()	Supported	-
MONTH()	Supported	-
MONTHNAME()	Supported	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
NOW()	Supported, with differences	The time value (after the decimal point) output by precision is rounded off in GaussDB and directly truncated in MySQL. The trailing 0s of the time value (after the decimal point) output by precision are not displayed in GaussDB but displayed in MySQL. GaussDB supports only an integer value within the range of [0,6] as the precision of the returned time. If the input integer value is greater than 6, an alarm is generated and the time value is output based on the precision 6. In MySQL, a precision value within [0,6] is valid, but an input integer value is divided by 256 to get a remainder. For example, if the input integer value is **257**, the time value with precision 1 is returned.
PERIOD_ADD()	Supported, with differences	If the input parameter **period** or result is less than 0, GaussDB reports an error by referring to the performance in MySQL 8.0.x. Integer wrapping occurs in MySQL 5.7. As a result, the calculation result is abnormal.
PERIOD_DIFF()	Supported, with differences	If the input parameter or result is less than 0, GaussDB reports an error by referring to the performance in MySQL 8.0.x. Integer wrapping occurs in MySQL 5.7. As a result, the calculation result is abnormal.
QUARTER()	Supported	-
SEC_TO_TIME()	Supported	-
SECOND()	Supported	-
STR_TO_DATE()	Supported, with differences	GaussDB returns values of the text type, while MySQL returns values of the datetime or date type.
SUBDATE()	Supported, with differences	The performance of this function is different from that of MySQL due to interval expression differences. For details, see **INTERVAL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
SUBTIME()	Supported, with differences	• MySQL returns NULL if the second input parameter is a string in the DATETIME format. GaussDB can calculate the value. • The value range of an input parameter is ['0001-01-01 00:00:00', 9999-12-31 23:59:59.999999]. • If the first parameter of the SUBTIME function in MySQL is a dynamic parameter (for example, in a prepared statement), the return type is TIME. Otherwise, the parse type of the function is derived from the parse type of the first parameter. The return value rules of the SUBTIME function in GaussDB are as follows: – The first input parameter is of the date type, the second input parameter is of the date type, and the return value is of the time type. – The first input parameter is of the date type, the second input parameter is of the text type, and the return value is of the text type. – The first input parameter is of the date type, the second input parameter is of the datetime type, and the return value is of the time type. – The first input parameter is of the date type, the second input parameter is of the time type, and the return value is of the time type. – The first input parameter is of the text type, the second input parameter is of the date type, and the return value is of the text type. – The first input parameter is of the text type, the second input parameter is of the text type, and the return value is of the text type. – The first input parameter is of the text type, the second input parameter is of the datetime type, and the return value is of the text type. – The first input parameter is of the text type, the second input parameter is of the time type, and the return value is of the text type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
		– The first input parameter is of the datetime type, the second input parameter is of the date type, and the return value is of the datetime type. – The first input parameter is of the datetime type, the second input parameter is of the text type, and the return value is of the text type. – The first input parameter is of the datetime type, the second input parameter is of the datetime type, and the return value is of the datetime type. – The first input parameter is of the datetime type, the second input parameter is of the time type, and the return value is of the datetime type. – The first input parameter is of the time type, the second input parameter is of the date type, and the return value is of the time type. – The first input parameter is of the time type, the second input parameter is of the text type, and the return value is of the text type. – The first input parameter is of the time type, the second input parameter is of the datetime type, and the return value is of the time type. – The first input parameter is of the time type, the second input parameter is of the time type, and the return value is of the time type.
SYSDATE()	Supported, with differences	In MySQL, an integer input value is wrapped when it reaches **255** (maximum value of a one-byte integer value), while GaussDB does not.
YEAR()	Supported	-
TIME_FORMAT()	Supported	-
TIME_TO_SEC()	Supported	-
TIMEDIFF()	Supported	-
WEEKOFYEAR()	Supported	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
TIMESTAMPADD()	Supported	-
TIMESTAMPDIFF()	Supported	-
TO_DAYS()	Supported	-
TO_SECONDS()	Supported	-
UNIX_TIMESTAMP()	Supported, with differences	GaussDB returns values of the numeric type, while MySQL returns values of the int type.
UTC_DATE()	Supported, with differences	• MySQL supports calling without parentheses, but GaussDB does not. In MySQL, an integer input value is wrapped when it reaches **255** (maximum value of a one-byte integer value). • MySQL input parameters support only integers ranging from 0 to 6. GaussDB supports input parameters that can be implicitly converted to integers ranging from 0 to 6.
UTC_TIME()	Supported, with differences	
UTC_TIMESTAMP()	Supported, with differences	
WEEK()	Supported	-
WEEKDAY()	Supported	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.3.2.4 String Functions

Table 3-178 String functions

MySQL	GaussDB	Difference
BIN()	Supported, with differences.	In GaussDB, the types supported by function input parameters are as follows: • Integer types: tinyint, smallint, mediumint, int, and bigint. • Unsigned integer types: tinyint unsigned, smallint unsigned, int unsigned, and bigint unsigned. • Character and text types: char, varchar, tinytext, text, mediumtext, and longtext. Only numeric integer strings are supported, and the integer range is within the bigint range. • Floating-point types: float, real, and double. • Fixed-point types: numeric, decimal, and dec. • Boolean type: bool.
CONCAT()	Supported, with differences.	The data type of the return value of CONCAT is always text regardless of the data type of the parameter. However, in MySQL, if CONCAT contains binary parameters, the return value is binary.
CONCAT_WS()	Supported, with differences.	The data type of the return value of CONCAT_WS is always text regardless of the data type of the parameter. However, in MySQL, if CONCAT_WS contains binary parameters, the return value is binary. In other cases, the return value is a string.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
ELT()	Supported, with differences.	• In GaussDB, the types supported by function input parameter 1 are as follows: – Integer types: tinyint, smallint, mediumint, int, and bigint. – Unsigned integer types: tinyint unsigned, smallint unsigned, and int unsigned. – Character and text types: char, varchar, tinytext, text, mediumtext, and longtext. Only numeric integer strings are supported, and the integer range is within the bigint range. – Floating-point types: float, real, and double. – Fixed-point types: numeric, decimal, and dec. – Boolean type: bool. • In GaussDB, the types supported by function input parameter 2 are as follows: – Integer types: tinyint, smallint, mediumint, int, and bigint. – Unsigned integer types: tinyint unsigned, smallint unsigned, int unsigned, and bigint unsigned. – Character and text types: char, varchar, tinytext, text, mediumtext, and longtext. – Floating-point types: float, real, and double. – Fixed-point types: numeric, decimal, and dec. – Boolean type: bool. – Large object types: tinyblob, blob, mediumblob, and longblob. – Date types: datetime, timestamp, date, and time.
FIELD()	Supported, with differences.	When function input parameters range from the maximum bigint value to the maximum bigint unsigned value, incompatibility occurs. When function input parameters are of the float(m, d), double(m, d), or real(m, d) type, the precision is higher and incompatibility occurs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
FIND_IN_SET()	Supported, with differences.	When the database encoding is set to **'SQL_ASCII'**, the default case sensitivity rule is not supported. That is, if no character set rule is specified, uppercase and lowercase letters are treated as distinct.
INSERT()	Supported, with differences.	• The range of input parameters of the Int64 type is from –9223372036854775808 to +9223372036854775807. If a value is out of range, an error is reported. MySQL does not limit the range of input parameters of the numeric type. If an exception occurs, an alarm is generated, indicating that the value is set to the upper or lower limit. • The maximum length of the input parameter of the text type is 2^30 – 5 bytes, and the maximum length of the input parameter of the BYTEA type is 2^30 – 512 bytes. • If any of the **s1** and **s2** parameters is of the BYTEA type and the result contains invalid characters, the displayed result may be different from that of MySQL, but the character encoding is the same as that of MySQL.
LOCATE()	Supported, with differences.	When input parameter 1 is of the BYTEA type and input parameter 2 is of the text type, the behavior of GaussDB is different from that of MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
MAKE_SET()	Supported, with differences.	• When the **bits** parameter is an integer, the maximum range is int128, which is smaller than the MySQL range. • When the **bits** parameter is of the date type (datetime, timestamp, date, or time), it is not supported because the conversion from the date type to the integer type is different from that in MySQL. • GaussDB and MySQL are inherently different in the bit and Boolean types, causing different returned results. When the **bits** input parameter is of the Boolean type, and the **str** input parameter is of the bit or Boolean type, they are not supported. • When the **bits** input parameter is of the character string or text type, only the pure integer format is supported. In addition, the value range of pure integers is limited to bigint. • The integer value of the **str** input parameter exceeds the range from 81 negative nines to 81 positive nines. The return value is different from that of MySQL. • When the **str** input parameter is expressed in scientific notation, trailing zeros are displayed in GaussDB but not displayed in MySQL. This is an inherent difference.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
QUOTE()	Supported, with differences.	• If the **str** character string contains "\Z", "\r", "\%", or "_", GaussDB does not escape it, which is different from MySQL. The slash followed by digits may also cause differences, for example, "\563". This function difference is the escape character difference between GaussDB and MySQL. • The output format of "\b" in the **str** character string is different from that in MySQL. This is an inherent difference between GaussDB and MySQL. • If the **str** character string contains "\0", GaussDB cannot identify the character because the UTF-8 character set cannot identify the character. As a result, the input fails. This is an inherent difference between GaussDB and MySQL. • If **str** is of the bit or Boolean type, this type is not supported because it is different in GaussDB and MySQL. • GaussDB transmits up to 1 GB data. The maximum length of the **str** input parameter is 536870908 bytes, and the maximum length of the result string returned by the function is 1 GB. • The integer value of the **str** input parameter exceeds the range from 81 negative nines to 81 positive nines. The return value is different from that of MySQL. • When the **str** input parameter is expressed in scientific notation, trailing zeros are displayed in GaussDB but not displayed in MySQL. This is an inherent difference.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
SPACE()	Supported, with differences.	• GaussDB allows an input parameter of no more than 1073741818 bytes. If the length exceeds the limit, an empty string is returned. By default, MySQL allows an input parameter of no more than 4194304 bytes. If the length exceeds the limit, an alarm is generated. • In GaussDB, the types supported by function input parameters are as follows: – Integer types: tinyint, smallint, mediumint, int, and bigint. – Unsigned integer types: tinyint unsigned, smallint unsigned, and int unsigned. – Character and text types: char, varchar, tinytext, text, mediumtext, and longtext. Only numeric integer strings are supported, and the integer range is within the bigint range. – Floating-point types: float, real, and double. – Fixed-point types: numeric, decimal, and dec. – Boolean type: bool.
SUBSTR()	Supported.	-
SUBSTRING()	Supported.	-
SUBSTRING_INDEX()	Supported.	-
STRCMP()	Supported, with differences.	• In GaussDB, the types supported by function input parameters are as follows: – Character types: char, varchar, nvarchar2, and text – Binary type: BYTEA – Value type: tinyint [unsigned], smallint [unsigned], integer [unsigned], bigint [unsigned], float4, float8, and numeric – Date and time type: date, time without time zone, datetime, and timestamptz • For the floating-point type in the numeric type, the precision may be different from that in MySQL due to different connection parameter settings. Therefore, this scenario is not recommended, or the NUMERIC type is used instead.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
SHA()/ SHA1()	Supported.	-
SHA2()	Supported.	-

3.3.3.2.5 Forced Conversion Functions

Table 3-179 Forced conversion functions

MySQL	GaussDB	Difference
CAST()	Supported, with differences.	The data type conversion rules and supported conversion types are subject to the conversion scope and rules supported by GaussDB.
CONVERT()	Supported, with differences.	The data type conversion rules and supported conversion types are subject to the conversion scope and rules supported by GaussDB.

3.3.3.2.6 Encryption Functions

Table 3-180 Encryption functions

MySQL	GaussDB	Difference
AES_DECRYPT()	Supported.	-
AES_ENCRYPT()	Supported.	-

3.3.3.2.7 JSON Functions

JSON function differences:

- If you add escape characters as input parameters to JSON functions and other functions that allow character inputs, the processing is different from that in MySQL by default. To be compatible with MySQL, set the GUC parameter **standard_conforming_strings** to **off**. In this case, the processing of escape characters is compatible with MySQL, but a warning is generated for non-standard character input. The escape characters \t and \u and escape digits are different from those in MySQL. The JSON_UNQUOTE() function is compatible with MySQL. Even if the GUC parameter is not set, no alarm is generated.
- When processing an ultra-long number (the number contains more than 64 characters), the JSON function of GaussDB parses the number as a DOUBLE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

and uses scientific notation for counting. The input parameters of the non-JSON type are the same as those of MySQL. However, when input parameters of the JSON type are used, the JSON type is not completely compatible with MySQL. As a result, differences occur in this scenario. MySQL displays complete numbers. (When the number length exceeds 82, MySQL displays an incorrect result.) GaussDB still parses an ultra-long number into a double-precision value. Long numbers are stored using floating-point numbers. During calculation, precision loss occurs in both GaussDB and MySQL. Therefore, you are advised to use character strings to store long numbers.

```
gaussdb=# SELECT json_insert('[1, 4,
99999999999999999999999999999999999999999999999999999999999999999999999999]','$[6]',json
_insert('[1,4]','$[5]',99999999999999999999999999999999999999999999999999999999999999999999999999
999999));
        json_insert
------------------------------
 [1, 4, 1e+74, [1, 4, 1e+74]]
(1 row)
```

Table 3-181 JSON functions

MySQL	GaussDB	Difference
JSON_APPEND()	Supported.	-
JSON_ARRAY()	Supported.	-
JSON_ARRAY_APPEND()	Supported.	-
JSON_ARRAY_INSERT()	Supported.	-
JSON_CONTAINS()	Supported.	-
JSON_CONTAINS_PATH()	Supported.	-
JSON_DEPTH()	Supported, with differences	GaussDB returns values of the int type, while MySQL returns values of the bigint type.
JSON_EXTRACT()	Supported.	-
JSON_INSERT()	Supported.	-
JSON_KEYS()	Supported.	-
JSON_LENGTH()	Supported, with differences	GaussDB returns values of the int type, while MySQL returns values of the bigint type.
JSON_MERGE()	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
JSON_OBJECT()	Supported.	-
JSON_QUOTE()	Supported, with differences	Return value difference: In GaussDB, JSON is returned. In MySQL, varchar or text is returned.
JSON_REMOVE()	Supported.	-
JSON_REPLACE()	Supported.	-
JSON_SEARCH()	Supported, with differences	GaussDB returns values of the text type, while MySQL returns values of the JSON type.
JSON_SET()	Supported.	-
JSON_TYPE()	Supported, with differences	JSON values of the numeric type are identified as number, which is different from MySQL.
JSON_UNQUOTE()	Supported.	-
JSON_VALID()	Supported.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.3.2.8 Aggregate Functions

Table 3-182 Aggregate functions

MySQL	GaussDB	Difference
GROUP_CONCAT()	Supported, with differences	• If the **group_concat** parameter contains both the DISTINCT and ORDER BY syntaxes, all expressions following ORDER BY must be in the DISTINCT expression. • **group_concat(... order by** *Number***)** does not indicate the sequence of the parameter. The number is only a constant expression, which is equivalent to no sorting. • The data type of the return value of **group_concat** is always text regardless of the data type of the parameter. For MySQL, if **group_concat** contains binary parameters, the return value is binary. In other cases, the return value is a character string. If the return value length is greater than 512 bytes, the data type is a character large object or binary large object. • The value of **group_concat_max_len** ranges from 0 to 1073741823. The maximum value is smaller than that of MySQL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
DEFAULT()	Supported, with differences	• The default value of a column is an array. GaussDB returns an array. MySQL does not support the array type. • GaussDB columns are hidden columns (such as **xmin** and **cmin**). The default function returns a null value. • GaussDB supports default values of partitioned tables, temporary tables, and multi-table join query. • GaussDB supports the query of nodes whose column names contain character string values (indicating names) and A_Star nodes (indicating asterisks [*]), for example, **default(tt.t4.id)** and **default(tt.t4.*)**. For invalid query column names and A_Star nodes, the error information reported by GaussDB is different from that reported by MySQL. • When the default value of a column is created in GaussDB, the range of the column type is not verified. As a result, an error may be reported when the default function is used. • If the default value of a column is a function expression, the default function in GaussDB returns the calculated value of the default expression of the column during table creation. The default function in MySQL returns NULL.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.3.2.9 Numeric Operation Functions

Table 3-183 Numeric operation functions

MySQL	GaussDB	Difference
log2()	Supported, but with differences.	• The display of decimal places is different from that in MySQL. Due to the limitation of the GaussDB floating-point data type, the **extra_float_digits** parameter is used to control the number of decimal places to be displayed. • Due to the internal processing difference of the input precision, the calculation results of GaussDB and MySQL are different. • The following data types are supported: – Integer types: bigint, int16, int, smallint, and tinyint. – Unsigned integer types: bigint unsigned, integer unsigned, smallint unsigned, and tinyint unsigned. – Floating-point number type: numeric and real. – Character string type: character, character varying, clob, and text. Only numeric integer strings are supported. – SET type. – NULL type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MySQL	GaussDB	Difference
log10()	Supported, but with differences.	• The display of decimal places is different from that in MySQL. Due to the limitation of the GaussDB floating-point data type, the **extra_float_digits** parameter is used to control the number of decimal places to be displayed. • Due to the internal processing difference of the input precision, the calculation results of GaussDB and MySQL are different. • The following data types are supported: – Integer types: bigint, int16, int, smallint, and tinyint. – Unsigned integer types: bigint unsigned, integer unsigned, smallint unsigned, and tinyint unsigned. – Floating-point number type: numeric and real. – Character string type: character, character varying, clob, and text. Only numeric integer strings are supported. – SET type. – NULL type.
RAND([seed])	Supported, but with differences.	Due to the random number generation algorithm used in the function, the random number returned by the function is different from that in MySQL.

3.3.3.2.10 Other Functions

Table 3-184 Other functions

MySQL	GaussDB	Difference
UUID()	Supported	-
UUID_SHORT()	Supported	-

3.3.3.3 Operators

GaussDB is compatible with most MySQL operators, but there are some differences. Unless otherwise specified, the operator behavior in MYSQL-compatible mode is the native GaussDB behavior by default.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 3-185 Operators

MySQL	GaussDB	Difference
NULL-safe equal (<=>)	Supported.	-
[NOT] REGEXP	Supported, with differences.	• If the GUC parameter **b_format_dev_version** is set to **'s2'** and a pattern string with escape characters such as "\\a", "\\d", "\\e", "\\n", "\\Z", or "\\u" is matched with source character strings "\a", "\d", "\e", "\n", "\Z", or "\u", the behavior of GaussDB is different from that of MySQL 5.7 but the same as that of MySQL 8.0. • When the GUC parameter **b_format_dev_version** is set to **'s2'**, "\b" in GaussDB can match "\\b", but the matching will fail in MySQL. • If the input parameter of the pattern string is invalid with only the right parenthesis ()), GaussDB and MySQL 5.7 will report an error, but MySQL 8.0 will not. • In the rule of matching the de|abc sequence with de or abc, when there are empty values on the left and right of the pipe symbol (|), MySQL 5.7 will report an error, but GaussDB and MySQL 8.0 will not. • The regular expression of the tab character "\t" can match the character class [:blank:] in GaussDB and MySQL 8.0 but cannot in MySQL 5.7. • GaussDB supports non-greedy pattern matching. That is, the number of matching characters is as small as possible. A question mark (?) is added after some special characters, for example, ?? *? +? {n}? {n,}? {n,m}? MySQL 5.7 does not support non-greedy pattern matching, and the error message "Got error 'repetition-operator operand invalid' from regexp" is displayed. MySQL 8.0 already supports this function. • In the BINARY character set, the TEXT and BLOB types are converted to the BYTEA type. The REGEXP operator does not support the BYTEA type. Therefore, the two types cannot be matched.
[NOT] RLIKE	Supported, with differences.	Same as [NOT] REGEXP.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3.3.3.4 Character Sets

GaussDB allows you to specify the following character sets for databases, schemas, tables, or columns.

Table 3-186 Character sets

MySQL	GaussDB
utf8mb4	Supported
gbk	Supported
gb18030	Supported

NOTE

Currently, GaussDB does not perform strict encoding logic verification on invalid characters that do not belong to the current character set. As a result, such invalid characters may be successfully entered. However, an error is reported during verification in MySQL.

3.3.3.5 Collation Rules

GaussDB allows you to specify the following collation rules for schemas, tables, or columns.

Table 3-187 Collation rules

MySQL	GaussDB
binary	Supported.
gb18030_bin	Supported.
gb18030_chinese_ci	Supported.
gbk_bin	Supported.
gbk_chinese_ci	Supported.
utf8_bin	Supported.
utf8_general_ci	Supported.
utf8mb4_0900_ai_ci	Supported.
utf8mb4_bin	Supported.
utf8mb4_general_ci	Supported.
utf8mb4_unicode_ci	Supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

Differences in collation rules:

- Currently, collation rules can only be specified for the character string type and some binary types. Other types do not support specifying collation rules. You can check the **typcollation** attribute of a type in the pg_type system catalog. If it is not **0**, the type supports the collation. In MySQL, collation can be specified for all types; however, specifying collation rules for types other than character strings and binary types has no practical meaning.
- The current collation rules can be specified only when the corresponding character set is the same as the database-level character set.
- The default collation of the utf8mb4 character set is utf8mb4_general_ci, which is the same as that in MySQL 5.7. utf8mb4_0900_ai_ci is not the default collation of utf8mb4.
- In GaussDB, UTF-8 and utf8mb4 are the same character set.

3.3.3.6 SQL

3.3.3.6.1 DDL

Table 3-188 DDL syntax compatibility

Description	Syntax	Difference
Create primary keys and UNIQUE indexes during table creation and modification.	ALTER TABLE and CREATE TABLE	• GaussDB does not support the UNIQUE INDEX|KEY index_name syntax. An error will be reported when the UNIQUE INDEX|KEY index_name syntax is used. However, MySQL supports these functions. • When a constraint is created as a global secondary index and USING BTREE is specified in the SQL statement, the underlying index is created as UB-tree. • When the table joined with the constraint is Ustore and USING BTREE is specified in the SQL statement, the underlying index is created as UB-tree.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Support prefix indexes.	CREATE INDEX	• The prefix length cannot exceed 2676. The actual length of the key value is restricted by the internal page. If a column contains multi-byte characters or an index has multiple keys, an error may be reported when the index line length exceeds the threshold. • In the CREATE INDEX syntax, the following keywords cannot be used as prefix keys for column names: COALESCE, EXTRACT, GREATEST, LEAST, LNNVL, NULLIF, NVL, NVL2, OVERLAY, POSITION, REGEXP_LIKE, SUBSTRING, TIMESTAMPDIFF, TREAT, TRIM, XMLCONCAT, XMLELEMENT, XMLEXISTS, XMLFOREST, XMLPARSE, XMLPI, XMLROOT, and XMLSERIALIZE. • Prefix keys are not supported in primary key and unique key indexes.
Specify character sets and collation rules.	ALTER SCHEMA, ALTER TABLE, CREATE SCHEMA, and CREATE TABLE	-
Create a partitioned table.	CREATE TABLE PARTITION	-
Specify table-level and column-level comments during table creation and modification.	CREATE TABLE and ALTER TABLE	-
Specify index-level comments during index creation.	CREATE INDEX	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Exchange the partition data of an ordinary table and a partitioned table.	ALTER TABLE PARTITION	Differences in ALTER TABLE EXCHANGE PARTITION: ● If MySQL tables or partitions use tablespaces, data in partitions and ordinary tables cannot be exchanged. If GaussDB tables or partitions use different tablespaces, data in partitions and ordinary tables can still be exchanged. ● MySQL does not verify the default values of columns. Therefore, data in partitions and ordinary tables can be exchanged even if the default values are different. GaussDB verifies the default values. If they are different, data in partitions and ordinary tables cannot be exchanged. ● After the DROP COLUMN operation is performed on a partitioned table or an ordinary table in MySQL, if the table definition is still consistent, data can be exchanged between partitions and ordinary tables. In GaussDB, data can be exchanged between partitions and ordinary tables only when the deleted columns of ordinary tables and partitioned tables are strictly aligned. ● MySQL and GaussDB use different hash algorithms. Therefore, data stored in the same hash partition may be inconsistent. As a result, the exchanged data may also be inconsistent. ● MySQL partitioned tables do not support foreign

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		keys. If an ordinary table contains foreign keys or other tables reference foreign keys of an ordinary table, data in partitions and ordinary tables cannot be exchanged. GaussDB partitioned tables support foreign keys. If the FOREIGN KEY constraints of two tables are the same, data in partitions and ordinary tables can be exchanged. If a GaussDB partitioned table does not contain foreign keys, an ordinary table is referenced by other tables, and the partitioned table is the same as the ordinary table, data in the partitioned table can be exchanged with that in the ordinary table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Support auto-increment columns.	ALTER TABLE and CREATE TABLE	• Currently, only local auto-increment columns of each DN are supported. • It is recommended that the auto-increment column be the first column of a non-global secondary index. Otherwise, an alarm is generated when a table is created, and errors may occur when some operations (such as ALTER TABLE EXCHANGE PARTITION) are performed on a table that contains auto-increment columns. The auto-increment column in MySQL must be the first column of the index. • In the syntax AUTO_INCREMENT = value, **value** must be a positive number less than 2^127. MySQL does not verify the value. • An error occurs if the auto-increment continues after an auto-increment value reaches the maximum value of a column data type. In MySQL, errors or warnings may be generated during auto-increment, and sometimes auto-increment continues until the maximum value is reached. • GaussDB does not support the *innodb_autoinc_lock_mode* system variable, but when its GUC parameter **auto_increment_cache** is set to **0**, the behavior of inserting auto-increment columns in batches is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		similar to that when the MySQL system variable *innodb_autoinc_lock_mode* is set to **1**. ● When 0s, NULLs, and definite values are imported or batch inserted into auto-increment columns, the auto-increment values inserted after an error occurs in GaussDB may not be the same as those in MySQL. The **auto_increment_cache** parameter is provided to control the number of reserved auto-increment values. ● In different execution plans, the auto-increment sequence and reserved auto-increment values may be different from those in MySQL. For example, "INSERT INTO table VALUES(...),(...),..." is distributed to different DNs. Therefore, in some execution plans, DNs cannot obtain the number of rows to be inserted. The **auto_increment_cache** parameter is provided to control the number of reserved auto-increment values. ● When auto-increment is triggered by parallel import or insertion of auto-increment columns, the cache value reserved for each parallel thread is used only in the thread. If the cache value is not used up, the values of auto-increment columns in the table are discontinuous. The auto-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		increment values generated by parallel insertion may not be the same as those generated in MySQL. • The SERIAL data type of GaussDB is an original auto-increment column, which is different from the **AUTO_INCREMENT** column. The SERIAL data type of MySQL is the **AUTO_INCREMENT** column. • The value of **auto_increment_offset** cannot be greater than that of **auto_increment_increment**. Otherwise, an error occurs. MySQL allows it and states that **auto_increment_offset** will be ignored. • If a table has a primary key or index, the sequence in which the **ALTER TABLE** command rewrites table data may be different from that in MySQL. GaussDB rewrites table data based on the table data storage sequence, while MySQL rewrites table data based on the primary key or index sequence. As a result, the auto-increment sequence may be different. • When the **ALTER TABLE** command is used to add or modify auto-increment columns, the number of auto-increment values reserved for the first time is the number of rows in the table statistics. The number of rows in the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		statistics may not be the same as that in MySQL. • When auto-increment is performed in a trigger or user-defined function, the return value of last_insert_id is updated. MySQL does not update it. • If the values of the GUC parameters **auto_increment_offset** and **auto_increment_increment** are out of range, an error occurs. MySQL automatically changes the value to a boundary value. • The last_insert_id function is not supported. • Currently, local temporary tables do not support auto-increment columns. • If **sql_mode** is set to **no_auto_value_on_zero**, the auto-increment columns of the table are not subject to NOT NULL constraints. In GaussDB and MySQL, when the value of an auto-increment column is not specified, **NULL** will be inserted into the auto-increment column, but auto-increment is triggered for the former and not triggered for the latter.
Delete the PRIMARY KEY constraints of a table.	ALTER TABLE	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
CREATE TABLE... LIKE syntax	CREATE TABLE ... LIKE	• In versions earlier than MySQL 8.0.16, CHECK constraints are parsed but their functions are ignored. In this case, CHECK constraints are not replicated. GaussDB supports replication of CHECK constraints. • For the set data type, MySQL supports replication while GaussDB does not during table creation. • When a table is created, all PRIMARY KEY constraint names in MySQL are fixed to **PRIMARY KEY**. GaussDB does not support replication of PRIMARY KEY constraint names. • When a table is created, MySQL supports replication of UNIQUE KEY constraint names, but GaussDB does not. • When a table is created, MySQL versions earlier than 8.0.16 do not have CHECK constraint information, but GaussDB supports replication of CHECK constraint names. • When a table is created, MySQL supports replication of index names, but GaussDB does not. • When a table is created across sql_mode, MySQL is controlled by the loose mode and strict mode. The strict mode may become invalid in GaussDB. For example, if the source table has the default value

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		"0000-00-00", GaussDB can create a table that contains the default value **"0000-00-00"** in "no_zero_date" strict mode, which means that the strict mode is invalid. MySQL fails to create the table because it is controlled by the strict mode. • MySQL supports cross-database table creation, but GaussDB does not. • If the source table is a temporary table, you can create a non-temporary table in MySQL but not in GaussDB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Compatible with syntax for changing table names.	ALTER TABLE[IF EXISTS] tbl_name RENAME [TO | AS | =] new_tbl_name; RENAME {TABLE | TABLES} tbl_name TO new_tbl_name [, tbl_name2 TO new_tbl_name2, ...];	• The ALTER RENAME syntax in GaussDB supports only the function of changing the table name and cannot be coupled with other function operations. • In GaussDB, only the old table name column supports the schema.table_name format, and the new and old table names belong to the same schema. • GaussDB does not support renaming of old and new tables across schemas. However, if you have the permission, you can modify the names of tables in other schemas in the current schema. • The syntax for renaming multiple groups of tables in GaussDB supports renaming of all local temporary tables, but does not support the combination of local temporary tables and non-local temporary tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
Create a partition.	ALTER TABLE [IF EXISTS] { table_name [*] | ONLY table_name | ONLY (table_name)} action [, ...]; action: move_clause | exchange_clause | row_clause | merge_clause | modify_clause | split_clause | add_clause | drop_clause | ilm_clause add_clause: ADD {{partition_less_than_item | partition_start_end_item | partition_list_item} | PARTITION({partition_less_than_item | partition_start_end_item | partition_list_item})}	• The ALTER TABLE table_name ADD PARTITION (partition_definition1, partition_definition1,…); syntax cannot be used to add multiple partitions. • Only the original syntax for adding multiple partitions is supported: ALTER TABLE table_name ADD PARTITION (partition_definition1), ADD PARTITION (partition_definition2[y1]), …;.

3.3.3.6.2 DML

Table 3-189 DML syntax compatibility

Description	Syntax	Difference
DELETE supports ORDER BY and LIMIT.	DELETE	-
UPDATE supports ORDER BY and LIMIT.	UPDATE	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

Support the REPLACE INTO syntax.	REPLACE	• Difference between the initial values of the time type. For example:
		– MySQL is not affected by the strict or loose mode. You can insert time 0 into a table.
		mysql> CREATE TABLE test(f1 TIMESTAMP NOT NULL, f2 DATETIME NOT NULL, f3 DATE NOT NULL);
		Query OK, 1 row affected (0.00 sec)
		
		mysql> REPLACE INTO test VALUES(f1, f2, f3);
		Query OK, 1 row affected (0.00 sec)
		
		mysql> SELECT * FROM test;
		+---------------------
		+---------------------+------------
		+
		| f1 | f2
		| f3 |
		+---------------------
		+---------------------+------------
		+
		| 0000-00-00 00:00:00 |
		0000-00-00 00:00:00 |
		0000-00-00 |
		+---------------------
		+---------------------+------------
		+
		1 row in set (0.00 sec)
		– The time 0 can be successfully inserted only when GaussDB is in loose mode.
		gaussdb=# SET b_format_version = '5.7';
		SET
		gaussdb=# SET b_format_dev_version = 's1';
		SET
		gaussdb=# SET sql_mode = '';
		SET
		gaussdb=# CREATE TABLE test(f1 TIMESTAMP NOT NULL, f2 DATETIME NOT NULL, f3 DATE NOT NULL) DISTRIBUTE BY HASH(f1);
		CREATE TABLE
		gaussdb=# REPLACE INTO test VALUES(f1, f2, f3);
		REPLACE 0 1
		gaussdb=# SELECT * FROM test;
		f1 | f2 | f3
		--------------------- 	
		+---------------------+------------

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		0000-00-00 00:00:00 |
		0000-00-00 00:00:00 |
		0000-00-00
		(1 row)
		In strict mode, the error is reported: date/time field value out of range: "0000-00-00 00:00:00".
		• Difference between the initial values of the BIT type when NOT NULL exists. For example:
		– The initial value of the BIT type is an empty string '' in MySQL, that is:
		mysql> CREATE TABLE test(f1 BIT(3) NOT NULL);
		Query OK, 0 rows affected (0.01 sec)
		
		mysql> REPLACE INTO test VALUES(f1);
		Query OK, 1 row affected (0.00 sec)
		
		mysql> SELECT f1, f1 IS NULL FROM test;
		+----+------------+
		| f1 | f1 is null |
		+----+------------+
		| | 0 |
		| | 0 |
		+----+------------+
		2 rows in set (0.00 sec)
		– If the initial value of the BIT type is **NULL** in GaussDB, an error is reported.
		gaussdb=# CREATE TABLE test(f1 int, f2 BIT(3) NOT NULL) DISTRIBUTE BY HASH(f1);
		CREATE TABLE
		gaussdb=# REPLACE INTO test VALUES(1, f2);
		ERROR: null value in column "f2" violates not-null constraint
		DETAIL: Failing row contains (1, null).

SELECT supports multi-partition query.	SELECT	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
UPDATE supports multi-partition update.	UPDATE	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

Import data by using LOAD DATA.	LOAD DATA	• The execution result of the LOAD DATA syntax is the same as that in MySQL strict mode. The loose mode is not adapted currently. For example, if the source data to be imported contains invalid characters, GaussDB reports an error, while MySQL reports an error in strict mode, or truncates the invalid characters and then import the data in loose mode.
		-- GaussDB
		gaussdb=# SET sql_mode='';
		SET
		gaussdb=# DROP TABLE IF EXISTS t1;
		DROP TABLE
		gaussdb=# CREATE TABLE t1(a text) CHARSET gbk;
		CREATE TABLE
		gaussdb=# SELECT unhex('1AFF1F') INTO OUTFILE '/tmp/3.data';
		COPY 1
		gaussdb=# LOAD DATA INFILE '/tmp/3.data' INTO TABLE t1 CHARACTER SET gbk;
		ERROR: invalid byte sequence for encoding "GBK": 0xff 0x1f
		CONTEXT: COPY t1, line 1.
		gaussdb=# DROP TABLE t1;
		DROP TABLE
		
		-- MySQL
		mysql> SET sql_mode='';
		Query OK, 0 rows affected (0.00 sec)
		mysql> DROP TABLE IF EXISTS t1;
		Query OK, 0 rows affected (0.02 sec)
		mysql> CREATE TABLE t1(a text) CHARSET gbk;
		Query OK, 0 rows affected (0.03 sec)
		mysql> SELECT unhex('1AFF1F') INTO OUTFILE '/tmp/3.data';
		Query OK, 1 row affected (0.00 sec)
		mysql> LOAD DATA INFILE '/tmp/3.data' INTO TABLE t1 CHARACTER SET gbk;
		Query OK, 1 row affected, 1 warning (0.01 sec)
		Records: 1 Deleted: 0 Skipped: 0 Warnings: 1
		mysql> SHOW warnings;
		+---------+------

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		+----------+------+ Message | +----------+------ +----------+ | Warning | 1366 | Incorrect string value: '\xFF\x1F' for column 'a' at row 1 | +----------+------ +----------+ 1 row in set (0.00 sec) mysql> SELECT hex(a) FROM t1; +--------+ | hex(a) | +--------+ | 1A | +--------+ 1 row in set (0.00 sec) mysql> DROP TABLE t1; Query OK, 0 rows affected (0.02 sec) • The **IGNORE** and **LOCAL** parameters are used only to ignore the conflicting rows when the imported data conflicts with the data in the table and to automatically fill default values for other columns when the number of columns in the file is less than that in the table. Other functions are not supported currently. • If the keyword LOCAL is specified and the file path is a relative path, the file is searched from the binary directory. If the keyword LOCAL is not specified and the file path is a relative path, the file is searched from the data directory. • If single quotation marks are specified as separators, escape characters, and newline characters in the syntax, lexical parsing errors occur.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		• The **[(col_name_or_user_var [, col_name_or_user_var]...)]** parameter cannot be used to specify a column repeatedly. • The newline character specified by **[FIELDS TERMINATED BY 'string']** cannot be the same as the separator specified by **[LINES TERMINATED BY'string']**. • If the data written to a table by running **LOAD DATA** cannot be converted to the data type of the table, an error is reported. • Columns can only be specified by column name instead of user variables. • The LOAD DATA SET expression does not support the calculation of a specified column name. • If there is no implicit conversion function between the return value type of the SET expression and the corresponding column type, an error is reported. • LOAD DATA does not support the INSERT or DELETE trigger. • LOAD DATA applies only to tables but not views. • The default newline character of the file in Windows is different from that in Linux. LOAD DATA cannot identify this scenario and reports an error. You are advised to check the newline character at the end of

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		lines in the file to be imported.
		● If the number of columns in the source data is greater than that in the target table, GaussDB displays an alarm, while MySQL displays an alarm in the following scenarios and reports an error in other scenarios:
		– The strict mode is not enabled for **sql_mode**.
		– The **IGNORE** option is specified.
		– The **LOCAL** option is specified and the **REPLACE** option is not specified.
		● When the source data file ends with a column delimiter, MySQL processes the file as if there is no new column after the delimiter, while GaussDB processes the file as if there is an empty string column after the delimiter.
		-- Source data in **1.csv**
		1,2,
		
		-- GaussDB
		gaussdb=# DROP TABLE IF EXISTS t1;
		NOTICE: table "t1" does not exist, skipping
		DROP TABLE
		gaussdb=# CREATE TABLE t1(a char(30), b char(30), c char(30));
		CREATE TABLE
		gaussdb=# LOAD DATA infile '/tmp/1.csv' INTO TABLE t1 fields terminated by ',';
		COPY 1
		gaussdb=# SELECT c IS NULL, length(c) FROM t1;
		?column? | length
		----------+--------
		f | 0
		(1 row)
		
		-- MySQL
		mysql> DROP TABLE IF EXISTS t1;
		Query OK, 0 rows affected (0.06

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		sec) mysql> CREATE TABLE t1(a char(30), b char(30), c char(30)); Query OK, 0 rows affected (0.04 sec) mysql> LOAD DATA infile '/tmp/1.csv' INTO TABLE t1 fields terminated by ','; ERROR 1261 (01000): Row 1 doesn't contain data for all columns

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

Compatible with INSERT IGNORE.	INSERT IGNORE	• GaussDB displays the error information after the downgrade. MySQL records the error information after the downgrade to the error stack and runs the **show warnings;** command to view the error information. For example:
		• Time type difference. For example:
		– The default values of **date**, **datetime**, and **timestamp** in GaussDB are **0**.
		gaussdb=# CREATE TABLE test(f1 DATE NOT NULL, f2 DATETIME NOT NULL, f3 TIMESTAMP NOT NULL);
		CREATE TABLE
		gaussdb=# INSERT IGNORE INTO test VALUES(NULL, NULL, NULL);
		WARNING: null value in column "f1" violates not-null constraint
		DETAIL: Failing row contains (null, null, null, null).
		WARNING: null value in column "f2" violates not-null constraint
		DETAIL: Failing row contains (null, null, null, null).
		WARNING: null value in column "f3" violates not-null constraint
		DETAIL: Failing row contains (null, null, null, null).
		INSERT 0 1
		gaussdb=#
		SELECT * FROM test;
		f1 | f2 | f3
		------------+---------------------+---------------------
		1970-01-01 | 1970-01-01 00:00:00 | 1970-01-01 00:00:00
		(1 row)
		– The default values of **date**, **datetime**, and **timestamp** in MySQL are **0**.
		mysql> CREATE TABLE test(f1 DATE NOT NULL, f2 DATETIME NOT NULL, f3 TIMESTAMP NOT NULL);

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		Query OK, 0 rows affected (0.00 sec)
		
		mysql> INSERT IGNORE INTO test VALUES(NULL, NULL, NULL);
		Query OK, 1 row affected, 3 warnings (0.00 sec)
		
		mysql> show warnings;
		+---------+------ +----------------------------+
		| Level | Code | Message |
		+---------+------ +----------------------------+
		| Warning | 1048 | Column 'f1' cannot be null |
		| Warning | 1048 | Column 'f2' cannot be null |
		| Warning | 1048 | Column 'f3' cannot be null |
		+---------+------ +----------------------------+
		3 rows in set (0.00 sec)
		
		mysql> SELECT * FROM test;
		+------------+---------- +---------------------+
		| f1 | f2 | f3 |
		+------------+---------- +---------------------+
		| 0000-00-00 | 0000-00-00 00:00:00 | 0000-00-00 00:00:00 |
		+------------+---------- +---------------------+
		1 row in set (0.00 sec)
		• GaussDB does not support the MySQL bit type. Therefore, the INSERT IGNORE error downgrade is not supported when the NOT NULL constraint of the bit type is ignored and the length of the inserted bit type is different from that defined.
		– Bit type in GaussDB
		gaussdb=# CREATE TABLE test(f1 BIT(10) NOT NULL);
		CREATE TABLE
		gaussdb=# INSERT IGNORE INTO test VALUES(NULL);
		ERROR: Un-support feature
		DETAIL: ignore null for insert statement is not supported in column f1.
		gaussdb=# INSERT IGNORE INTO test VALUES('1010');

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		ERROR: bit string length 4 does not match type bit(10)
		CONTEXT: referenced column: f1
		– Bit type in MySQL
		mysql> CREATE TABLE test(f1 BIT(10) NOT NULL);
		Query OK, 0 rows affected (0.00 sec)
		
		mysql> INSERT IGNORE INTO test VALUES(NULL);
		Query OK, 1 row affected, 1 warning (0.00 sec)
		
		mysql> INSERT IGNORE INTO test VALUES('1010');
		Query OK, 1 row affected, 1 warning (0.01 sec)
		• If the precision is specified for the time type in MySQL, the precision is displayed when the zero value is inserted. It is not displayed in GaussDB. For example:
		– Time precision specified in GaussDB
		gaussdb=# CREATE TABLE test(f1 TIME(3) NOT NULL, f2 DATETIME(3) NOT NULL, f3 TIMESTAMP(3) NOT NULL);
		CREATE TABLE
		gaussdb=# INSERT IGNORE INTO test VALUES(NULL,NULL,NULL);
		WARNING: null value in column "f1" violates not-null constraint
		DETAIL: Failing row contains (null, null, null).
		WARNING: null value in column "f2" violates not-null constraint
		DETAIL: Failing row contains (null, null, null).
		WARNING: null value in column "f3" violates not-null constraint
		DETAIL: Failing row contains (null, null, null).
		INSERT 0 1
		gaussdb=# SELECT * FROM test;
		f1 | f2 | f3
		----------+----------+----------
		00:00:00 | 1970-01-01 00:00:00 | 1970-01-01 00:00:00
		(1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		– Time precision specified in MySQL
		mysql> CREATE TABLE test(f1 TIME(3) NOT NULL, f2 DATETIME(3) NOT NULL, f3 TIMESTAMP(3) NOT NULL);
		Query OK, 0 rows affected (0.00 sec)
		
		mysql> INSERT IGNORE INTO test VALUES(NULL,NULL,NULL);
		Query OK, 1 row affected, 3 warnings (0.00 sec)
		
		mysql> SELECT * FROM test;
		+--------------
		+-------------------------
		+-------------------------+
		| f1 | f2 | f3 |
		+--------------
		+-------------------------
		+-------------------------+
		| 00:00:00.000 | 0000-00-00 00:00:00.000 | 0000-00-00 00:00:00.000 |
		+--------------
		+-------------------------
		+-------------------------+
		1 row in set (0.00 sec)
		• The execution process in MySQL is different from that in GaussDB. Therefore, the number of generated warnings may be different. For example:
		– Number of warnings generated in GaussDB
		gaussdb=# CREATE TABLE test(f1 INT, f2 INT not null);
		CREATE TABLE
		gaussdb=# INSERT INTO test VALUES(1,0),(3,0),(5,0);
		INSERT 0 3
		gaussdb=# INSERT IGNORE INTO test SELECT f1+1, f1/f2 FROM test;
		WARNING: division by zero
		CONTEXT: referenced column: f2
		WARNING: null value in column "f2" violates not-null constraint
		DETAIL: Failing row contains (2, null).
		WARNING: division by zero
		CONTEXT: referenced column: f2
		WARNING: null value in column "f2" violates not-null

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		constraint
		DETAIL: Failing row contains (4, null).
		WARNING: division by zero
		CONTEXT: referenced column: f2
		WARNING: null value in column "f2" violates not-null constraint
		DETAIL: Failing row contains (6, null).
		INSERT 0 3
		– Number of warnings generated in MySQL
		mysql> CREATE TABLE test(f1 INT, f2 INT not null);
		Query OK, 0 rows affected (0.01 sec)
		
		mysql> INSERT INTO test VALUES(1,0),(3,0),(5,0);
		Query OK, 3 rows affected (0.00 sec)
		Records: 3 Duplicates: 0 Warnings: 0
		
		mysql> INSERT IGNORE INTO test SELECT f1+1, f1/f2 FROM test;
		Query OK, 3 rows affected, 4 warnings (0.00 sec)
		Records: 3 Duplicates: 0 Warnings: 4
		• The differences between MySQL's and GaussDB's INSERT IGNORE in triggers are as follows:
		– INSERT IGNORE used in a GaussDB trigger
		gaussdb=# CREATE TABLE test1(f1 INT NOT NULL);
		CREATE TABLE
		gaussdb=# CREATE TABLE test2(f1 INT);
		CREATE TABLE
		gaussdb=# CREATE OR REPLACE FUNCTION trig_test() RETURNS TRIGGER AS $$
		gaussdb$# BEGIN
		gaussdb$# INSERT IGNORE INTO test1 VALUES(NULL);
		gaussdb$# RETURN NEW;
		gaussdb$# END;
		gaussdb$# $$ LANGUAGE plpgsql;
		CREATE FUNCTION
		gaussdb=# CREATE TRIGGER trig2 BEFORE INSERT ON test2 FOR EACH ROW EXECUTE PROCEDURE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		trig_test();
		CREATE TRIGGER
		gaussdb=# INSERT INTO test2 VALUES(NULL);
		WARNING: null value in column "f1" violates not-null constraint
		DETAIL: Failing row contains (null).
		CONTEXT: SQL statement "INSERT IGNORE INTO test1 VALUES(NULL)"
		PL/pgSQL function trig_test() line 3 at SQL statement
		INSERT 0 1
		gaussdb=# SELECT * FROM test1;
		 f1
		---- 	
		  0
		(1 rows)
		
		gaussdb=# SELECT * FROM test2;
		 f1
		---- 	
		
		(1 rows)
		– INSERT IGNORE used in a MySQL trigger
		mysql> CREATE TABLE test1(f1 INT NOT NULL);
		Query OK, 0 rows affected (0.01 sec)
		
		mysql> CREATE TABLE test2(f1 INT);
		Query OK, 0 rows affected (0.00 sec)
		
		mysql> DELIMITER ||
		mysql> CREATE TRIGGER trig2 BEFORE INSERT ON test2 FOR EACH ROW
		    -> BEGIN
		    -> INSERT IGNORE into test1 values(NULL);
		    -> END||
		Query OK, 0 rows affected (0.01 sec)
		
		mysql> DELIMITER ;
		mysql> INSERT INTO test2 VALUES(NULL);
		ERROR 1048 (23000): Column 'f1' cannot be null
		mysql> INSERT IGNORE INTO test2 VALUES(NULL);
		Query OK, 1 row affected (0.00 sec)
		
		mysql> SELECT * FROM test1;
		+----+
		| f1 |

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference

		+----+
		| 0 |
		+----+
		1 row in set (0.00 sec)
		
		mysql> SELECT * FROM test2;
		+------+
		| f1 |
		+------+
		| NULL |
		+------+
		1 row in set (0.00 sec)
		• The implementation mechanism of Boolean and serial in GaussDB is different from that in MySQL. Therefore, the default zero value in GaussDB is different from that in MySQL. For example:
		– Behavior in GaussDB
		gaussdb=# CREATE TABLE test(f1 SERIAL, f2 BOOL NOT NULL);
		NOTICE: CREATE TABLE will create implicit sequence "test_f1_seq" for serial column "test.f1"
		CREATE TABLE
		gaussdb=# INSERT IGNORE INTO test values(NULL,NULL);
		WARNING: null value in column "f1" violates not-null constraint
		DETAIL: Failing row contains (null, null).
		WARNING: null value in column "f2" violates not-null constraint
		DETAIL: Failing row contains (null, null).
		INSERT 0 1
		gaussdb=# SELECT * FROM test;
		f1 | f2
		----+----
		0 | f
		(1 row)
		– Behavior in MySQL
		mysql> CREATE TABLE test(f1 SERIAL, f2 BOOL NOT NULL);
		Query OK, 0 rows affected (0.00 sec)
		
		mysql> INSERT IGNORE INTO test values(NULL,NULL);
		Query OK, 1 row affected, 1 warning (0.00 sec)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Syntax	Difference
		mysql> SELECT * FROM test; +----+----+ | f1 | f2 | +----+----+ | 1 | 0 | +----+----+ 1 row in set (0.00 sec)

3.3.3.6.3 DCL

Table 3-190 DCL syntax compatibility

Description	Syntax	Difference
Set names with COLLATE specified.	SET [SESSION | LOCAL] NAMES {'charset_name' [COLLATE 'collation_name'] | DEFAULT};	GaussDB does not allow **charset_name** to be different from the database character set. For details, see **SET**.

3.3.3.7 Drivers

3.3.3.7.1 JDBC API Reference

The JDBC API definitions in GaussDB are the same as those in MySQL and comply with industry standards. This section describes the behavior differences of JDBC APIs between GaussDB in MySQL-compatible mode and MySQL.

Obtaining Data from a Result Set

ResultSet objects provide a variety of methods to obtain data from a result set. **Table 3-191** describes the common methods for obtaining data. If you want to know more about other methods, see JDK official documents.

Table 3-191 Common methods for obtaining data from a result set

Method	Description	Difference
int getInt(int columnIndex)	Obtains int data by column index.	-
int getInt(String columnLabel)	Obtains int data by column name.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Method	Description	Difference
String getString(int columnIndex)	Obtains string data by column index.	If the column type is integer and the column contains the **ZEROFILL** attribute, GaussDB pads 0s to meet the width required by the **ZEROFILL** attribute and outputs the result. MySQL directly outputs the result.
String getString(String columnLabel)	Obtains string data by column name.	If the column type is integer and the column contains the **ZEROFILL** attribute, GaussDB pads 0s to meet the width required by the **ZEROFILL** attribute and outputs the result. MySQL directly outputs the result.
Date getDate(int columnIndex)	Obtains date data by column index.	-
Date getDate(String columnLabel)	Obtains date data by column name.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

4 Tool Reference

4.1 Downloading and Installing Tools

Downloading and Installing Tools

1. Download the driver package and its verification package listed in **Table 4-1**.

   Table 4-1 Driver package download list

Version	Download Address
V2.0-10.x	**Driver package** **Verification package for the driver package**

2. Verify the driver package.

   To prevent a software package from being tampered with during transmission or storage, download the corresponding verification package and perform the following steps to verify the software package:

   - Verifying the software package integrity on a Linux server:
     i. Upload the software package and verification package to the same directory on the VM.
     ii. Run the following command to verify the integrity of the software package:

     ```
     F=$(echo DBS-GaussDB-driver*.tar.gz);echo "$(awk '{print $1}' "$F.sha256")  $F" | sha256sum -c -
     ```

     If **OK** is displayed in the command output, the verification is successful. For example:

     ```
     DBS-GaussDB-driver_507.0_20260629130806.133858239.tar.gz: OK
     ```

   - Verifying the software package integrity on a Windows server:
     i. Press **Win+R** to open the **Run** dialog box. Type **cmd** in the text box and press **Enter** to open the **Command Prompt** window.
     ii. Run the following command to obtain the hash value of the driver package:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

certutil -hashfile *{local_directory_of_the_driver_package}*\ *{driver_package_name}* **sha256**

- Replace *{local_directory_of_the_driver_package}* with the actual download path, for example, **C:\Users**.
- Replace *{driver_package_name}* with the name of the downloaded driver package, for example, **DBS-GaussDB-driver_507.0_20260629130806.133858239.tar.gz**.

Example: **certutil -hashfile** *C:|Users|DBS-GaussDB-driver_507.0_20260629130806.133858239.tar.gz* **sha256**

iii. Compare the hash value obtained in **2.ii** with the hash value of the verification package obtained in **Table 4-1**.

- If they are consistent, the verification is successful.
- If they are inconsistent, download the driver package again and repeat **2.i** to **2.iii** to verify the driver package.

3. Log in to the server and create a directory for storing the tool package.
```
mkdir -p /opt/bin
```
4. Upload the driver package obtained in **1** to the created directory.
5. Run the following commands to decompress the driver package. Replace the driver package name with the actual one. **DBS-GaussDB-driver_507.0_20260629130806.133858239.tar.gz** is used as an example.
```
cd /opt/bin
tar -zxvf DBS-GaussDB-driver_507.0_20260629130806.133858239.tar.gz
```
6. Run the following commands to obtain the gsql client tool package:
```
cd /opt/bin/DBS-GaussDB-driver_507.0_20260629130806.133858239/Distributed/
tar -zxvf gsql_driver.tar.gz
cd /opt/bin/DBS-GaussDB-driver_507.0_20260629130806.133858239/Distributed/gsql_driver/Euler2.5_X86_64
tar -zxvf GaussDB-Kernel_507.0.0.B071_Gsql_X86_Distributed.tar.gz
```

NOTE

- The relative path of the tool package varies depending on where you decompressed it. In this section, the tool package of Euler2.5_x86_64 is used as an example.
- Replace the client tool package name with the actual one. **GaussDB-Kernel_507.0.0.B071_Gsql_x86_Distributed.tar.gz** is used as an example.

7. Copy the decompressed **GaussDB-Kernel_507.0.0_Euler_64bit_Gsql.tar.gz** client tool package to the **/opt/bin** directory.
```
cp GaussDB-Kernel_507.0.0_Euler_64bit_Gsql.tar.gz /opt/bin
```

NOTE

The gsql tool package contains the gsql client, gs_dump, gs_dumpall, and gs_restore.

8. Run the following commands to decompress the package:
```
cd /opt/bin
tar -zxvf GaussDB-Kernel_507.0.0_Euler_64bit_Gsql.tar.gz
```
9. Use the script to automatically configure the environment variables.
```
source gsql_env.sh
```

4.2 Database Connection Tools

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

4.2.1 gsql for Connecting to a Database

4.2.1.1 gsql Usage Guide

Description

gsql, provided by GaussDB, is a database connection tool running in the command-line interface. You can use gsql to connect to the server and perform operations and maintenance on it. gsql enables users to perform basic operations on the database and provides several advanced features.

Basic functions:

- Connect to a database: By default, only the localhost can be connected. To connect to a remote database, you must configure it on the localhost. For details, see "Database Quick Start > Connecting to a Database > Using gsql to Connect to a Database" in *Developer Guide*.

  NOTE

  If gsql is used to connect to a database, the connection timeout period will be 5 minutes. If the database has not correctly set up a connection and authenticated the identity of the client within this period, gsql will time out and exit. To resolve this problem, see **FAQ**.

- Run SQL statements: You can interactively enter SQL statements and run them, or run specified SQL statements in a file.
- Run meta-commands: Meta-commands can help administrators view database object information, query cache information, format SQL output, and connect to a new database. For details about meta-commands, see **Meta-Command Reference**.

Advanced functions:

Table 4-2 lists the advanced features of gsql.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-2 Advanced gsql features

Feature	Description
Variable	gsql provides a variable feature that is similar to the **shell** command of Linux. The following **\set** meta-command of gsql can be used to set a variable: \set *varname value* To delete the variables set by the **\set** command, run the following command: \unset *varname* **NOTE** • A variable is a simple name-value pair. The value can be any characters in any length. • Variable names must consist of case-sensitive letters (including non-Latin letters), digits, and underscores (_). • If the **\set** *varname* meta-command (without the second parameter) is used, the variable is set without a value specified. • If the **\set** meta-command without parameters is used, values of all variables are displayed. For details about variable examples and descriptions, see **Variables**.
SQL substitution	Common SQL statements can be set to variables using the variable feature of gsql to simplify operations. For details about examples and descriptions about SQL substitution, see **SQL substitution**.
Custom prompt	Prompts of gsql can be customized. Prompts can be modified by changing the reserved three variables of gsql: *PROMPT1*, *PROMPT2*, and *PROMPT3*. These variables can be user-defined or set to the values predefined by gsql. For details, see **Prompt**.
Historical client operation records	gsql can record historical client operations. This function is enabled by specifying the **-r** parameter when a client is connected. The number of historical records can be set using the **\set** command. For example, **\set HISTSIZE 50** indicates that the number of historical records is set to **50**. **\set HISTSIZE 0** indicates that the operation history is not recorded. **NOTE** • The default number of historical records is 32. The maximum number of historical records is 500. If interactively entered commands contain Chinese characters, only the UTF-8 encoding environment is supported. • For security purposes, SQL statements containing case-insensitive character strings such as PASSWORD, IDENTIFIED, GS_ENCRYPT_AES128, GS_DECRYPT_AES128, GS_ENCRYPT, GS_DECRYPT, GS_ENCRYPT_BYTEA, GS_DECRYPT_BYTEA, PG_CREATE_PHYSICAL_REPLICATION_SLOT_EXTERN, SECRET_ACCESS_KEY, SECRETKEY, CREATE_CREDENTIAL, ACCESSKEY, SECRET_KEY, DESENCRYPT, DESDECRYPT, DES3ENCRYPT and DES3DECRYPT are identified as statements containing sensitive information and are not recorded in historical information. That is, the statements cannot be displayed by scrolling up or down.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Variables

  To set a variable, run the **\set** meta-command of gsql. For example, to set variable *foo* to **bar**, run the following command:

  ```
  gaussdb=#  \set foo bar
  ```

  To reference the value of a variable, add a colon (:) before the variable. For example, to view the value of variable *foo*, run the following command:

  ```
  gaussdb=#  \echo :foo
  bar
  ```

  This variable reference method is applicable to regular SQL statements and meta-commands except **\copy**, **\ef**, **\help**, **\sf**, and **\!**.

  gsql pre-defines some special variables and plans the values of these variables. To ensure compatibility with later versions, do not use these variables for other purposes. For details about special variables, see **Table 4-3**.

  NOTE

  - All the special variables consist of upper-case letters, digits, and underscores (_).
  - To view the default value of a special variable, run the **\echo :***varname* meta-command, for example, **\echo :***DBNAME*.

  Table 4-3 Settings of special variables

Variable	Setting Method	Description
DBNAME	\set DBNAME *dbname*	Name of the connected database. This variable is set again when a database is connected.
ECHO	\set ECHO all | queries	• If this variable is set to **all**, only the query information is displayed. This has the same effect as specifying the **-a** parameter when gsql is used to connect to a database. • If this variable is set to **queries**, the command line and query information are displayed. This has the same effect as specifying the **-e** parameter when gsql is used to connect to a database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Variable	Setting Method	Description
ECHO_HIDDEN	\set ECHO_HIDDEN on | off | noexec	When a meta-command (such as **\dg**) is used to query database information, the value of this variable determines the query behavior. ● If this variable is set to **on**, the query statements that are called by the meta-command are displayed, and then the query result is displayed. This has the same effect as specifying the **-E** parameter when gsql is used to connect to a database. ● If this variable is set to **off**, only the query result is displayed. ● If this variable is set to **noexec**, only the query information is displayed, and the query is not run.
ENCODING	\set ENCODING *encoding*	Character set encoding of the current client.
FETCH_COUNT	\set FETCH_COUNT *variable*	● If the value is an integer greater than 0, for example, *n*, *n* lines will be selected from the result set to the cache and displayed on the screen when the SELECT statement is run. ● If this variable is not set or set to a value less than or equal to 0, all results are selected at a time to the cache when the SELECT statement is run. **NOTE** A proper variable value helps reduce the memory usage. The recommended value range is from 100 to 1000.
HISTCONTROL	\set HISTCONTROL ignorespace | ignoredups | ignoreboth | none	● **ignorespace**: A line started with a space is not written to the historical record. ● **ignoredups**: A line that exists in the historical record is not written to the historical record. ● **ignoreboth**, **none**, or other values: All the lines read in interaction mode are saved in the historical record. **NOTE** **none** indicates that **HISTCONTROL** is not set.
HISTFILE	\set HISTFILE *filename*	Specifies the file for storing historical commands. The default value is **~/.bash_history**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Variable	Setting Method	Description
HISTSIZE	\set HISTSIZE *size*	Specifies the number of commands to store in the command history. The default value is **500**.
HOST	\set HOST *hostname*	Specifies the name of a connected host.
IGNOREEOF	\set IGNOREEOF *variable*	• If this variable is set to a number, for example, **10**, the first nine EOF characters (generally **Ctrl**+**C**) entered in gsql are neglected and the gsql program exits when the tenth **Ctrl**+**C** is entered. • If this variable is set to a non-numeric value, the default value is **10**. • If this variable is deleted, gsql exits when an EOF is entered.
LASTOID	\set LASTOID *oid*	Specifies the last OID, which is the value returned by an **INSERT** or **lo_import** command. This variable is valid only before the output of the next SQL statement is displayed.
ON_ERROR_ROLLBACK	\set ON_ERROR_ROLLBACK on | interactive | off	• **on**: When an error occurs in a statement in a transaction block, the system automatically rolls back to the state before the previous command is executed and continues to execute subsequent commands. This mode works by implicitly creating a SAVEPOINT before each command of a transaction block. When an error occurs, the system automatically rolls back to the previously created SAVEPOINT to cancel the impact of the failed command. • **interactive**: Automatic rollback is enabled only in interactive sessions. • **off** (default value): The entire transaction is terminated when an error occurs in a statement in a transaction block.
ON_ERROR_STOP	\set ON_ERROR_STOP on | off	• **on**: The execution stops if an error occurs. In interactive mode, gsql returns the output of executed commands immediately. • **off** (default value): An error, if occurring during the execution, is ignored, and the execution continues.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Variable	Setting Method	Description
PORT	\set PORT *port*	Specifies the port number of a connected database.
USER	\set USER *username*	Specifies the database user you are currently connected as.
VERBOSITY	\set VERBOSITY terse | default | verbose	Controls the detail level of the error report. • **terse**: Only critical and major error texts and text locations are returned (which is generally suitable for single-line error information). • **default**: Critical and major error texts and text locations, error details, and error messages (possibly involving multiple lines) are all returned. • **verbose**: All error information is returned.

- SQL substitution

  gsql, like a parameter of a meta-command, provides a key feature that enables you to substitute a standard SQL statement for a gsql variable. gsql also provides a new alias or identifier for the variable. To replace the value of a variable using the SQL substitution method, add a colon (:) before the variable. For example:

```
gaussdb=#  \set foo 'HR.areaS'
-- Run the following command to query the HR.areaS table:
gaussdb=#  SELECT * FROM :foo;
 area_id |       area_name
---------+------------------------
       4 | Middle East and Africa
       3 | Asia
       1 | Europe
       2 | Americas
(4 rows)
```

  NOTE

  The value of a variable is copied word by word and can even contain asymmetric quotation marks or backslash commands. Therefore, the input content must be meaningful.

- Prompts

  The gsql prompt can be set using the three variables in **Table 4-4**. These variables consist of characters and special escape characters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-4 Prompt variables

Variable	Description	Example

PROMPT1	Specifies the normal prompt used when gsql requests a new command.
	The default value of *PROMPT1* is **%o%R%#**.
		PROMPT1 can be used to change the prompt.
		• Change the prompt to **local**:
		`gaussdb=# \set PROMPT1 %M`
		`local:/data/huawei/wisequery/perfadm_mppdb`
		• Change the prompt to **name**:
		`gaussdb=# \set PROMPT1 name`
		`name`
		• Change the prompt to **=**:
		`gaussdb=# \set PROMPT1 %R`
		`=`

PROMPT2	Specifies the prompt displayed when more input is expected (when the command is not terminated with a semicolon (;) or the quotation marks ("") are not in pair).	*PROMPT2* can be used to display the prompt. `gaussdb=#  \set PROMPT2 TEST` `gaussdb=#  SELECT * FROM HR.areaS` `TEST;` ` area_id	area_name` `---------+--------------------` `1	Europe` `2	Americas` `4	Middle East and Africa` `3	Asia` `(4 rows)`

PROMPT3	Specifies the prompt displayed when the **COPY** statement (such as **COPY FROM STDIN**) is run and data input is expected.	*PROMPT3* can be used to display the COPY prompt.
		`gaussdb=#  \set PROMPT3 '>>>>'`
		`gaussdb=#  COPY HR.areaS FROM STDIN;`
		`Enter data to be copied followed by a newline.`
		`End with a backslash and a period on a line by itself.`
		`>>>>1 aa`
		`>>>>2 bb`
		`>>>>\.`

The value of the selected prompt variable is printed literally. However, a value containing a percent sign (%) is replaced by the predefined contents depending on the character following the percent sign (%). For details about the defined substitutions, see **Table 4-5**.

Table 4-5 Defined substitutions

Symbol	Description
%M	Replaced by the full host name (with domain name). The full name is **[local]** if the connection is over a UDS, or **[local:/dir/name]** if the UDS is not at the compiled default location.
%m	Replaced by the host name truncated at the first dot. It is **[local]** if the connection is over a UDS.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Symbol	Description
%o	Database name. For the **postgres** database, **gaussdb** is displayed. For other databases, the database name is displayed.
%>	Replaced by the number of the port that the host is listening on.
%n	Replaced by the database session username.
%/	Replaced by the name of the current database.
%~	Similar to %/. However, the tilde (~) is used if the database is your default database.
%#	The number sign (#) is used if the session user is the database administrator; otherwise, the greater-than sign (>) is used.
%R	• For *PROMPT1*, **=** typically, but **^** in single-line mode, or **!** if the session is disconnected from the database (which can happen if **\connect** fails). • For *PROMPT2*, it is replaced by a hyphen (-), an asterisk (*), a single or double quotation mark, or a dollar sign ($), depending on whether gsql expects more input, for example, the query is not terminated, in a /* ... */ the comment, enclosed with quotation marks, or in a dollar sign extension.
%x	Replaced by the transaction status. • An empty string when it is not in a transaction block • An asterisk (*) when it is in a transaction block • An exclamation mark (!) when it is in a failed transaction block • A question mark (?) when the transaction status is indefinite (for example, because there is no connection).
%digits	Replaced by the character with the specified byte.
%:name	Replaced by the value of the *name* variable of gsql.
%`command`	Replaced by the output of the command specified by **command**, similar to the backquote substitution.
%[. . . %]	Prompts may contain terminal control characters which, for example, change the color, background, or style of the prompt text, or change the title of the terminal window. For example: `gaussdb=# \set PROMPT1 '%[%033[1;33;40m%]%n@%/%R%[%033[0m%]%#'` The output is a boldfaced (1;) yellow-on-black (33;40) prompt on VT100-compatible, color-capable terminals.

Environment variables:

For details about gsql-related environment variables, see **Table 4-6**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-6 Environment variables related to gsql

Name	Description
COLUMNS	If **\set columns** is set to **0**, this parameter controls the width of the wrapped format. This width determines whether to change the wide output mode into the vertical output mode if automatic expansion is enabled.
PAGER	If the query results do not fit on the screen, they are redirected through this command. You can use the **\pset** command to disable the pager. Typically, the **more** or **less** command is used for viewing the query result page by page. The default is platform-dependent. **NOTE** Display of the **less** command is affected by the *LC_CTYPE* environment variable.
PSQL_EDITOR	The **\e** and **\ef** commands use the editor specified by the environment variables. The variables are examined in the order listed. The default editor on Unix is vi.
EDITOR	
VISUAL	

PSQL_EDITOR_LINENUMBER_ARG	When the **\e** or **\ef** command is used with a line number parameter, this variable specifies the command-line parameter used to pass the starting line number to the editor. For the Emacs or Vi editor, the value of this parameter is a plus sign (+). Include a space in the value of the variable if space is needed between the option name and the line number. For example:
	`PSQL_EDITOR_LINENUMBER_ARG = '+'`
	`PSQL_EDITOR_LINENUMBER_ARG='--line '`
	A plus sign (+) is used by default on Unix.

PSQLRC	Specifies the location of the user's .gsqlrc file.
SHELL	Using the **\!** command has the same effect as executing the shell command.
TMPDIR	Specifies the directory for storing temporary files. The default value is **/tmp**.

NOTE

gsql does not support displaying dates of the datea type in the format specified by **nls_date_format**. Currently, only the YYYY-MM-DD HH24-MI-SS AD format is supported.

Prerequisites

- The user using gsql must have the permission to access the database.
- gsql must match the database version.
- If the gsql client-side encoding is set to UTF-8 and the data encoding format returned by GaussDB is not UTF-8, the data cannot be correctly displayed on the gsql client.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- You can use the **gsql_env.sh** script in the release package to set environment variables, for example, **source gsql_env.sh**.
- If environment variables have been set, the environment variables in the script take precedence after the **gsql_env.sh** script is executed.

Syntax

```
gsql [OPTION]... [DBNAME [USERNAME]]
```

Parameters

For details about gsql parameters, see **Table 4-7**, **Table 4-8**, **Table 4-9**, **Table 4-10**, and **Table 4-11**.

Table 4-7 Common parameters

Parameter	Description	Value Range
-c, --command=COMMAND	Specifies that gsql is to run a string command and then exit.	-
-d, --dbname=DBNAME	Specifies the name of the database to be connected. If the database name is not specified, the default database name generated during initialization is used. In addition, gsql allows you to use extended DBNAMEs, that is, connection strings in the format of "postgres[ql]://[user[:password]@][netloc][:port][, ...][/dbname][?param1=value1&...]" or "[key=value] [...]" as DBNAMEs. gsql parses connection information from the connection strings and preferentially uses the information. **NOTE** When gsql uses the extended DBNAMEs to create a connection, the **replication** parameter cannot be specified.	A string.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range
-f, --file=FILENAME	Specifies that files are used as the source of commands instead of reading commands entered interactively. After the files are processed, gsql exits. If *FILENAME* is set to a hyphen (-), then standard input is read. **NOTE** If the server-side encoding format is GB18030 or GB18030_2022 and **gsql -f** is executed together with the pipe character ('>'), for example, **gsql -f xxx.sql > xxx.log**, the encoding format of the imported file must be the same as that of the server. You are advised to manually set **client_encoding** to the server-side encoding format in the file.	An absolute path or relative path that meets the OS path naming convention.
-l, --list	Lists all available databases and then exits.	-
-v, --set=, --variable=NAME=VALUE	Sets gsql variable *NAME* to *VALUE*. For details about variable examples and descriptions, see **Variables**.	-
-X, --no-gsqlrc	Does not read the startup file (neither the system-wide **gsqlrc** file nor the user's **~/.gsqlrc** file). **NOTE** The startup file is **~/.gsqlrc** by default or it can be specified by the environment variable *PSQLRC*.	-
-1 ("one"), --single-transaction	When gsql uses the **-f** option to execute a script, "START TRANSACTION" and "COMMIT" are added to the start and end of the script, respectively, so that the script is executed as one transaction. This ensures that the script is executed successfully. If the script cannot be executed, the script is invalid. **NOTE** If the script already contains "START TRANSACTION", "COMMIT", and "ROLLBACK", this parameter is invalid.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range
-y, --slash-command	Terminates the current statement and sends it to the kernel for execution or re-executes executed statements (excluding gsql meta-commands). **NOTE** • This function applies to only ORA-compatible databases and does not support the **-c, --command** parameter. • In ORA-compatible mode, statements are separated using semicolons (;) by default. In the PL/SQL syntax, the slash (/) is used as the separator. When this parameter is used, the separator cannot be changed.	-
-?, --help	Displays help information about gsql command-line parameters, and exits.	-
-V, --version	Prints the gsql version information and exits.	-

Table 4-8 Input and output parameters

Parameter	Description	Value Range
-a, --echo-all	Prints all input lines to the standard output as they are read. **NOTE** When this parameter is used in some SQL statements, the sensitive information, such as user password, may be disclosed. Use this parameter with caution.	-
-B, --base64	Indicates that the **-c** parameter passes the content encoded using Base64. **NOTE** This parameter must be used with **-c**.	-
-e, --echo-queries	Displays all queries sent to the server to the standard output as well. **NOTE** When this parameter is used in some SQL statements, the sensitive information, such as user password, may be disclosed. Use this parameter with caution.	-
-E, --echo-hidden	Echoes the actual queries generated by **\d** and other backslash commands.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range
-k, --with-key=KEY	Uses gsql to decrypt imported encrypted files. **NOTE** • For key characters, such as the single quotation mark (') or double quotation mark (") in shell commands, Linux shell checks whether the input single quotation mark (') or double quotation mark (") matches. If no match is found, Linux shell does not enter the gsql program until the input is complete. • Stored procedures and functions cannot be decrypted and imported.	-
-L, --log-file=FILENAME	Writes normal output source and all query output into the file specified by *FILENAME*. **NOTE** • When this parameter is used in some SQL statements, the sensitive information, such as user password, may be disclosed. Use this parameter with caution. • This parameter retains only the query result in the corresponding file, so that the result can be easily found and parsed by other callers (for example, automatic O&M scripts). Logs about gsql operations are not retained.	An absolute path or relative path that meets the OS path naming convention.
-m, --maintenance	Allows a cluster to be connected when a two-phase transaction is being restored. **NOTE** The parameter is for developers only. When this parameter is used, gsql can be connected to the standby node to check data consistency between the primary and standby nodes.	-
-n, --no-libedit	Closes command line editing.	-
-o, --output=FILENAME	Puts all query output into the file specified by *FILENAME*.	An absolute path or relative path that meets the OS path naming convention.
-q, --quiet	Specifies the quiet mode. No additional information will be printed.	By default, gsql displays various information.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range
-s, --single-step	Runs in single-step mode. It indicates that the user is prompted before each command is sent to the server. This option can also be used for canceling execution. This option is mainly used to debug scripts. **NOTE** When this parameter is used in some SQL statements, the sensitive information, such as user password, may be disclosed. Use this parameter with caution.	-
-S, --single-line	Runs in single-line mode where a line break terminates a command.	-
-C, -C1, --enable-client-encryption=1	Enables the encrypted database feature. In this way, encrypted equality query is supported.	-
-C3, --enable-client-encryption=3	Enables a memory decryption emergency channel. The encrypted equality query and memory decryption emergency channel features are supported.	-

Table 4-9 Output format parameters

Parameter	Description	Value Range
-A, --no-align	Switches to unaligned output mode. The default output mode is aligned.	-
-F, --field-separator=STRING	Specifies the field separator, which is the vertical bar (|) by default.	-
-H, --html	Enables the HTML output format.	-
-P, --pset=VAR[=ARG]	Specifies the print option in the **\pset** format in the command line. **NOTE** The equal sign (=), instead of the space, is used here to separate the name and value. For example, enter **-P format=latex** to set the output format to **LaTeX**.	-
-R, --record-separator=STRING	Sets the record separator.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range
-r	Enables the insert mode on the client. It is disabled by default.	-
-t, --tuples-only	Prints only tuples.	-
-T, --table-attr=TEXT	Specifies options to be placed within the HTML table tag. Use this parameter with the **-H,--html** parameter to specify the output to the HTML format.	-
-x, --expanded	Displays in the expanded format.	-
-z, --field-separator-zero	Sets the field separator in the unaligned output mode to be blank. Use this parameter with the **-A, --no-align** parameter to switch to unaligned output mode.	-
-0, --record-separator-zero	Sets the record separator in the unaligned output mode to be blank. Use this parameter with the **-A, --no-align** parameter to switch to unaligned output mode.	-
-2, --pipeline	Uses a pipe to transmit the password. This parameter cannot be used on devices and must be used together with the **-c** or **-f** parameter.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-10 Connection parameters

Parameter	Description	Value Range
-h, --host=HOSTNAME	Specifies the host name of the running server, the UDS path, or the domain name. You can specify multiple host addresses by using character strings separated by commas (,), or specify an IPv6 host address. If multiple host addresses are specified, the faulty node is skipped by default and the first available primary node is automatically selected for connection. You can set the *PGTARGETSESSIONATTRS* environment variable to connect to different types of nodes. The values of the *PGTARGETSESSIONATTRS* variable are as follows: • **read-write**: readable and writable node. • **read-only**: read-only node. • **primary** or not set: primary node. • **standby**: standby node. • **prefer-standby**: preferred standby node. If there is no standby node, **any** is used. • **any**: no role check. **NOTE** Primary CNs and DNs are primary nodes. Generally, DNs are read-only nodes. You can set **PGTARGETSESSIONATTRS** to **read-write** to connect to CNs.	If the host name is omitted, gsql connects to the localhost over UDS or TCP/IP if no UDS is available.
-p, --port=PORT	Port number of the database server. You can configure one or more port numbers. When one port number is configured, all host addresses use the same port for connection. When multiple port numbers are configured, the sequence is the same as the host address sequence, and the number of port numbers must be the same as the number of host addresses. If they are different, an error is reported. You can modify the default port number using the **-p, --port=PORT** parameter.	The default port number can be specified by using compilation parameters. If the port number is not specified, the default port number **5432** is used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range
-U, --username=USERNAME	Specifies the user who connects to the database. **NOTE** • If this parameter is specified, you also need to enter your password for identity authentication when connecting to a database. You can enter the password interactively or use the **-W** parameter to specify a password. • To connect to a database, add an escape character before any dollar sign ($) in the username.	A string. The default user is the current user who operates the system.
-W, --password=PASSWORD	Specifies a password when the **-U** parameter is used to connect to the local database or a remote database. **NOTE** • When you attempt to connect to the CN after you have logged in to the server where the CN resides, the trust connection is used by default, and this parameter does not need to be set. • To connect to a database, add an escape character before any backslash (\) or back quote (`) in the password. • If this parameter is not specified but database connection requires your password, you will be prompted to enter your password in interactive mode. The maximum length of the password is 999 bytes, which is restricted by the maximum value of the GUC parameter **password_max_length**.	A string.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-11 O&M command of the UDS O&M channel

Parameter	Description	Value Range

-O, --emergency-command=EMERGENCYCOMMAND	O&M command of the UDS O&M channel.
	If gsql cannot connect to GaussDB in extreme cases, you can run this command for emergency O&M. This command communicates with GaussDB through the UDS O&M channel to implement one-click O&M.
	The parameters of the **-O** option are described as follows:
	`"{port} {command} [{param} [...]]"`
	{port}: database listening port.
	{command}: specific O&M command.
	{param}: parameter of the O&M command. There may be zero or more parameters.
	NOTE
	• When this option is used, the gsql tool must be located on the same server as the GaussDB process.
	• If the parameter of the O&M command contains spaces, enclose the parameter with quotation marks.
		• **"*{port}* mem_log"**: records the global memory overview, including details about the top 20 memory context usage at the global, thread, and session levels. This function is valid only when the GUC parameter **use_workload_manager** is enabled. The range is the same as that of the **mem_log** feature (recording memory overloading). There is no parameter. Example:
		`-O "19995 mem_log"`
		• **"*{port}* kill_wait_event *{wait_event}*"**: kills the session of a specified wait event. The *{wait_event}* parameter indicates information about locks, lightweight locks, and I/Os. Example:
		`-O "19995 kill_wait_event relation"`
		• **"*{port}* kill_business_sql"**: kills all service SQL statements of non-**rdsAdmin** users. There is no parameter. Example:
		`-O "19995 kill_business_sql"`
		• **"*{port}* kill_slow_sql *{username}*"**: kills SQL statements that are executed for more than 1 minute and are not in the idle state in a non-default database in batches. The *{username}* parameter specifies the username. Example:
		`-O "19995 kill_slow_sql omm"`
		• **"*{port}* kill_slow_sql *{username} {database}*"**: kills SQL statements that are executed for more

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range

		than 1 minute in a specified database by a specified user in batches. The *{username}* parameter specifies the username, and the *{database}* parameter specifies the database name. Example:
		`-O "19995 kill_slow_sql omm db1"`
		• "*{port}* **kill_executing_sql** *{username}*": kills the SQL statements that are being executed by a specified user whose status is **active** in a non-default database. The *{username}* parameter specifies the username. Example:
		`-O "19995 kill_executing_sql omm"`
		• "*{port}* **kill_mem_topn** *{top n}*": kills the SQL statements that occupy top *n* memory. The *{top n}* parameter specifies the top *n*. Example:
		`-O "19995 kill_mem_topn 10"`
		• "*{port}* **kill_cpu_topn** *{top n}*": kills the SQL statements that occupy top *n* CPU time. The *{top n}* parameter specifies the top *n*. Example:
		`-O "19995 kill_cpu_topn 10"`

Example

Example 1

Connect to a database and run the command.

Step 1 Connect to the GaussDB server using the gsql tool.

The gsql tool uses the **-d** parameter to specify the target database name, the **-U** parameter to specify the database username, the **-h** parameter to specify the host name, and the **-p** parameter to specify the port number. For details about the gsql parameters, see **Parameters**.

```
-- Connect to the 8000 port of the postgres database on the remote host as user jack. (Replace the parameter values as required.)
gsql -h 10.180.123.163 -d postgres -U jack -p 8000
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If the database name is not specified, the default database name generated during initialization is used.
- If the database username is not specified, the current OS user is used as the database username by default.
- When the preceding parameters (such as **-d** or **-U**) are not specified, if the database name is not specified in the connection command (**-d**), the parameter is interpreted as the database name; if the database name is specified (**-d**) but the database username is not specified (**-U**), this parameter is interpreted as the database username.

  For example, in the following command, **postgres** and **omm** are not options and are interpreted as the database name and username, respectively.

  ```
  gsql postgres omm -p 8000
  ```

  It is equivalent to the following command:

  ```
  gsql -d postgres -U omm -p 8000
  ```

Step 2 Execute gsql meta-commands and SQL statements.

```
-- Create the human_staff database.
gaussdb=# CREATE DATABASE human_staff;
CREATE DATABASE

-- List all databases and description information in GaussDB.
gaussdb=#  \l
                                 List of databases
     Name      |  Owner   | Encoding  | Collate | Ctype |   Access privileges
---------------+----------+-----------+---------+-------+-----------------------
 human_resource | omm | SQL_ASCII | C       | C     |
 postgres       | omm | SQL_ASCII | C       | C     |
 template0      | omm | SQL_ASCII | C       | C     | =c/omm          +
                |          |           |         |       | omm=CTc/omm
 template1      | omm | SQL_ASCII | C       | C     | =c/omm          +
                |          |           |         |       | omm=CTc/omm
 human_staff    | omm | SQL_ASCII | C       | C     |
 gaussdb        | omm | SQL_ASCII | C       | C     |
(6 rows)

-- Drop the human_staff database.
gaussdb=# DROP DATABASE human_staff;
DROP DATABASE
```

For details about gsql meta-commands, see **Meta-Command Reference**.

----End

Example 2

Obtain the help information.

```
gsql --help
gsql is the GaussDB Kernel interactive terminal.

Usage:
  gsql [OPTION]... [DBNAME [USERNAME]]

General options:
  -c, --command=COMMAND    run only single command (SQL or internal) and exit
  -d, --dbname=DBNAME      database name to connect to (default: "omm")
  -f, --file=FILENAME      execute commands from file, then exit
  -l, --list               list available databases, then exit
  -v, --set=, --variable=NAME=VALUE
                           set gsql variable NAME to VALUE
  -V, --version            output version information, then exit
  -X, --no-gsqlrc          do not read startup file (~/.gsqlrc)
  -1 ("one"), --single-transaction
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
                      execute command file as a single transaction
  -?, --help              show this help, then exit

Input and output options:
  -a, --echo-all          echo all input from script
  -e, --echo-queries      echo commands sent to server
  -E, --echo-hidden       display queries that internal commands generate
  -k, --with-key=KEY      the key for decrypting the encrypted file
  -L, --log-file=FILENAME  send session log to file
  -m, --maintenance       can connect to cluster during 2-pc transaction recovery
  -n, --no-libedit        disable enhanced command line editing (libedit)
  -o, --output=FILENAME   send query results to file (or |pipe)
  -q, --quiet             run quietly (no messages, only query output)
  -C, -C1, --enable-client-encryption=1
                      enable client encryption feature
  -C2, --enable-client-encryption=2
                      enable client sorting based client encryption feature
  -C3, --enable-client-encryption=3
                      enable Trusted Domain operation based on client encryption feature

  -s, --single-step       single-step mode (confirm each query)
  -S, --single-line       single-line mode (end of line terminates SQL command)
  -y, --slash-command     enable slash command to re-execute query in buffer

Output format options:
  -A, --no-align          unaligned table output mode
  -F, --field-separator=STRING
                      set field separator (default: "|")
  -H, --html              HTML table output mode
  -P, --pset=VAR[=ARG]    set printing option VAR to ARG (see \pset command)
  -R, --record-separator=STRING
                      set record separator (default: newline)
  -r                      if this parameter is set,use libedit
  -t, --tuples-only       print rows only
  -T, --table-attr=TEXT   set HTML table tag attributes (e.g., width, border)
  -x, --expanded          turn on expanded table output
  -z, --field-separator-zero
                      set field separator to zero byte
  -0, --record-separator-zero
                      set record separator to zero byte
  -2, --pipeline          use pipeline to pass the password, forbidden to use in terminal
                      must use with -c or -f

Connection options:
  -h, --host=HOSTNAME     database server host or socket directory (default: "/data/huawei/wisequery/
perfadm_mppdb")
                      allow multi host IP address with comma separator, Use the
                      PGTARGETSESSIONATTRS(default: "primary") variable to select the
                      IP address to be connected.
                      (PGTARGETSESSIONATTRS can be {read-write|read-only|primary|
                       standby|prefer-standby|any})
  -p, --port=PORT         database server port (default: "5432")
  -U, --username=USERNAME  database user name (default: "omm")
  -W, --password=PASSWORD  the password of specified database user

Emergency O&M command:
  -O, --emergency-command=EMERGENCYCOMMAND
                      emergency O&M command

For more information, type "\?" (for internal commands) or "\help" (for SQL
commands) from within gsql, or consult the gsql section in the GaussDB Kernel
documentation.
```

Example 3

Run an O&M command through the UDS O&M channel. The following is a simple example of how to check the memory usage through the UDS O&M channel:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Step 1 Log in to the server where the GaussDB node is located and run the following gsql command to check whether the memory usage records, such as memory overview and memory context, are generated:

```
gsql -O "{port} mem_log"
```

{port} specifies the database node name. If **true** is displayed, the detailed information is generated. If the execution fails, the failure cause is displayed.

```
-- In the example, 19995 is the database listening port.
gsql -O "19995 mem_log"
true
```

Step 2 Check the content of the **mem_log** file generated in the *$GAUSSLOG*/**mem_log/***{nodename}* directory. *{nodename}* specifies the database node name.

```
-- In the example, datanode specifies the database node name.
ls -l $GAUSSLOG/mem_log/datanode
total 8
-rw-------. 1 omm omm 7543 May 28 15:15 mem_log-2025-05-28_151518.log
```

Step 3 Run the following command to view the file:

```
cat $GAUSSLOG/mem_log/datanode/mem_log-time.log
```

mem_log-time.log is the file generated in the *$GAUSSLOG*/**mem_log/***datanode* directory in **Step 2**. (The file name extension is .log.)

----End

4.2.1.2 Meta-Command Reference

This section describes meta-commands provided by gsql after GaussDB CLI tool is used to connect to a database. After connecting to the database, you can run the **\?** command to display the help information about all available meta-commands.

Precautions

- The format of the gsql meta-command is a backslash (\) followed by a command verb, and then a parameter. The parameters are separated from the command verb and from each other by any number of whitespace characters.
- To include whitespace characters into an argument, you must quote them with single quotation marks. To include a single quotation mark into such an argument, precede it by a backslash. Anything contained in single quotation marks is furthermore subject to C-like substitutions for \n (new line), \t (tab), \b (backspace), \r (carriage return), \f (form feed), \digits (octal), and \xdigits (hexadecimal).
- Text enclosed in backquotes (`) is taken as a command line input to the shell. The command output (without new line characters at the end) replaces the text enclosed in backquotes (`) in the original argument, which is used as the argument value.
- If an unquoted argument begins with a colon (:), the argument is taken as a gsql variable and the value of the variable is used as the argument value instead.
- Some commands take an SQL identifier (such as a table name) as a parameter. These parameters follow the SQL syntax rules: Unquoted letters are forced to lowercase, while double quotation marks ("") protect letters from case conversion and allow incorporation of whitespace into the identifier. Within double quotation marks, paired double quotation marks

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

reduce to a single double quotation mark in the result name. For example, **FOO"BAR"BAZ** is interpreted as **fooBARbaz**, and **"Aweird""name"** becomes **Aweird"name**.

- The analysis of parameters stops at another unquoted backslash (\), which is considered the beginning of a new meta-command. Parameters end with two backslashes (\\) and then SQL statements can be analyzed, if any. In this way, SQL and gsql commands can be used together in a line. But in any case, the arguments of a meta-command cannot continue beyond the end of the line.
- M-compatible databases do not support the **\h** meta-command.

Commands and Parameters

For details about meta-commands, see **Table 4-12**, **Table 4-13**, **Table 4-14**, **Table 4-15**, **Table 4-17**, **Table 4-19**, **Table 4-20**, **Table 4-21**, and **Table 4-23**.

NOTE

FILE mentioned in the following commands indicates a file path. This path can be an absolute path such as **/home/gauss/file.txt**, or a relative path such as **file.txt**. By default, a **file.txt** is created in the path where the user runs gsql commands.

Table 4-12 Common meta-commands

Parameter	Description	Value Range
\copyright	Displays GaussDB version and copyright information.	-
\g [FILE] or ;	Performs a query operation and sends the result to a file or pipe.	-
\h(\help) [NAME]	Provides syntax help on the specified SQL statement.	If the name is not specified, then gsql will list all the commands for which syntax help is available. If the name is an asterisk (*), syntax help on all SQL statements is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range
\parallel [on [num]|off]	Controls the parallel execution function. • **on**: The switch is enabled and the maximum number of concurrently executed tasks is **num**. The default value of **num** is **1024**. • **off**: This switch is disabled. **NOTE** • Parallel execution is not allowed in a running transaction and a transaction is not allowed to be started during parallel execution. • Parallel execution of **\d** meta-commands is not allowed. • If SELECT statements are run concurrently, the return results may be displayed randomly but acceptable; however, if a core dump or process response failure occurs, it is not acceptable. • It is not recommended that you run **SET** statements in concurrent tasks because they may cause unexpected results. • Creation of a temporary table is not supported. If temporary tables are required, create them before parallel execution is enabled, and use them only in the parallel execution. Temporary tables cannot be created in parallel execution. • When **\parallel** is executed, *num* independent gsql processes can be connected to the database server. • The total duration of all **\parallel** tasks cannot exceed **session_timeout**. Otherwise, the connection may fail during concurrent execution. • One or more commands following **\parallel on** will be executed only after **\parallel off** is executed. Therefore, **\parallel on** must be followed by **\parallel off**. Otherwise, the commands following **\parallel on** cannot be executed. • The maximum number of connections allowed by the server is determined based on **max_connection** and the number of current connections. Set **num** based on the allowed number of connections.	-
\q	Exits the gsql program. This command is executed in a script file only when the script terminates.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-13 Query buffer meta-commands

Parameter	Description
\e [FILE] [LINE]	Uses an external editor to edit the query buffer or file.
\ef [FUNCNAME [LINE]]	Edits the function definition using an external editor. If **LINE** is specified, the cursor will point to the specified line of the function body.
\p	Prints the current query buffer to the standard output.
\r	Resets or clears the query buffer.
\w FILE \w |COMMAND	Outputs the current query buffer to a file specified by **FILE**. Sends the current query buffer to the shell command specified by **COMMAND**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-14 Input/Output commands

Parameter	Description
\copy { table [(column_list)] | (query) } { from | to } { filename | stdin | stdout | pstdin | pstdout } [copy_parameter] [parallel integer] [NO_SPLIT_FILE] [FIELD_NO_EOL]	After logging in to the database on any gsql client, you can import and export data. This is an operation of running the **SQL COPY** command, but not the server that reads or writes data to a specified file. Instead, data is transferred between the server and the local file system. In this way, local user permissions instead of server permissions are required for file access, and the user does not need to be the initial user.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
	NOTE ● \COPY only applies to small-batch data import with uniform formats. GDS or COPY is preferred for data import. ● \COPY specifies the number of clients to import data to implement parallel import of data files. Currently, the value range is [1, *Twice the number of CPU cores of the client*]. ● When **\COPY** specifies the **parallel** parameter for parallel import, multiple threads are started to import data. Each thread is an independent transaction. Therefore, data may be partially imported. Exercise caution when importing data in parallel after fully understanding the possible results of parallel import. ● \COPY can specify **pstdin** and **pstdout**. **pstdin** declares that the input is the standard input from gsql, and **pstdout** declares that the output is the standard output from gsql. ● The parallel import using \COPY has the following limitations: Parallel import of temporary tables is not supported. Parallel import within a transaction is not supported. Parallel import of binary files is not supported. Parallel import of data encrypted using AES-128 is not supported. Parallel import is not supported when the **COPY** option contains **EOL**. In these cases, even if the **parallel** parameter is specified, the non-parallel process is still executed. If the data file contains fewer than 100 lines, the non-parallel process is also executed. Parallel import is not supported when a sequence is specified. Otherwise, an error is reported. If a table (for example, **t1**) with a trigger inserting data to the table is imported in parallel and the table has constraints, a deadlock may occur due to parallel import. Specifically, multiple parallel import processes operate the **t1** table. Due to constraints (such as the UNIQUE and FOREIGN KEY), lock wait occurs between different transactions. If a loop is formed during the wait, the database deadlock detection mechanism is triggered. As a result, the import fails and a deadlock error is reported. Therefore, if the target table with constraints contains a trigger inserting data into the table, parallel import is not recommended. When parallel import is performed on a table with constraints (such as the UNIQUE constraint), if the LOAD syntax is not used and the data file contains data that violates the constraints, a deadlock may occur due to parallel import. Specifically, multiple parallel import threads operate the **t1** table. Due to constraints (such as UNIQUE and FOREIGN KEY), lock wait occurs between different transactions. If a loop is formed during the wait, the database deadlock detection mechanism is triggered. As a result, the data import fails and an error is reported. Therefore, if the target table contains constraints (such as UNIQUE KEY), you are advised to use the LOAD syntax for parallel import. The **FIELD_NO_EOL** option takes effect when the value of **parallel** is greater than 1. With the **FIELD_NO_EOL** option, the data import speeds up. However, you need to ensure that the data rows in the imported data file do not contain newline characters. If the data rows contain newline characters, the following problems may occur: ● A complete data row is split into two parts by the newline character in the data field. The two parts are allocated to different threads for sending data to the client. As a result, the inserted field does not comply with the original table definition. ● Once **SKIP** is used, the specified number of data rows will be skipped. A newline character in the field may cause a row of data to be incorrectly considered as two rows of data. As a result, the number of data rows skipped is incorrect. ● The **NO_SPLIT_FILE** option added to the \COPY statement takes effect when the value of **parallel** is greater than 1. When this option is added, the original parallelism logic is kept and maximum value of DOP is **8**. By default, the DOP upper limit 8 is removed and data files are automatically split to implement parallelism.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
\echo [STRING]	Writes character strings to the standard output.
prompt [STRING]	Writes character strings to the standard output, which is equivalent to **\echo**.
\i FILE	Reads content from *FILE* and uses them as the input for a query.
\i+ FILE KEY	Runs commands in an encrypted file.
\ir FILE	Similar to **\i**, but resolves relative path names differently.
\ir+ FILE KEY	Similar to **\i+**, but resolves relative path names differently.
\o [FILE]	Sends all subsequent query results to the file specified by **FILE**.
\o [| COMMAND]	Sends all subsequent query results to the shell command specified by **COMMAND**.
\qecho [STRING]	Writes character strings to the query output flow.
\print_copy_lines [on | off]	Specifies whether to print "COPY XXX" in the output of the **COPY FROM STDIN** command. This function is enabled by default. • **on**: Running the **COPY FROM STDIN** command will print the number of data rows that have been imported. • **off**: Running the **COPY FROM STDIN** command does not print how many rows of data have been imported.

Table 4-15 Information display meta-commands

Parameter	Description	Value Range	Example
\d[S+]	Lists all tables, views, and sequences of all schemas in **search_path**. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed.	-	Lists all tables, views, and sequences of all schemas in **search_path**. `gaussdb=# \d`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range	Example
\d[S+] NAME	Lists the structure of specified tables, views, and indexes. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed.	-	Lists the structure of table **a**. `gaussdb=# \dtable+ a`
\d+ [PATTERN]	Lists all tables, views, and indexes. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only tables, views, and indexes whose names match **PATTERN** are displayed.	-	Lists all tables, views, and indexes whose names start with **f**. `gaussdb=# \d+ f*`
\da[S] [PATTERN]	Lists all available aggregate functions, together with the data type they perform operations on and the return value types. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only aggregate functions whose names match **PATTERN** are displayed.	-	Lists all available aggregate functions whose names start with **f**, together with their return value types and the data types. `gaussdb=# \da f*`
\db[+] [PATTERN]	Lists all available tablespaces. If **PATTERN** is specified, only tablespaces whose names match **PATTERN** are displayed.	-	Lists all available tablespaces whose names start with **p**. `gaussdb=# \db p*`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range	Example
\dc[S+] [PATTERN]	Lists all available conversions between character-set encodings. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only conversions whose names match **PATTERN** are displayed.	-	Lists all available conversions between character-set encodings. `gaussdb=# \dc *`
\dC[+] [PATTERN]	Lists all available type conversions. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only conversions whose names match **PATTERN** are displayed.	PATTERN must be the actual type name and cannot be an alias.	Lists all type conversions whose patten names start with **c**. `gaussdb=# \dC c*`
\dd[S] [PATTERN]	Displays all visible objects. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only matched objects are displayed. The objects include aggregations, functions, operators, types, relations (tables, views, indexes, and sequences), and rules.	-	Lists all visible objects. `gaussdb=# \dd`
\ddp [PATTERN]	Lists all default permissions. If **PATTERN** is specified, only permissions whose names match **PATTERN** are displayed.	-	Lists all default permissions. `gaussdb=# \ddp`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range	Example
\dD[S+] [PATTERN]	Lists all available domains. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only domains whose names match **PATTERN** are displayed.	-	Lists all available domains. `gaussdb=# \dD`
\det[+] [PATTERN]	Lists all foreign tables. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only tables whose names match **PATTERN** are displayed.	-	Lists all foreign tables. `gaussdb=# \det`
\des[+] [PATTERN]	Lists all foreign servers. If **PATTERN** is specified, only servers whose names match **PATTERN** are displayed.	-	Lists all foreign servers. `gaussdb=# \des`
\deu[+] [PATTERN]	Lists all user mappings. If **PATTERN** is specified, only information whose name matches **PATTERN** is displayed.	-	Lists all user mappings. `gaussdb=# \deu`
\dew[+] [PATTERN]	Lists all encapsulated external data. If **PATTERN** is specified, only data whose name matches **PATTERN** is displayed.	-	Lists all encapsulated external data. `gaussdb=# \dew`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range	Example
\df[antw] [S+] [PATTERN]	Lists all available functions, together with their parameters and return types. **a** indicates an aggregate function, **n** indicates a common function, **t** indicates a trigger, and **w** indicates a window function. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only functions whose names match **PATTERN** are displayed.	-	Lists all available functions, together with their parameters and return types. `gaussdb=# \df`
\dF[+] [PATTERN]	Lists all text search configuration information. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only configurations whose names match **PATTERN** are displayed.	-	Lists all text search configuration information. `gaussdb=# \dF+`
\dFd[+] [PATTERN]	Lists all text search dictionaries. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only dictionaries whose names match **PATTERN** are displayed.	-	Lists all text search dictionaries. `gaussdb=# \dFd`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range	Example
\dFp[+] [PATTERN]	Lists all text search analyzers. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only analyzers whose names match **PATTERN** are displayed.	-	Lists all text search analyzers. `gaussdb=# \dFp`
\dFt[+] [PATTERN]	Lists all text search templates. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only templates whose names match **PATTERN** are displayed.	-	Lists all text search templates. `gaussdb=# \dFt`
\dg[+] [PATTERN]	Lists all database roles. If **PATTERN** is specified, only roles whose names match **PATTERN** are displayed. **NOTE** Since the concepts of "users" and "groups" have been unified into "roles", this command is now equivalent to **\du**. Both commands are retained to ensure compatibility with earlier versions.	-	Lists all database roles named **j?e** (the question mark (?) indicates any character). `gaussdb=# \dg j?e`
\dL[S+] [PATTERN]	Lists all available program languages. If **PATTERN** is specified, only languages whose names match **PATTERN** are displayed.	-	Lists all available program languages. `gaussdb=# \dL`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range	Example
\dm[S+] [PATTERN]	Lists materialized views. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only materialized views whose names match **PATTERN** are displayed.	-	Lists materialized views. `gaussdb=# \dm`
\dn[S+] [PATTERN]	Lists all schemas (namespace). If **+** is added to the command, the permission and description of each schema are listed. If **PATTERN** is specified, only schemas whose names match the pattern are shown. By default, only schemas you created are displayed.	-	Lists information about all schemas whose names start with **d**. `gaussdb=# \dn+ d*`
\do[S] [PATTERN]	Lists available operators with their operand and return types. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only operators whose names match **PATTERN** are displayed. By default, only the operators created by the user are listed.	-	Lists available operators with their operand and return types. `gaussdb=# \do`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range	Example
\dO[S+] [PATTERN]	Lists collation rules. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only rules whose names match **PATTERN** are displayed. By default, only user-created rules are shown.	-	Lists collation rules. gaussdb=# \dO
\dp [PATTERN]	Lists tables, views, and related permissions. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only tables and views whose names match the pattern are shown.	-	Lists tables, views, and related permissions. gaussdb=# \dp The following result about **\dp** is displayed: rolename=xxxx/yyyy -- Assigns permissions to a role. =xxxx/yyyy -- Assigns permissions to public. *xxxx* indicates the assigned permissions, and *yyyy* indicates the roles with the assigned permissions. For details about permission descriptions, see **Table 4-16**.
\drds [PATTERN1 [PATTERN2]]	Lists all parameters that have been modified. These settings can be for roles, for databases, or for both. **PATTERN1** and **PATTERN2** indicate a role pattern and a database pattern, respectively. If **PATTERN** is specified, only rules whose names match **PATTERN** are displayed. If the default value is used or * is specified, all settings are listed.	-	List all the modified configuration parameters of the database. gaussdb=# \drds * *dbname*

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range	Example
\dT[S+] [PATTERN]	Lists all data types. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. If **PATTERN** is specified, only types whose names match **PATTERN** are displayed.	-	Lists all data types. `gaussdb=# \dT`
\du[+] [PATTERN]	Lists all database roles. If **PATTERN** is specified, only roles whose names match **PATTERN** are displayed. **NOTE** Since the concepts of "users" and "groups" have been unified into "roles", this command is now equivalent to **\dg**. Both commands are retained to ensure compatibility with earlier versions.	-	Lists all database roles. `gaussdb=# \du`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range	Example
\dE[S+] [PATTERN] \di[S+] [PATTERN] \ds[S+] [PATTERN] \dt[S+] [PATTERN] \dv[S+] [PATTERN]	In this group of commands, the letters E, i, s, t, and v stand for foreign table, index, sequence, table, and view, respectively. You can specify any or a combination of these letters sequenced in any order to obtain an object list. When objects with the same name exist in different schemas in **search_path**, only the object in the schema that ranks first in **search_path** is displayed. For example, **\dit** lists all indexes and tables. If **PATTERN** is specified, only objects whose names match **PATTERN** are displayed. By default, only objects you created are displayed. You can specify **PATTERN** or **S** to view other system objects.	-	Lists all indexes and views. `gaussdb=# \div`
\dx[+] [PATTERN]	Lists installed extensions. If **PATTERN** is specified, only extensions whose names match **PATTERN** are displayed.	-	Lists installed extensions. `gaussdb=# \dx`
\l[+]	Lists the names, owners, character set encodings, and permissions of all the databases in the server.	-	List the names, owners, character set encodings, and permissions of all the databases in the server. `gaussdb=# \l`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description	Value Range	Example

\sf[+] FUNCNAME	Displays the definition of a function.
	NOTE
	• If the function name contains parentheses, enclose the function name with double quotation marks and add the parameter type list following the double quotation marks. Also enclose the list with parentheses.
	• If a function with the same name exists, the definitions of multiple functions are returned.
		-	Assume a function **function_a** and a function **func()name**. This parameter will be as follows:
			`gaussdb=# \sf function_a`
			`gaussdb=# \sf "func()name"(argtype1, argtype2)`

\z [PATTERN]	Lists all tables, views, and sequences in the database and their access permissions. If a pattern is given, it is a regular expression, and only matched tables, views, and sequences are shown.	-	Lists all tables, views, and sequences in the database and their access permissions. `gaussdb=# \z`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- In Table 4-15, **S** indicates that the system object is displayed and **+** indicates that additional object information is displayed.
- **PATTERN** specifies the name of the object to be displayed. Pay attention to the following points about **PATTERN**:
  - In the simplest case, PATTERN is the exact name of the object. Characters in **PATTERN** are usually converted to lowercase (as in SQL names), for example, **\dt FOO** will display a table named **foo**. As in SQL names, placing double quotation marks (") around a pattern prevents them being folded to lower case. If you need to include a double quotation mark (") in a pattern, write it as a pair of double quotation marks ("") within a double-quote sequence, which is in accordance with the rules for SQL quoted identifiers. For example, **\dt "FOO""BAR"** will be displayed as a table named **FOO"BAR** instead of **foo"bar**. You cannot put double quotation marks around just part of a pattern, which is different from the normal rules for SQL names. For example, **\dt FOO"FOO"BAR** will be displayed as a table named **fooFOObar** if just part of a pattern is quoted.
  - Whenever the **PATTERN** parameter is omitted completely, the **\d** commands display all objects that are visible in the current schema search path, which is equivalent to using an asterisk (*) as the pattern. An object is regarded to be visible if it can be referenced by name without explicit schema qualification. To see all objects in the database regardless of their visibility, use a dot within double quotation marks (*.*) as the pattern.
  - Within a pattern, the asterisk (*) matches any sequence of characters (including no characters) and a question mark (?) matches any single character. This notation is comparable to Unix shell file name patterns. For example, **\dt int*** displays tables whose names start with **int**. But within double quotation marks, the asterisk (*) and the question mark (?) lose these special meanings and are just matched literally.
  - A pattern that contains a dot (.) is interpreted as a schema name pattern followed by an object name pattern. For example, **\dt foo*.*bar*** displays all tables (whose names include **bar**) in schemas starting with **foo**. If no dot appears, then the pattern matches only visible objects in the current schema search path. Likewise, the dot within double quotation marks loses its special meaning and becomes an ordinary character.
  - Senior users can use regular-expression notations, such as character classes. For example [0-9] can be used to match any digit. All regular-expression special characters work as specified in POSIX. The following characters are excluded:
    - A dot (.) is used as a delimiter.
    - An asterisk (*) is translated into an asterisk prefixed with a dot (.*), which is a regular-expression marking.
    - A question mark (?) is translated into a dot (.).
    - A dollar sign ($) is matched literally.
  - You can write ?, (*R*+|), (*R*|), and *R* to the following pattern characters: ., *R*.*, and *R*.?. The dollar sign ($) does not need to be used as a regular expression character because **PATTERN** must match the entire name instead of being interpreted as a regular expression (in other words, $ is automatically appended to **PATTERN**). If you do not expect a pattern to be anchored, write an asterisk (*) at its beginning or end. All regular-expression special characters within double quotation marks lose their special meanings and are matched literally. Regular-expression special characters in operator name patterns (such as the **\do** parameter) are also matched literally.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-16 Description of permissions

Parameter	Description
r	**SELECT**: allows users to read data from specified tables and views.
w	**UPDATE**: allows users to update columns for specified tables.
a	**INSERT**: allows users to insert data to specified tables.
d	**DELETE**: allows users to delete data from specified tables.
D	**TRUNCATE**: allows users to delete all data from specified tables.
x	**REFERENCES**: allows users to create a FOREIGN KEY constraint (not supported in distributed mode).
t	**TRIGGER**: allows users to create a trigger on specified tables.
X	**EXECUTE**: allows users to use specified functions and the operators that are realized by the functions.
U	**USAGE**: ● For procedural languages, allows users to specify a procedural language when creating a function. ● For schemas, allows users to access objects included in specified schemas. ● For sequences, allows users to use the nextval function.
C	**CREATE**: ● For databases, allows new schemas to be created within the database. ● For schemas, allows users to create objects in a schema. ● For tablespaces, allows users to create tables in a tablespace and set the tablespace to default one when creating databases and schemas.
c	**CONNECT**: allows users to connect to specified databases.
T	**TEMPORARY**: allows users to create temporary tables.
A	**ALTER**: allows users to modify the attributes of a specified object.
P	**DROP**: allows users to delete specified objects.
m	**COMMENT**: allows users to define or modify comments of a specified object.
i	**INDEX**: allows users to create indexes on specified tables.
v	**VACUUM**: allows users to perform ANALYZE and VACUUM operations on specified tables.
*	Grants the preceding permissions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-17 Formatting meta-commands

Parameter	Description	Value Range
\a	Switches between aligned and unaligned table output formats.	-
\C [STRING]	Sets the title for the table output, or cancels the title (if there is no parameter).	Any character string or null.
\f [STRING]	Specifies the character or string used to separate fields in unaligned output mode.	Any character string.
\H	Switches between standard output and HTML table output.	-
\pset NAME [VALUE]	Sets options affecting the output of query result tables.	For details about the value of **NAME**, see **Table 4-18**.
\t [on | off]	Switches the table output format.	• **on**: Only data rows are displayed. Column titles and row counts are not displayed. • **off**: The entire table, including the column titles and row counts, is displayed.
\T [STRING]	Specifies attributes to be placed within the table tag in HTML output format. If this parameter is left empty, no setting is performed.	-
\x [on | off | auto]	Switches the display mode of the extended rows.	• **on**: The extended display mode is enabled. Each field is displayed in a separate row. • **off**: The extended display mode is disabled and the standard table display mode is used. • **auto**: The extended display mode is used depending on the width of the query result.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-18 Adjustable printing options

Option	Description	Value Range
border	Controls the border style of the table output.	• The value is an integer greater than 0 in HTML format. • The value range in other formats is as follows: – **0**: no border. No border is displayed when the table is output. – **1**: inner borders dividing columns, without outside borders. – **2**: all borders, including the outside and inner borders.
expanded (or x)	Switches between regular and expanded formats.	• **on**: The extended display mode is enabled. Each field is displayed in a separate row. • **off**: The extended display mode is disabled and the standard table display mode is used. The standard format is effective only in the aligned and wrapped modes. • **auto**: The extended display mode is used depending on the width of the query result.
fieldsep	Specifies the delimiter between columns in unaligned output mode.	Any character string. Common values include **","**, **"\t"**, and **"|"**. If this option is not set, spaces are used as delimiters by default.
fieldsep_zero	Sets the delimiter between columns in unaligned output mode to ASCII NUL.	-
footer	Sets the footer of the query result.	• **on**: The footer information is displayed. • **off**: The footer information is not displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Option	Description	Value Range
format	Specifies the output format of the query result.	• **unaligned (u)**: Columns are displayed in unaligned mode and are separated by the current delimiter. • **aligned (a)**: The table is displayed in aligned mode. • **wrapped (w)**: The table is displayed similar to the aligned mode but automatically wraps any excessively long value. • **html (h)**: The table is displayed in HTML table format. • **latex (l)**: The table is displayed in LaTeX format. • **troff-ms (t)**: The table is displayed in the troff ms macro format.
null	Specifies the display mode of NULL values.	Any character string. If this option is not set, whitespace characters are used to replace NULL values by default.
numericlocale	Determines whether to use the locale of the current system when numbers are output.	• **on**: The locale settings of the current system are used for formatting when numbers are output. For example, the number 1,000.00 is displayed as 1,000.00 in English and 1.000,00 in German. • **off**: The standard numeric format, such as 1000.00, is used for number output.
pager	Determines the use of a pager for query and gsql help outputs. If the *PAGER* environment variable is set, the output is redirected to the specified program. Otherwise, the platform-dependent default value is used.	• **on**: The pager is used for terminal output that does not fit the screen. • **off**: The pager is not used. • **always**: The pager is used for all terminal output regardless of whether it fits the screen.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Option	Description	Value Range
recordsep	Sets the record delimiter in unaligned output mode.	Any character string. Common values include the newline characters (\n), carriage return character (\r), and carriage return line feed character (\r\n). If this option is not set, the newline character (\n) is used as the record delimiter by default.
recordsep_zero	Sets the record delimiter between columns in unaligned output mode to ASCII NUL.	-
tableattr (or T)	Specifies attributes to be placed inside the HTML table tag in HTML output format (such as cellpadding or bgcolor). Note: If the **\pset** border command has been executed, you do not need to set the border.	For example, bgcolor=#FFFFFF. If no value is given, the table attributes are not set.
title	Specifies the title of the query result.	Any character string. If this option is not set, no title is displayed in the query result.
tuples_only (or t)	Enables or disables the tuples-only mode. Full display may show extra information, such as column headers, titles, and various footers. In tuples-only mode, only the table data is shown.	-
feedback	Specifies whether to output the number of result lines.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-19 Connection meta-commands

Parameter	Description	Value Range
\c[onnect] [DBNAME|- USER|- HOST|- PORT|-]	Connects to a new database. If a database name contains more than 63 bytes, only the first 63 bytes are valid and are used for connection. However, the database name displayed in the command line of gsql is still the name before the truncation. **NOTE** If the database login user is changed during reconnection, you need to enter the password of the new user. The maximum length of the password is 999 bytes, which is restricted by the maximum value of the GUC parameter **password_max_length**.	-
\encoding [ENCODING]	Sets the client character set encoding.	Without an argument, this command shows the current encoding.
\conninfo	Prints information about the current connected database.	-

Table 4-20 OS meta-commands

Parameter	Description	Value Range
\cd [DIR]	Changes the current working directory.	An absolute path or relative path that meets the OS path naming convention.
\setenv NAME [VALUE]	Sets the *NAME* environment variable to **VALUE**. If **VALUE** is not provided, do not set the environment variable.	-
\timing [on | off]	Toggles a display of how long each SQL statement takes, in milliseconds (exclude the time of screen displaying).	• **on**: The display is enabled. • **off**: The display is disabled.
\! [COMMAND]	Runs OS commands. If no command follows **\!**, running it will open an independent shell UI for user interaction.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-21 Variable meta-commands

Parameter	Description
\prompt [TEXT] NAME	Prompts the user to enter a value and stores it in a gsql variable. *TEXT* indicates the text that prompts the user to enter, and *NAME* indicates the gsql variable that stores the input value. Example: gaussdb=# \prompt 'Enter area name: ' area Enter area name: Asia gaussdb=# SELECT * FROM HR.areaS WHERE area_name = :'area'; area_id | area_name ---------+----------- 3 | Asia (1 row)
\set [NAME [VALUE]]	Sets the internal variable *NAME* to **VALUE**. If more than one value is given, the variable value is the concatenation result of all values. If no second argument is given, the variable is just set with no value. Some common variables are processed differently in gsql and they are combinations of uppercase letters, numbers and underscores. **Table 4-22** is a list of all variables that are processed differently.
\unset NAME	Deletes the variable name of gsql.

Table 4-22 Common **\set** commands

Command	Command Description	Value Range
\set VERBOSITY value	Controls the redundant lines in error reports.	• **default**: Critical and major error texts and text locations, error details, and error messages (possibly involving multiple lines) are all returned. • **verbose**: All error information is returned. • **terse**: Only critical and major error texts and text locations are returned (which is generally suitable for single-line error information).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Command	Command Description	Value Range
\set ON_ERROR_STOP value	If this variable is set, the script execution stops immediately. If this script is called from another script, that script will be stopped immediately as well. If the primary script is called using the **-f** option rather than from one gsql session, gsql will return error code **3**, indicating the difference between the current error and critical errors. (The error code for critical errors is **1**.)	• **on**: The execution stops if an error occurs. In interactive mode, gsql returns the output of executed commands immediately. • **off**: An error, if occurring during the execution, is ignored, and the execution continues.
\set AUTOCOMMIT [on | off]	Controls the auto commit behavior of the current gsql connection. By default, the gsql connection is automatically committed, and each individual statement is implicitly committed. If auto commit is disabled for performance or other purposes, you need to explicitly run the **COMMIT** command to ensure that transactions are committed. For example, execute the COMMIT statement to explicitly commit transactions after a specified service SQL statement is executed. Particularly, ensure that all transactions are committed before the gsql client exits. **NOTE** Auto commit is enabled in gsql by default. If you disable it, all the statements executed later will be packaged in implicit transactions, and you cannot execute statements that cannot be executed within a transaction.	• **on**: Auto commit is enabled. • **off**: Auto commit is disabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-23 Fully-encrypted meta-commands

Parameter	Description
\send_token	A full encryption function used to send keys to the server for caching. This function is used when the memory decryption emergency channel is enabled.
\st	A full encryption function used to send keys to the server for caching. This function is used when the memory decryption emergency channel is enabled.
\clear_token	A full encryption function used to destroy the keys cached on the server. This function is used when the memory decryption emergency channel is enabled.
\ct	A full encryption function used to destroy the keys cached on the server. This function is used when the memory decryption emergency channel is enabled.
\key_info KEY_INFO	A full encryption function used to set the parameters for accessing an external key manager.

NOTE

- The fully-encrypted memory decryption emergency channel is not supported in a distributed system.
- The M-compatible mode does not support fully-encrypted databases.

Table 4-24 Common simple O&M meta-commands

Simple O&M Command	Description	SQL Statement
\view stmt_track {unique_sql_id}	**stmt_track**: records the level-2 full SQL information of a specified SQL statement. **unique_sql_id**: specifies the unique SQL ID of the SQL statement to be recorded.	SELECT * FROM pg_catalog.dynamic_func_control('LOCAL', 'STMT', 'TRACK', '{"{unique_sql_id}", "L2"}');
\view stmt_untrack {unique_sql_id}	**stmt_untrack**: cancels the recording of the level-2 full SQL information of a specified SQL statement. **unique_sql_id**: specifies the unique SQL ID of the SQL statement whose recording needs to be canceled.	SELECT * FROM pg_catalog.dynamic_func_control('LOCAL', 'STMT', 'UNTRACK', '{"{unique_sql_id}"}');

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Simple O&M Command	Description	SQL Statement
\view stmt_list	**stmt_list**: queries the record list.	SELECT * FROM pg_catalog.dynamic_func_control('LOCAL', 'STMT', 'LIST', '{}');
\view stmt_clean	**stmt_clean**: clears the record list.	SELECT * FROM pg_catalog.dynamic_func_control('LOCAL', 'STMT', 'CLEAN', '{}');
\view explain_and_stack {pid}	**explain_and_stack**: queries the runtime plan and stack of a specified session. **pid**: specifies the PID of a session.	SELECT pg_catalog.gs_get_explain(pid), pg_catalog.gs_stack(pid) FROM pg_catalog.pgxc_stat_activity WHERE pid = {pid};
\view slow_sql {top_n}	**slow_sql**: queries real-time slow SQL statements. **top_n**: specifies the number of rows to be queried.	SELECT pid, sessionid, pg_catalog.SUBSTRING(query, 0, 100), state, usename, pg_catalog.NOW() - query_start AS elp FROM pg_catalog.pgxc_stat_activity WHERE state != 'idle' AND usename NOT IN ('rdsAdmin', 'rdsMetric', 'rdsBackup', 'rdsRepl', 'dbmind_monitor_agent', 'dbmind_monitor') ORDER BY elp DESC LIMIT {top_n};
\view slow_trans {top_n}	**slow_trans**: queries real-time long transactions. **top_n**: specifies the number of rows to be queried.	SELECT pid, sessionid, pg_catalog.SUBSTRING(query, 0, 100), state, usename, pg_catalog.NOW() - xact_start AS elp FROM pg_catalog.pgxc_stat_activity WHERE state != 'idle' AND usename NOT IN ('rdsAdmin', 'rdsMetric', 'rdsBackup', 'rdsRepl', 'dbmind_monitor_agent', 'dbmind_monitor') ORDER BY elp DESC LIMIT {top_n};

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Simple O&M Command	Description	SQL Statement
\view wait_event	**wait_event** (without parameters): queries the current statement and wait event.	SELECT a.pid, a.sessionid, a.datname, a.usename, a.client_addr, a.xact_start, a.query_start, (pg_catalog.NOW() - a.query_start)::text "query_runtime", w.wait_status, w.wait_event, w.locktag, w.lockmode, w.block_sessionid, a.query FROM pg_catalog.pgxc_stat_activity a JOIN pg_catalog.pgxc_thread_wait_status w ON a.sessionid = w.sessionid WHERE a.pid <> pg_catalog.pg_backend_pid() AND a.state = 'active' AND a.client_addr IS NOT NULL ORDER BY query_runtime DESC;
\view wait_event {unique_sql_id} {start_time} {end_time}	**wait_event** (with three parameters): queries the historical wait events of a specified slow SQL statement. **unique_sql_id**: specifies the unique SQL ID of the slow SQL statement to be queried. **start_time**: specifies the start time. For example, '2025-01-01 00:00'. **end_time**: specifies the end time. For example, '2025-01-01 00:00'.	SELECT *, pg_catalog.statement_detail_decode(details, 'plaintext', true) FROM dbe_perf.statement_history WHERE unique_query_id = {unique_sql_id} AND start_time > {start_time} AND start_time < {end_time} ORDER BY db_time DESC;
\view thr_wait_event	**thr_wait_event** (without parameters): queries real-time overall wait events.	SELECT wait_status, wait_event, pg_catalog.COUNT(*) FROM pg_catalog.pgxc_thread_wait_status GROUP BY 1, 2 ORDER BY 3 DESC;
\view thr_wait_event {sessionid}	**thr_wait_event** (with one parameter): queries wait events of a specified statement. **sessionid**: specifies the session ID.	SELECT * FROM pg_catalog.pgxc_thread_wait_status WHERE sessionid = {sessionid};

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Simple O&M Command	Description	SQL Statement
\view gs_asp {start_time} {end_time}	**gs_asp**: queries the overall wait events in a specified period. **start_time**: specifies the start time. For example, '2025-01-01 00:00'. **end_time**: specifies the end time. For example, '2025-12-31 23:59'.	SELECT wait_status, event, pg_catalog.COUNT(*) FROM pg_catalog.gs_asp WHERE sample_time BETWEEN {start_time} AND {end_time} GROUP BY 1, 2 ORDER BY 3 DESC;
\view pg_stats {table_name} {att_name}	**pg_stats**: queries the distribution of statistics information in a specified table. **table_name**: specifies the table name. **att_name**: specifies the column name.	SELECT * FROM pg_catalog.pg_stats WHERE tablename = '{table_name}' AND attname = '{att_name}';
\view autovac {table_name}	**autovac**: queries the threshold of updating statistics information. **table_name**: specifies the table name.	SELECT * FROM pg_catalog.pg_autovac_status('{table_name}'::regclass);
\view thr_pool	**thr_pool**: queries the thread pool usage.	SELECT node_name, worker_info FROM dbe_perf.global_threadpool_status;
\view failed_proc {proc_name}	**failed_proc**: queries invalid views or stored procedures. **proc_name**: specifies the invalid function.	SELECT re.ev_class, cl.relname FROM pg_catalog.pg_rewrite re, pg_catalog.pg_class cl WHERE re.ev_action LIKE '%{proc_name}%' AND cl.oid = re.ev_class;

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Simple O&M Command	Description	SQL Statement
\view failed_proc_pkg	**failed_proc_pkg**: queries the package of the invalid stored procedure.	SELECT DISTINCT object_type, pg_catalog.pg_get_userbyid(creator) AS owner, CASE object_type WHEN 'P' THEN (SELECT DISTINCT pg_catalog.UPPER(pkgname) FROM pg_catalog.gs_package WHERE oid = (SELECT propackageid FROM pg_catalog.pg_proc WHERE oid = object_oid)) WHEN 'B' THEN (SELECT DISTINCT pg_catalog.UPPER(pkgname) FROM pg_catalog.gs_package WHERE oid = object_oid) WHEN 'S' THEN (SELECT DISTINCT pg_catalog.UPPER(pkgname) FROM pg_catalog.gs_package WHERE oid = object_oid) END package_name FROM pg_catalog.pg_object WHERE valid = 'f' AND object_type IN ('P', 'B', 'S');
\view failed_index	**failed_index**: queries unavailable indexes.	SELECT a.schemaname, a.relname, a.indexrelname, b.indisusable FROM pg_catalog.pg_stat_user_indexes a, pg_catalog.pg_index b WHERE a.indexrelid = b.indexrelid AND b.indisusable = 'f';
\view table_size	**table_size**: queries the sizes of all tables in the current database and outputs the sizes in descending order.	SELECT schemaname, relname, pg_catalog.pg_total_relation_size(pg_catalog.CONCAT(schemaname, '.', relname)) / 1024 / 1024 / 1024 "GB" FROM pg_catalog.pg_stat_user_tables ORDER BY 3 DESC;

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Simple O&M Command	Description	SQL Statement
\view dead_tup {top_n}	**dead_tup**: queries the number of dead tuples. **top_n**: specifies the number of rows to be queried.	SELECT relname, n_live_tup, n_dead_tup, (n_dead_tup / (n_live_tup + 1)) AS rating, last_vacuum, last_autovacuum, last_analyze, last_autoanalyze FROM pg_catalog.pg_stat_user_tables ORDER BY rating DESC, n_dead_tup DESC LIMIT {top_n};
\view table_size {table_name}	**table_size**: queries the size of a specified table. **table_name**: specifies the table name.	SELECT * FROM pg_catalog.pg_size_pretty(pg_catalog.pg_total_relation_size('{table_name}'::regclass));
\view login {start_time} {end_time} {ip} {port}	**login**: queries the login and logout information of the specified IP address and port. **start_time**: specifies the start time. For example, '2024-01-01 00:00:00'. **end_time**: specifies the end time. For example, '2025-12-31 23:59:59'. **ip**: specifies the IP address. **port**: specifies the port number.	SELECT *, pg_catalog.SUBSTRING(client_conninfo::varchar, pg_catalog.POSITION(client_conninfo::varchar, '@'::varchar) + 1, 15) AS ip FROM pg_catalog.pg_query_audit({start_time}, {end_time}) WHERE ip = '{ip}' AND remote_port = '{port}';
\view mem	**mem**: queries the current memory usage.	SELECT * FROM pg_catalog.pg_total_memory_detail;
\view thr_mem {top_n}	**thr_mem** (with one parameter): queries the memory context usage of all database threads. **top_n**: specifies the number of rows to be queried.	SELECT contextname, pg_catalog.SUM(totalsize) / 1024 / 1024 "totalsize", pg_catalog.SUM(freesize) / 1024 / 1024 "freesize", pg_catalog.COUNT(*) "sum" FROM pg_catalog.pv_thread_memory_context GROUP BY contextname ORDER BY sum DESC LIMIT {top_n};

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Simple O&M Command	Description	SQL Statement
\view thr_mem {context_name} {top_n}	**thr_mem** (with two parameters): queries the thread ID of the thread that uses the most memory. **context_name**: specifies the name of the memory context. **top_n**: specifies the number of rows to be queried.	SELECT * FROM pg_catalog.pv_thread_memory_detail() WHERE contextname = '{context_name}' ORDER BY totalsize DESC LIMIT {top_n};
\view sess_mem {top_n}	**sess_mem**: queries the memory context usage of all sessions in the database. **top_n**: specifies the number of rows to be queried.	SELECT contextname, pg_catalog.SUM(totalsize) / 1024 / 1024 "totalsize", pg_catalog.SUM(freesize) / 1024 / 1024 "freesize", pg_catalog.COUNT(*) "sum" FROM pg_catalog.pv_session_memory_context GROUP BY contextname ORDER BY sum DESC LIMIT {top_n};
\view thr_mem_detail {tid} {context_name}	**thr_mem_detail**: queries the memory usage of a specified thread. **tid**: specifies the thread ID. **context_name**: specifies the name of the memory context.	SELECT * FROM pg_catalog.gs_get_thread_memctx_detail({tid}, '{context_name}');
\view sess_mem_tp {context_name}	**sess_mem_tp**: queries the memory usage of a specified memory context. **context_name**: specifies the name of the memory context.	SELECT * FROM pg_catalog.gs_get_session_memctx_detail('{context_name}');
\view shared_mem {top_n}	**shared_mem**: queries the usage of the global shared memory context in the database. **top_n**: specifies the number of rows to be queried.	SELECT contextname, pg_catalog.SUM(totalsize) / 1024 / 1024 "totalsize MB", pg_catalog.SUM(freesize) / 1024 / 1024 "freesize MB", pg_catalog.COUNT(*) "sum" FROM pg_catalog.pg_shared_memory_detail GROUP BY contextname ORDER BY sum DESC LIMIT {top_n};

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Simple O&M Command	Description	SQL Statement
\view sess_mem_detail {top_n}	**sess_mem_detail**: queries the usage of a session-level memory context. **top_n**: specifies the number of rows to be queried.	SELECT contextname, pg_catalog.SUM(totalsize) / 1024 / 1024 "totalsize MB", pg_catalog.COUNT(1) "count" FROM pg_catalog.pv_session_memory_detail() GROUP BY contextname ORDER BY 2 DESC LIMIT {top_n};
\view sess_mem_detail_tp {top_n}	**sess_mem_detail_tp**: queries the usage of a session-level memory context (valid only when the thread pool mode is enabled). **top_n**: specifies the number of rows to be queried.	SELECT sessid, pg_catalog.SUM(totalsize) / 1024 / 1024 "totalsize MB", pg_catalog.COUNT(1) "count" FROM pg_catalog.pv_session_memory_context GROUP BY sessid ORDER BY 2 DESC LIMIT {top_n};
\view thr_mem_pv {top_n}	**thr_mem_pv**: queries the usage of a thread-level memory context. **top_n**: specifies the number of rows to be queried.	SELECT contextname, pg_catalog.SUM(totalsize) / 1024 / 1024 "totalsize", pg_catalog.SUM(freesize) / 1024 / 1024 "freesize", pg_catalog.COUNT(1) "count" FROM pg_catalog.pv_thread_memory_detail() GROUP BY 1 ORDER BY 2 DESC LIMIT {top_n};
\view total_mem {top_n}	**total_mem**: queries the memory usage of the current database node. **top_n**: specifies the number of rows to be queried.	SELECT * FROM pg_catalog.pg_total_memory_detail LIMIT {top_n};
\view avg_cpu {top_n}	**avg_cpu**: queries the average CPU time of SQL statements and outputs the query result in descending order. **top_n**: specifies the number of rows to be queried.	SELECT unique_sql_id, cpu_time / (n_calls) "average_cpu_time" FROM dbe_perf.statement WHERE n_calls != 0 ORDER BY 2 DESC LIMIT {top_n};

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Simple O&M Command	Description	SQL Statement
\om create_abort_sql_patch {patch_name} {unique_sql_id}	**create_abort_sql_patch**: creates an abort SQL patch. **patch_name**: specifies the SQL patch name. **unique_sql_id**: specifies the unique SQL ID.	SELECT * FROM dbe_sql_util.create_abort_sql_patch('{patch_name}', {unique_sql_id});
\om kill_sess_thr {username} {database} {context_name}	**kill_sess_thr**: kills SQL statements that occupy a large amount of memory based on the thread memory usage. **username**: specifies the username. **database**: specifies the database name. **context_name**: specifies the name of the memory context.	EXECUTE DIRECT ON COORDINATORS 'SELECT pg_catalog.pg_terminate_session(pid, sessionid), pid, sessionid FROM pg_catalog.pgxc_stat_activity WHERE usename = ''{username}'' AND datname != ''postgres'' AND datname = ''{database}'' AND pid IN (SELECT tid FROM pg_catalog.pv_thread_memory_context WHERE contextname = ''{context_name}'') AND state = ''idle'';';
\om kill_sess_sess {username} {database} {context_name}	**kill_sess_sess**: kills SQL statements that occupy a large amount of memory based on session memory usage. **username**: specifies the username. **database**: specifies the database name. **context_name**: specifies the name of the memory context.	EXECUTE DIRECT ON COORDINATORS 'SELECT pg_catalog.pg_terminate_session(pid, sessionid), pid, sessionid FROM pg_catalog.pgxc_stat_activity WHERE usename = ''{usename}'' AND datname != ''postgres'' AND datname = ''{database}'' AND pid IN (SELECT threadid FROM pg_catalog.pv_session_memory_context WHERE contextname = ''{context_name}'');';

Examples

Example 1

Obtain the help information.

```
-- Connect to a database. (Replace the database name and port number as required.)
gsql -d postgres -p 8000
gsql ((GaussDB Kernel XXX.X.XXX build f521c606) compiled at 2021-09-16 14:55:22 last mr 6385 release)
Non-SSL connection (SSL connection is recommended when requiring high-security)
Type "help" for help.

-- View all SQL statements supported by GaussDB.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=#  \h
Available help:
  ABORT
  ALTER AGGREGATE
 ... ...

-- View the parameters of the DROP TABLE syntax.
gaussdb=#  \h DROP TABLE
Command:     DROP TABLE
Description: remove a table
Syntax:
DROP TABLE [ IF EXISTS ]
    { [schema.]table_name } [, ...] [ CASCADE | RESTRICT ] [ PURGE ];

-- View meta-commands supported by gsql.
gaussdb=#  \?
General
  \copyright             show GaussDB Kernel usage and distribution terms
  \g [FILE] or ;         execute query (and send results to file or |pipe)
... ...
```

Example 2

Use a meta-command.

```
-- Connect to a database. (Replace the database name and port number as required.)
gsql -d postgres -p 8000
gsql ((GaussDB Kernel XXX.X.XXX build f521c606) compiled at 2021-09-16 14:55:22 last mr 6385 release)
Non-SSL connection (SSL connection is recommended when requiring high-security)
Type "help" for help.

-- Create a table.
gaussdb=# CREATE SCHEMA HR;
CREATE SCHEMA
gaussdb=# CREATE TABLE HR.areaS(area_ID NUMBER,area_NAME VARCHAR2(25));
NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'area_id' as the distribution column by default.
HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
CREATE TABLE

-- View the definition of a table.
gaussdb=#  \d HR.areaS
              Table "hr.areas"
  Column   |         Type          | Modifiers
-----------+-----------------------+-----------
 area_id   | numeric               |
 area_name | character varying(25) |


-- Insert four rows of data into HR.areaS.
gaussdb=# INSERT INTO HR.areaS (area_ID, area_NAME) VALUES (1, 'Europe');
INSERT 0 1
gaussdb=# INSERT INTO HR.areaS (area_ID, area_NAME) VALUES (2, 'Americas');
INSERT 0 1
gaussdb=# INSERT INTO HR.areaS (area_ID, area_NAME) VALUES (3, 'Asia');
INSERT 0 1
gaussdb=# INSERT INTO HR.areaS (area_ID, area_NAME) VALUES (4, 'Middle East and Africa');
INSERT 0 1

-- View a table.
gaussdb=# SELECT * FROM HR.areaS;
area_id |       area_name
---------+------------------------
       4 | Middle East and Africa
       1 | Europe
       2 | Americas
       3 | Asia
(4 rows)

-- Use the \pset command to display a table in different ways.
gaussdb=# \pset border 2
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
Border style is 2.

gaussdb=# SELECT * FROM HR.areaS;
+---------+------------------------+
| area_id |       area_name        |
+---------+------------------------+
|       4 | Middle East and Africa |
|       1 | Europe                 |
|       2 | Americas               |
|       3 | Asia                   |
+---------+------------------------+
(4 rows)

gaussdb=# \pset border 0
Border style is 0.

gaussdb=# SELECT * FROM HR.areaS;
area_id       area_name
------- ----------------------
      4 Middle East and Africa
      1 Europe
      2 Americas
      3 Asia
(4 rows)

-- Use a meta-command.
gaussdb=# \a \t \x
Output format is unaligned.
Showing only tuples.
Expanded display is on.

gaussdb=# SELECT * FROM HR.areaS;
area_id|4
area_name|Middle East and Africa

area_id|1
area_name|Europe

area_id|2
area_name|Americas

area_id|3
area_name|Asia

gaussdb=# \a \t \x
Output format is aligned.
Tuples only is off.
Expanded display is off.

gaussdb=# SELECT * FROM HR.areaS;
area_id       area_name
------- ----------------------
      4 Middle East and Africa
      1 Europe
      2 Americas
      3 Asia
(4 rows)

gaussdb=# DROP TABLE HR.areaS;
DROP TABLE

gaussdb=# DROP SCHEMA HR;
DROP SCHEMA
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

4.2.1.3 FAQ

Low Connection Performance

- Problem: **log_hostname** is enabled, but DNS is incorrect.

  Solution:

  Connect to the database, and run **show log_hostname** to check whether **log_hostname** is enabled in the database.

  If it is enabled, the database kernel will use DNS to check the name of the host where the client is deployed. If the host where the CN is located is configured with an incorrect or unreachable DNS, the database connection will take a long time to set up. For details, see GUC parameter **log_hostname**.

- Problem: The database kernel runs the initialization statement slowly.

  Solution:

  It is difficult to locate faults in this scenario. Try using the **strace** Linux command.

  ```
  strace gsql -U MyUserName -d gaussdb -h 127.0.0.1 -p 23508 -r -c '\q'
  Password for MyUserName:
  ```

  The database connection process will be printed on the screen. For example, if the following operations are suspended for a long time, the database executes the **SELECT VERSION()** statement slowly.

  ```
  sendto(3, "Q\0\0\0\25SELECT VERSION()\0", 22, MSG_NOSIGNAL, NULL, 0) = 22
  poll([{fd=3, events=POLLIN|POLLERR}], 1, -1) = 1 ([{fd=3, revents=POLLIN}])
  ```

  After the database is connected, you can run the **explain performance select version()** statement to find the reason why the initialization statement was run slowly. For more information, see "Introduction to the SQL Execution Plan" in *SQL Optimization Guide*.

  An uncommon scenario is that the disk of the machine where the database CN resides is full or faulty, affecting queries and leading to user authentication failures. As a result, the connection process is suspended. To solve this problem, clear the data disk space of the database CN.

- Problem: The TCP connection is set up slowly.

  Solution:

  Run **strace** to check whether the initialization statement is run slowly. If the following information is displayed for a long time:

  ```
  connect(3, {sa_family=AF_FILE, path="/home/test/tmp/gaussdb_llt1/.s.PGSQL.61052"}, 110) = 0
  ```

  Or,

  ```
  connect(3, {sa_family=AF_INET, sin_port=htons(61052), sin_addr=inet_addr("127.0.0.1")}, 16) = -1
  EINPROGRESS (Operation now in progress)
  ```

  It indicates that the physical connection between the client and the database is set up slowly. In this case, check whether the network is unstable or has high throughput.

- Problem: The connection is slow due to full resource load.

  Solution:

  When the CPU, memory, or I/O usage is close to 100%, the gsql connection is slow.

  – Run the **top** command to check the CPU usage. Run the **free** command to check the memory usage. Run the **iostat** command to check the I/O

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

load. You can also check the monitor logs in the CM Agent and the monitoring records on the database O&M platform.

- For peak load scenarios caused by a large number of slow queries in a short period of time, you can use the port specified by (*Port number of the database server* + 1) to query the pg_stat_activity view. For slow queries, you can use the system function pg_terminate_backend to kill sessions.
- If service overloading exists for a long time (that is, there is no obvious slow query, or new queries still become slow after slow queries are killed), reduce the service load and increase database resources.

Problems in Setting Up Connections

- Problem: The following error message is displayed: "gsql: could not connect to server: No route to host."

  Solution:

  This problem occurs because an unreachable address or port is specified. Check whether the values of **-h** and **-p** parameters are correct.

- Problem: The following error message is displayed: "gsql: FATAL: Invalid username/password,login denied."

  Solution:

  This problem occurs generally because an incorrect username or password was entered. Contact the database administrator to check whether the username and password are correct.

- Problem: The following error message is displayed: "gsql: FATAL: Forbid remote connection with trust method!"

  Solution:

  For security purposes, remote database login in trust mode is forbidden. In this case, you need to modify the connection authentication information in the **gs_hba.conf** file. Contact the administrator.

  NOTE

  Do not modify the configurations of database cluster hosts in the **gs_hba.conf** file. Otherwise, the database may become faulty. It is recommended that service applications be deployed outside the database cluster.

- Problem: The database can be connected from the host where the CN is located by adding **-h 127.0.0.1**, but the connection fails if **-h 127.0.0.1** is removed.

  Solution:

  Run the SQL statement **show unix_socket_directory** to check whether the **unix socket directory** used by the database CN is the same as that specified by the environment variable *$PGHOST* in the **shell** directory. If they are different, set *$PGHOST* to the directory specified by **unix_socket_directory**.

  For more details about **unix_socket_directory**.

- Problem: The following error message is displayed: "The "libpq.so" loaded mismatch the version of gsql, please check it."

  Solution:

  This problem occurs because the version of **libpq.so** used in the environment does not match that of **gsql**. Run the **ldd gsql** command to check the version

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

of the loaded **libpq.so**, and then load correct **libpq.so** by modifying the environment variable **LD_LIBRARY_PATH**.

- Problem: The following error message is displayed: "gsql: symbol lookup error: xxx/gsql: undefined symbol: libpqVersionString."

  Solution:

  This problem occurs because the version of **libpq.so** used in the environment does not match that of gsql (or the PostgreSQL **libpq.so** exists in the environment). Run the **ldd gsql** command to check the version of the loaded **libpq.so**, and then load correct **libpq.so** by modifying the environment variable *LD_LIBRARY_PATH*.

- Problem: The following error message is displayed: "gsql: connect to server failed: Connection timed out."

  Is the server running on host "xx.xxx.xxx.xxx" and accepting TCP/IP connections on port xxxx?

  Solution:

  This problem is caused by network connection faults. Check the network connection between the client and the database server. If you cannot ping from the client to the database server, the network connection is abnormal. Contact network management personnel for troubleshooting.

  ```
  ping -c 4 10.10.10.1
  PING 10.10.10.1 (10.10.10.1) 56(84) bytes of data.
  From 10.10.10.1: icmp_seq=2 Destination Host Unreachable
  From 10.10.10.1 icmp_seq=2 Destination Host Unreachable
  From 10.10.10.1 icmp_seq=3 Destination Host Unreachable
  From 10.10.10.1 icmp_seq=4 Destination Host Unreachable
  --- 10.10.10.1 ping statistics ---
  4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 2999ms
  ```

- Problem: The following error message is displayed: "gsql: FATAL: permission denied for database "gaussdb"

  DETAIL: User does not have CONNECT privilege."

  Solution:

  This problem occurs because the user does not have the permission to access the database. To solve this problem, perform the following steps:

  a. Connect to the database as a database administrator.

     ```
     gsql -d gaussdb -U dbadmin -p 8000
     ```

  b. Grant the user with the permission to access the database.

     ```
     GRANT CONNECT ON DATABASE gaussdb TO user1;
     ```

  In addition, many common misoperations may cause users to fail to connect to the database, for example, entering an incorrect database name, username, or password. In this case, the client tool will display the corresponding error messages.

  ```
  gsql -d gaussdb -p 8000
  gsql: FATAL:  database "gaussdb" does not exist

  gsql -d gaussdb -U user1 -p 8000
  Password for user user1:
  gsql: FATAL:  Invalid username/password,login denied.
  ```

- Problem: The following error message is displayed: "gsql: FATAL: sorry, too many clients already, active/non-active: 197/3."

  Solution:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

This problem occurs because the number of system connections exceeds the allowed maximum. Contact the DBA database administrator to release unnecessary sessions.

You can check the number of connections of user sessions as described in **Table 4-25**.

You can view the session status in the **PG_STAT_ACTIVITY** view. To release unnecessary sessions, use the **pg_terminate_backend** function.

```
SELECT datid,pid,state FROM pg_stat_activity;
 datid |       pid       | state
-------+-----------------+--------
 13205 | 139834762094352 | active
 13205 | 139834759993104 | idle
(2 rows)
```

The value of **pid** is the thread ID of the session. Terminate the session using its thread ID.

```
SELECT PG_TERMINATE_BACKEND(139834759993104);
```

If a command output similar to the following is displayed, the session is successfully terminated:

```
PG_TERMINATE_BACKEND
----------------------
 t
(1 row)
```

Table 4-25 Viewing the number of session connections

Description	Command

View the maximum number of sessions connected to a specific user.	Run the following command to view the upper limit of **user1**'s connections. **-1** indicates that no upper limit is set for **USER1**'s session connections.
	`gaussdb=# SELECT ROLNAME,ROLCONNLIMIT FROM PG_ROLES WHERE ROLNAME='user1';`
	` rolname | rolconnlimit`
	`---------+--------------`
	` user1 |  -1`
	`(1 row)`

View the number of session connections that have been used by a specified user.	Run the following command to view the number of connections that have been used by **user1**. **1** indicates the number of connections that have been used by **user1**.
	`gaussdb=# SELECT COUNT(*) FROM dv_sessions WHERE USERNAME='user1';`
	` count`
	`-------`
	`  1`
	`(1 row)`

View the maximum number of session connections of a specific database.	Run the following command to view the upper limit of **gaussdb**'s session connections. **-1** indicates that no upper limit is set for **gaussdb**'s session connections.
	`gaussdb=# SELECT DATNAME,DATCONNLIMIT FROM PG_DATABASE WHERE DATNAME='gaussdb';`
	` datname | datconnlimit`
	`----------+--------------`
	` gaussdb |  -1`
	`(1 row)`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Description	Command
View the number of session connections that have been used by a specified database.	Run the following command to view the number of session connections that have been used by **gaussdb**. **1** indicates the number of session connections that have been used by **gaussdb**. gaussdb=# SELECT COUNT(*) FROM PG_STAT_ACTIVITY WHERE DATNAME='gaussdb'; count ------- 1 (1 row)
View the number of session connections that have been used by all users.	Run the following command to view the number of session connections that have been used by all users: gaussdb=# SELECT COUNT(*) FROM dv_sessions; count ------- 10 (1 row)

- Problem: The following error message is displayed: "gsql: wait xxx.xxx.xxx.xxx:xxxx timeout expired."

  Solution:

  When gsql initiates a connection request to the database, a 5-minute timeout period is used. If the database cannot correctly authenticate the client request and client identity within this period, gsql will exit the connection process for the current session, and will report the above error.

  Generally, this problem is caused by the incorrect host and port (that is, the *xxx* part in the error information) specified by the **-h** and **-p** parameters. As a result, the communication fails. Occasionally, this problem is caused by network faults. To resolve this problem, check whether the host name and port number of the database are correct.

- Problem: The following error message is displayed: "gsql: could not receive data from server: Connection reset by peer."

  Solution:

  Check whether CN logs contain information similar to "FATAL: cipher file "/data/coordinator/server.key.cipher" has group or world access." This error is usually caused by incorrect tampering with the permissions for data directories or some key files. For details about how to correct the permissions, see related permissions for files on other normal instances.

- Problem: The following error message is displayed: "gsql: FATAL: GSS authentication method is not allowed because XXXX user password is not disabled."

  Solution:

  In **gs_hba.conf** of the target CN, the authentication mode is set to **gss** for authenticating the IP address of the current client. However, this authentication algorithm cannot authenticate clients. Change the authentication algorithm to **sha256** and try again. For details, contact the administrator..

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- Do not modify the configurations of database cluster hosts in the **gs_hba.conf** file. Otherwise, the database may become faulty.
- It is recommended that service applications be deployed outside the database cluster.

Other Faults

- Problem: A core dump or abnormal exit occurs due to a bus error.

  Solution:

  Generally, this problem is caused by changes to the shared dynamic library (.so file in Linux) loaded during process running. Alternatively, if the process binary file changes, the execution code for the OS to load machines or the entry for loading a dependency library will change accordingly. In this case, the OS terminates the process for protection purposes, generating a core dump file. To resolve this problem, please try again. In addition, do not run service programs in a cluster during O&M operations, such as an upgrade, preventing such a problem caused by file replacement during the upgrade.

  NOTE

  Possibly, a stack of the core dump file contains dl_main and its function calling. The file is used by the OS to initialize a process and load the shared dynamic library. If the process has been initialized but the shared dynamic library has not been loaded, the process cannot be considered completely started.

4.3 Data Import and Export Tools

4.3.1 Overview

Table 4-26 Overview

Tool	Description
COPY	Syntax for importing and exporting database data in batches. COPY can be used to import and export table-level data and export query result sets. For more information, see **COPY**.
\COPY	A meta-command used on the gsql client for importing and exporting table-level data in batches. It is based on the COPY ... FROM STDIN/STDOUT usage and encapsulates packets through libpq so as to store data on the gsql client. For details, see **Table 4-14**.
gs_dump	A logical database backup tool. To export information about a single database, you can export the definitions and data of all or some objects (such as schemas, tables, and views) in a database. For details, see **gs_dump for Exporting Database Information**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Tool	Description
gs_dumpall	A logical database backup tool. You can export object definitions and data of all databases in an instance, including data of the default database, data of user-defined databases, and global objects of all databases. For details, see **gs_dumpall for Exporting All Database Information**.
gs_restore	A logical database restoration tool. It imports files exported using gs_dump. For details, see **gs_restore for Importing Data**.

Instance-Level Import and Export

gs_dumpall is recommended for users who want to logically back up an entire database instance. gs_dumpall can export data of all databases in an instance, including the data of the default database, user-defined database, and global objects shared with all databases.

NOTICE

- gs_dumpall does not back up or restore GUC parameters.
- gs_dumpall supports only plain-text format export. Therefore, only gsql can be used to restore a file exported using gs_dumpall.
- After restoration, you are advised to run **ANALYZE** on each database to provide useful statistics for the optimizer.

During backup, you are advised to run the following command as the initial user or a user with the SYSADMIN permission:

```
nohup gs_dumpall -U root -W ******** -f /data/backup/bkp2.sql -p 8000 > /data/backup/dumpall_backup.log &
```

During restoration, you are advised to run the following command as the initial user or a user with the SYSADMIN permission:

```
nohup gsql -d postgres -p 8000 -U root -W ******** -f /data/backup/bkp2.sql -a > /data/backup/dumpall_restore.log &
```

Database-Level Import and Export

gs_dump is recommended for users who want to back up a single database logically. gs_dump supports four archive formats. The following table lists the application scenarios of different archive formats. You can select an archive format as needed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 4-27 Formats of exported files

Format	Value of -F	Description	Suggestion	Import Tool
Plain-text	p	A plain-text script file containing SQL statements and commands. The commands can be executed on gsql, a command line terminal, to rebuild database objects and load table data.	For a small-sized database or the exported SQL file needs to be modified, the plain-text format is recommended.	Before using **gsql** to restore database objects, you can use a text editor to edit the exported plain-text file as needed.
Custom	c	A binary file that allows the restoration of all or selected database objects from an exported file.	For medium- or large-sized databases, the backup result needs to be exported to a single file. In this case, the custom format is recommended.	You can use **gs_restore** to import database objects from a custom-, directory-, or tar-format archive.
Directory	d	A directory containing directory files of database objects and the data files of tables and BLOBs.	For medium- or large-sized databases, database objects and data files need to be stored and exported in different directories. In this case, the directory format is recommended.	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Format	Value of -F	Description	Suggestion	Import Tool
.tar archive	t	A tar-format archive that allows the restoration of all or selected database objects from an exported file. The .tar file cannot be further compressed and has an 8-GB limitation on the size of each single file.	For a small-sized database, you need to export the archive result and pack it. In this case, the .tar format is recommended.	

NOTICE

- gs_dump does not back up or restore global objects (roles, tablespaces, and corresponding permissions) shared with all databases. Therefore, ensure that global objects have been created on the target database or new instance before the restoration.
- gs_dump and gs_restore do not support import and export across database compatibility modes. Ensure that the database compatibility mode and compatibility configuration parameters of the source database are the same as those of the target database.
- Do not modify the files and contents exported using the **-F c**, **-F d**, or **-F t** format. Otherwise, the restoration may fail. If you need to modify or replace the file exported using the **-F p** format, edit them with caution.
- After restoration, you are advised to run **ANALYZE** on the database to provide useful statistics for the optimizer.

You are advised to run the following command as the initial user or a user with the SYSADMIN permission to back up data. The source database is **my_database**, and the exported data contains object definitions and data.

```
-- Plain-text
nohup gs_dump my_database -U root -W ******** -p 8000 -F p -f /data/backup/my_database_backup.sql > /
data/backup/my_database_backup.log &
-- Custom
nohup gs_dump my_database -U root -W ******** -p 8000 -F c -f /data/backup/my_database_backup.dmp > /
data/backup/my_database_backup.log &
-- Directory
nohup gs_dump my_database -U root -W ******** -p 8000 -F d -f /data/backup/my_database_backup > /data/
backup/my_database_backup.log &
-- .tar archive
nohup gs_dump my_database -U root -W ******** -p 8000 -F t -f /data/backup/my_database_backup.tar > /
data/backup/my_database_backup.log &
```

Before restoration, you need to create a target database that has the same attributes as the source database and does not contain any data.

```
-- Run the following gsql meta-command to view the database attribute information:
\l+
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
-- Create a target database based on the queried attribute information.
create database my_database2 encoding='xxxxx' LC_COLLATE='xxxxx' LC_CTYPE ='xxxxx' TEMPLATE=xxx
DBCOMPATIBILITY 'xxx';
```

Run the following command as the initial user or a user with the SYSADMIN permission to restore the database:

```
-- Plain-text
nohup gsql -d my_database2 -p 8000 -U root -W ******** -f /data/backup/my_database_backup.sql -a > /
data/backup/my_database_restore.log &
-- Custom
nohup gs_restore /data/backup/my_database_backup.dmp -d my_database2 -p 8000 -U root -W ******** -F c
-v > /data/backup/my_database_restore.log &
-- Directory
nohup gs_restore /data/backup/my_database_backup -d my_database2 -p 8000 -U root -W ******** -F d -v
> /data/backup/my_database_restore.log &
-- .tar archive
nohup gs_restore /data/backup/my_database_backup.tar -d my_database2 -p 8000 -U root -W ******** -F t -
v > /data/backup/my_database_restore.log &
```

Schema-Level Import and Export

To back up user data by schema, you are advised to use the gs_dump tool with the **-n** parameter. The **-n** parameter can be used repeatedly to back up multiple schemas.

You are advised to run the following command as the initial user or a user with the SYSADMIN permission to back up data. The source database is **my_database**, and the exported data contains object definitions and data.

```
-- Plain-text
nohup gs_dump my_database -U root -W ******** -p 8000 -F p -f /data/backup/my_schema_backup.sql -n
my_schema > /data/backup/my_schema_backup.log &
-- Custom
nohup gs_dump my_database -U root -W ******** -p 8000 -F c -f /data/backup/my_schema_backup.dmp -n
my_schema > /data/backup/my_schema_backup.log &
-- Directory
nohup gs_dump my_database -U root -W ******** -p 8000 -F d -f /data/backup/my_schema_backup -n
my_schema > /data/backup/my_schema_backup.log &
-- .tar archive
nohup gs_dump my_database -U root -W ******** -p 8000 -F t -f /data/backup/my_schema_backup.tar -n
my_schema > /data/backup/my_schema_backup.log &
```

Before restoration, you need to create a target database that has the same attributes as the source database and does not contain the target schema.

```
-- Run the following gsql meta-command to view the database attribute information:
\l+
-- Create a target database based on the queried attribute information.
create database my_database2 encoding='xxxxx' LC_COLLATE='xxxxx' LC_CTYPE ='xxxxx' TEMPLATE=xxx
DBCOMPATIBILITY 'xxx';
```

Run the following command as the initial user or a user with the SYSADMIN permission to restore the database:

```
-- Plain-text
nohup gsql -d my_database2 -p 8000 -U root -W ******** -f /data/backup/my_schema_backup.sql -a > /data/
backup/my_schema_restore.log &
-- Custom
nohup gs_restore /data/backup/my_database_backup.dmp -d my_database2 -p 8000 -U root -W ******** -F c
-v -n my_schema > /data/backup/my_schema_restore.log &
-- Directory
nohup gs_restore /data/backup/my_database_backup -d my_database2 -p 8000 -U root -W ******** -F d -v -
n my_schema > /data/backup/my_schema_restore.log &
-- .tar archive
nohup gs_restore /data/backup/my_database_backup.tar -d my_database2 -p 8000 -U root -W ******** -F t -
v -n my_schema > /data/backup/my_schema_restore.log &
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table-Level Import and Export

Many tools are available for table-level import and export. You can select a proper tool based on the scenarios.

1. You need to export the definition and data of a single table to the same file. You are advised to use the plain-text archive exported by gs_dump together with the **-t** parameter. The **-t** parameter can be used repeatedly to back up multiple tables.

   You are advised to run the following command as the initial user or a user with the SYSADMIN permission to back up data. The source database is **my_database**, and the target table is **my_table** in **my_schema**.

   ```
   nohup gs_dump my_database -U root -W ******** -p 8000 -F p -f /data/backup/my_table_backup.sql -t my_schema.my_table > /data/backup/my_table_backup.log &
   ```

   Before restoration, create a target database with the same attributes as the source database, and ensure that the target schema exists in the database and no target table exists. Then, run the following command as the initial user or a user with the SYSADMIN permission to restore the database:

   ```
   nohup gsql -d my_database2 -p 8000 -U root -W ******** -f /data/backup/my_table_backup.sql -a > /data/backup/my_table_restore.log &
   ```

2. You only need to export the definition of a single table and do not need the data in the table.

   You are advised to use the plain-text archive exported by the gs_dump tool together with the **-s** parameter. The command is as follows:

   ```
   nohup gs_dump my_database -U root -W ******** -p 8000 -F p -f /data/backup/my_table_backup.sql -t my_schema.my_table -s > /data/backup/my_table_backup.log &
   ```

   Before restoration, create a target database with the same attributes as the source database, and ensure that the target schema exists in the database and no target table exists. Then, run the following command as the initial user or a user with the SYSADMIN permission to restore the database:

   ```
   nohup gsql -d my_database2 -p 8000 -U root -W ******** -f /data/backup/my_table_backup.sql -a > /data/backup/my_table_restore.log &
   ```

3. You need to export the data of a single table and do not need the table definition.

   a. To export data to a database node, you are advised to use the COPY command.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- In most scenarios, for database security purposes, when **enable_copy_server_files** is disabled, only the initial user can run the **COPY FROM FILENAME** or **COPY TO FILENAME** command. When **enable_copy_server_files** is enabled, users with the SYSADMIN permission or users who inherit the permissions of the built-in role gs_role_copy_files can run the commands. By default, **COPY FROM FILENAME** or **COPY TO FILENAME** cannot be executed on database configuration files, key files, certificate files, or audit logs to prevent unauthorized users from viewing or modifying sensitive files. When **enable_copy_server_files** is enabled, the system administrators can configure the GUC parameter **safe_data_path** to set the whitelist of paths that can be used by common users for import and export. If this GUC parameter is not enabled (default), the paths used by common users are not intercepted.
- If the database is deployed in a sandbox, the file specified by using COPY is in the sandbox.

You are advised to run the following command to export data to a standard CSV file on the corresponding database node:

```
COPY my_schema.my_table TO '/data/backup/my_table.csv' CSV;
```

Then run the following command to import data to the target table:

```
COPY my_schema.my_table2 FROM '/data/backup/my_table.csv' CSV;
```

b. To export data to the client, you are advised to use the \COPY meta-command.

Run the following command to export data to a standard CSV file on the client:

```
\COPY my_schema.my_table TO '/data/backup/my_table.csv' CSV;
```

Then run the following command to import data to the target table:

```
\COPY my_schema.my_table2 FROM '/data/backup/my_table.csv' CSV;
```

4. You need a custom data format for data import and export, for example, the delimiter and newline character.

   COPY or \COPY is recommended.

4.3.2 gs_dump for Exporting Database Information

Description

gs_dump, provided by GaussDB, is used to export database information. You can export a database or its objects (such as schemas, tables, and views). The database can be the default **postgres** database or a user-specified database.

- gs_dump supports SSL-encrypted communication.
- When gs_dump is used to export data, other users can still access (read or write) the database cluster.
- gs_dump can export complete, consistent data. For example, if gs_dump is started to export database A at T1, data of the database at that time point will be exported, and modifications on the database after that time point will not be exported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- gs_dump can export database information to a plain-text SQL script file or archive file.
  - Plain-text SQL script: It contains the SQL statements required to restore the database. You can use **gsql for Connecting to a Database** to execute the SQL script. With only a little modification, the SQL script can rebuild a database on other hosts or database products.
  - Archive file: It contains data required to restore the database. It can be a tar-, directory-, or custom-format archive. For details, see **Table 4-28**. The export result must be used with **gs_restore for Importing Data** to restore the database. The system allows users to select or even to sort the content to be imported.
- gs_dump supports four export file formats, which are specified by **[-F, --format=]**. For details, see **Table 4-28**.

Table 4-28 Formats of exported files

Format	Value of -F	Description	Suggestion	Import Tool
Plain-text	p	A plain-text script file containing SQL statements and commands. The commands can be executed on gsql, a command line terminal, to rebuild database objects and load table data.	For a small-sized database or the exported SQL file needs to be modified, the plain-text format is recommended.	Before using **gsql for Connecting to a Database** to restore database objects, you can use a text editor to edit the exported plain-text file as required.
Custom	c	A binary file that allows the restoration of all or selected database objects from an exported file.	You are advised to use custom-format archive files for medium or large database.	Objects and data of the three archive formats can only be imported using the **4.7.5-gs_restore** tool. You can also use gs_restore to select the database objects to be imported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Format	Value of -F	Description	Suggestion	Import Tool
Directory	d	A directory containing directory files of database objects and the data files of tables and BLOBs.	Database objects and data files need to be stored and exported in different directories. Therefore, the directory format is recommended.	
.tar archive	t	A tar-format archive that allows the restoration of all or selected database objects from an exported file. The tar file cannot be further compressed and has an 8-GB limitation on the size of each single file.	You need to export the archive result and pack it. The tar format is recommended.	

NOTE

To reduce the size of an exported file, you can use the gs_dump tool to compress it to a directory archive file or custom-format file. When a directory archive or custom-format archive is generated, medium compression is used by default. Archived exported files cannot be compressed using gs_dump.

Precautions

- Before using gs_dump, ensure that the gs_dump version is consistent with the database version. gs_dump of a later version may not be fully compatible with kernel data of an earlier version.
- gs_dump is not suitable for scenarios where there are too many objects (such as tables, views, and indexes) in the database. If the number of objects in the database exceeds 100,000 or the dependency between objects is too complex, the export using gs_dump takes a long time.
- Do not modify the files and contents exported using the **-F c**, **-F d**, or **-F t** format. Otherwise, the restoration may fail. For files exported using the **-F p** format, you can edit the exported files with caution if necessary.
- To ensure data consistency and integrity, gs_dump sets an access shared lock on the table to be dumped by default. If another conflicting lock has been set

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

for the table in other transactions, gs_dump locks the table after that lock is released. If the table cannot be locked within the specified time, the dump fails. You can customize the timeout interval to wait for lock release by specifying the **--lock-wait-timeout** parameter. For details about lock conflicts, see **LOCK**.

- Stored procedures and functions cannot be exported in encrypted mode.
- For materialized views, this tool supports only definition export. After importing materialized views, you need to manually run the REFRESH statement to restore data.
- gs_dump does not export materialized view-related tables whose names start with matviewmap_ or mlog_, tables with the **internal_mask** option, or tables in the recycle bin.
- For temporary objects, this tool can export only global temporary tables.
- This tool cannot be used on standby nodes.
- The distribution information in the system catalog on the DN is incomplete. Therefore, when gs_dump is used on the DN, the table distribution information is not dumped.
- When gs_dump is used to export partitioned indexes, the attributes of some index partitions cannot be exported, for example, the unusable status of index partitions. You can query the PG_PARTITION system catalog or ADM_IND_PARTITIONS view to obtain the attributes of an index partition. You can run the ALTER INDEX statement to manually set the attributes of an index partition.
- For scheduled tasks, this tool can export only non-periodic scheduled tasks created using advanced packages from a MySQL-compatible database.
- gs_dump does not export user-defined tokenweight dictionaries. You can manually create the corresponding tokenweight dictionary according to the dictionary information displayed in the error message "WARNING: dictionary xx cannot be automatically exported, please create it manually."
- Common users cannot export directories and synonyms. If a common user attempts to export related data, the message "WARNING: xx not dumped because current user is not a superuser" is displayed. To perform export, you must be the initial user or a user with the SYSADMIN permission. When importing the exported data, ensure that the syntax meets the user permission and operation restrictions. For details, see **CREATE DIRECTORY** and **CREATE SYNONYM**.
- If local data exists in **template1** of the source database instance, restore the output of gs_dump to an empty database (copy from **template0** instead of **template1**). Otherwise, an error may occur because the definition of the added object is copied. To create an empty database without any local additions, copy data from template0 rather than template1. For example:
  ```
  CREATE DATABASE foo WITH TEMPLATE template0;
  ```
- The size of a single tar file must be smaller than 8 GB. (This is the tar file format limitations.) The total size of a .tar archive and any of the other output formats are not limited, except possibly by the OS.
- The dump file generated by gs_dump does not contain the statistics used by the optimizer to make execution plans. Therefore, you are advised to run **ANALYZE** after restoring from a dump file to ensure optimal performance. The dump file does not contain the **ALTER DATABASE... SET** command,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

database users, and other installation settings. These settings are dumped by gs_dumpall.

- gs_dump does not export invalid functions. Before exporting functions, ensure that the functions are valid. If a function is invalid, an alarm is generated, indicating that the function corresponding to the OID is invalid.
- If the parameter type of a non-PACKAGE function is the table column type (table.column%type) or view column type (view.column%type) and the column type is a basic type, the parameter type is converted to the corresponding basic type in the exported function definition. Changing the column type of a table by executing **ALTER TABLE MODIFY COLUMN** and then running **ALTER FUNCTION COMPILE**, or changing the column type of a view query table, rebuilding the view, and executing **ALTER FUNCTION COMPILE** does not change the parameter type in the **PROARGSRC** column of the function in the PG_PROC system catalog. In the exported function definition, the parameter type remains the basic type before the object column type is changed.
- The current design of gs_dump is incompatible with separation of duties. In separation of duties, gs_dump can be used only by the initial user. The file to be imported must also be executed by the initial user.
- If the object to be exported contains a trigger, add the **--use-set-session-authorization --all** parameter and export the object in text format. Otherwise, the owner information of the trigger cannot be correctly restored.
- Using gs_dump will not dump model objects.
- When a system administrator uses gs_dump to export data, objects created by the initial user are not exported. The objects created by the initial user can only be exported by the initial user. When a common user uses gs_dump to export data, only all objects whose owner is the common user are exported.
- If the exported objects contain views that depend on internal functions of packages, an error indicating that the view dependency object does not exist is generated when the objects are imported to the target end for the first time. In this case, you need to run the **gs_dump --schema-only** command to back up metadata and import the metadata to the target end again.
- When using gs_dump to export database-level advanced compression policies, ensure that the entire database is exported and you have the system administrator permission. If the preceding conditions are not met, the related policies are exported as independent policies at the table, partition, or subpartition level instead of being processed as database-level policies.
- gs_dump exports the actual status of a sequence. After the sequence is restored to the target end, the sequence continues in the previous state.
- For some backup files that are exported from an earlier version not supporting Ustore, when they are restored to a version supporting Ustore, tables will be created in Ustore format and subject to Ustore specification limitations. If you want to restore the tables to the Astore format, set **enable_default_ustore_table** to **off**.
- When gs_dump is used to export an interval partitioned table, only the range partitions of the partitioned table can be exported. The interval partitions of the partitioned table cannot be exported. For the sparsely partitioned index of the partitioned table, the same principle applies. That is, only the sparsely partitioned index created on range partitions can be exported. The sparsely partitioned index on the interval partitions cannot be exported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If **ddl_invalid_mode** is set to **invalid**, dependencies on the objects that are dropped will not be deleted, but the dependency relationships will be invalidated. In other words, the pg_depend system catalog still contains dependency records. In this case, when gs_dump is used for backup, an error message "cache lookup failed" will be displayed because a dependency relationship exists but the corresponding dependent object cannot be found. To continue export, you need to restore the dependency relationship (by rebuilding the dependent object or deleting the object that is depended on).
- By default, gs_dump exports invalid views and displays a message if any. An error may be reported when the views are imported in versions earlier than 507.0. You can specify the **--ignore-invalid-views** option to skip exporting invalid views.
- gs_dump does not export data in the **rowid** system column. It exports only data in the user column when **rowid** is used as the data type.
- If the target database to which the full-text search data is imported does not have the corresponding full-text search file, or the **-c** option is specified during the import or export, the import will fail. In this case, you need to create these data objects in the target database and then import the data again, or do not specify the **-c** option during the import or export.
- When gs_dump is used to export a view, if the view definition contains SELECT * and the columns in the base table have been changed after the view is created, the asterisks (*) in the exported file will be expanded based on the current column structure of the target table after the file is imported to the target database.
- During gs_redis redistribution, gs_redis locks the data in CANCELABLE mode and proactively sends a SIGTERM signal to the gs_dump thread that holds the conflicting lock. As a result, gs_dump export may be forcibly terminated and fail.
- If the objects exported by gs_dump contain user views, ensure that the **proc_outparam_override** configuration of the **behavior_compat_options** parameter in the export environment is the same as that in the restoration environment. Otherwise, the import may fail.
- The export result of gs_dump does not contain status information such as all session parameter settings (for example, **behavior_compat_options**). If the statement execution result depends on the status information, the session parameters of the source and target databases must be the same. Otherwise, an error may be reported or data may be inconsistent.

Syntax

```
gs_dump [OPTION]... [DBNAME]
```

Parameters

- DBNAME

  Specifies the database to be connected. No short or long option is required. For example:

  ```
  gs_dump -p port_number testdb -f dump1.sql
  ```

  Or,

  ```
  export PGDATABASE=testdb
  gs_dump -p port_number -f dump1.sql
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

For details about how to use **DBNAME**, see **Table 4-7** in "gsql Usage Guide."

- OPTION

  There are three types of parameters: general parameters, dump parameters, and connection parameters.

  Common parameters:

  – -f, --file=FILENAME

    Sends the output to the specified file or directory. If this parameter is omitted, the standard output is generated. If the output format is **-F c**, **-F d**, or **-F t**, the **-f** parameter must be specified. If the value of the **-f** parameter contains a directory, the current user must have the read and write permission on the directory.

  – -F, --format=c|d|t|p

    Selects an exported file format. The formats are as follows:

    - **c|custom**: outputs a custom-format archive as a directory to be used as the input of gs_restore. This is the most flexible output format in which users can manually select it and reorder the archived items during the restoration process. An archive in this format is compressed by default.
    - d|directory: creates a directory containing directory files and the data files of tables and BLOBs.
    - **t|tar**: outputs a tar-format archive as the input of gs_restore. The .tar format is compatible with the directory format. Extracting a .tar archive generates a valid directory-format archive. However, the .tar archive cannot be further compressed and has an 8-GB limitation on the size of a single table. The order of table data items cannot be changed during restoration.
    - p|plain: generates a text SQL script file. This is the default value.

  – -v, --verbose

    Specifies the verbose mode. If it is specified, gs_dump writes detailed object comments and number of startups/stops to the dump file, and progress messages to standard errors.

  – -V, --version

    Prints the gs_dump version and exits.

  – -Z, --compress=0-9

    Specifies the used compression level.

    Value range: 0 to 9

    - 0 indicates no compression.
    - 1 indicates the lowest compression ratio and the fastest processing speed.
    - 9 indicates the highest compression ratio and the slowest processing speed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- For the custom-format archive, this option specifies the compression level of a single table data segment. By default, data is compressed at a medium level. The .tar archive format and plain-text format do not support compression currently.
- For the parameter value, the part before the first non-digit character will be used. If the string begins with a non-digit character, **0** is used.

– --lock-wait-timeout=TIMEOUT

Specifies the timeout period (in milliseconds) for waiting for a level-1 shared table lock when the dump starts. If no such lock is obtained in the specified period, the dump fails. The timeout interval can be specified in any of the formats accepted by SET statement_timeout.

The lock is held throughout the entire export process, without blocking DML statements.

– -?, --help

Displays help information about gs_dump parameters and exits.

– --ignore-invalid-views

If this parameter is specified, invalid views will not be exported. If invalid views are detected during the export, a message will be displayed.

Dump parameters:

– -a, --data-only

Generates only the data, not the schema (data definition). Dumps the table data and sequence values.

– -b, --blobs

Reserved for function extension. The option is not recommended.

– -c, --clean

Before writing the command of creating database objects into the backup files, writes the command of cleaning (deleting) database objects to the backup files. If no objects exist in the target database, gsql or gs_restore probably displays some error information.

This parameter is used only for the plain-text format. For the archive format, you can specify the option when using gs_restore.

NOTE

- None of the statements exported to the backup file for deleting database objects are cascading DROP statements.
- At t1, database 1 is connected to export table 1 to the **backup01.dmp** file using the **-c** parameter. At t2, view 1 is created in database 1 and depends on table 1. At t3, **backup01.dmp** is restored to database 1 using the **-c** parameter of gs_restore. Non-cascading DROP is used. When table 1 is deleted, the error message "ERROR: table table cannot be dropped because other objects depend on it." is displayed, and when table 1 is created, the error message "ERROR: Relation "table1" already exists." is displayed. In this case, delete related dependent objects or execute the cascading DROP command after evaluating the impact.

– -C, --create

The backup file content starts with the commands of creating the database and connecting to the created database. If the command script

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

is executed in this mode, the database that is connected before the script is executed is not affected.

This parameter is used only for the plain-text format. For the archive format, you can specify the option when using gs_restore.

NOTE

- This option is not supported in M-compatible databases. To import data, create a database in M-compatible mode on the target instance, export data from the source instance, and connect to the newly created M-compatible database on the target instance.
- If **-n**, **-t**, or **--include-table-file** is specified, the **-C** option becomes invalid.

– -E, --encoding=ENCODING

Creates the dump in the specified character set encoding. By default, the dump file is created in the database encoding. Setting the environment variable *PGCLIENTENCODING* to the required dump encoding can also achieve the same result.

NOTE

If transcoding is required for the specified dump encoding and data in the table contains invalid characters, the error message "invalid byte sequence" will be displayed during export. You are advised to specify the **-s** parameter when using gs_dump to export only definitions and execute **COPY** with the encoding fault tolerance option separately to export and import data.

– -g, --exclude-guc=GUC_PARAM

Specifies the **set** command that does not contain related GUC parameters in the exported text file. Currently, only **enable_cluster_resize** is supported. If other parameters are specified, no error is reported but the setting does not take effect.

– -j, --jobs=NUM

Exports data in parallel.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If **-j** is not specified, serial export is used by default.
- The specified number of parallel tasks cannot exceed twice the number of CPU cores on the client. When the client runs in a container, the number of CPUs obtained is the number of CPUs on the host, which may be greater than the number of CPUs that can be used by the container. You are advised to set the degree of parallelism to a value less than twice the number of CPUs that can be used by the client. Parallel export of a table is implemented by concurrently executing multiple transactions through multiple processes. The actual degree of parallelism is limited by the thread pool model of the server and increases the pressure on the server. Set the degree of parallelism based on the actual needs. For example, the degree of parallelism cannot exceed the number of idle CPUs on the client or the maximum number of connections on the server. It is recommended that the number of parallel tasks be greater than or equal to 4.
- Parallel export supports only the directory archive format.
- By default, parallel export is specific to a single table. That is, when only one table is exported, the data of the table can be split into multiple export tasks and exported to multiple files in parallel. You can specify the **--disable-inter-table-parallel** parameter to disable single-table parallel export. When single-table parallel export is disabled, table-level parallel export is used. That is, parallel export takes effect only when data of multiple tables is exported.
- When data is exported and imported in parallel, data in the same table will be placed in the same transaction. That is, the parallel import efficiency is not improved.
- Parallel export conflicts with the **--inserts**, **--column-inserts**, and **--attribute-inserts** options.
- Parallel export conflicts with the **--query** option.
- Bucket tables do not support single-table parallel export.
- The parallel export performance is affected by the compression parameter **-Z**. You are advised to set the compression level properly. The default compression level is medium; alternatively, you can set it to **0** (indicating no compression). A higher compression level indicates higher CPU usage on the client. In addition, the export speed may decrease due to CPU preemption on the client.
- Parallel export cannot be used during the upgrade observation period.
- Single-table parallel export cannot be used on DNs. That is, **-j** and **--disable-inter-table-parallel** must be specified on DNs at the same time.
- Parallel export in distributed mode has certain overheads. When the number of parallel tasks is small, the parallel optimization effect is not obvious.

– -n, --schema=SCHEMA

Dumps only schemas matching the schema names. This option contains the schema and all its contained objects. If this option is not specified, all non-system schemas in the target database will be dumped. Multiple **-n** options can be specified to select multiple modes. The schema parameter is interpreted as a pattern according to the same rule used by the **\d** command of gsql. Therefore, multiple schemas can also be selected by writing wildcard characters in the pattern. When you use wildcard characters, quote patterns to prevent the shell from expanding the wildcard characters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If **-n** is specified, gs_dump does not dump any other database objects that the selected schemas might depend upon. Therefore, there is no guarantee that the results of a specific-schema dump can be automatically restored to an empty database.
- If **-n** is specified, the non-schema objects are not dumped.
- In M-compatible mode, if a database with **templatem** is created by running **CREATE DATABASE**, data is exported by specifying **db_name**; if a database is created by running **CREATE DATABASE** *db_name*, data is exported by specifying **-n** because such database is equivalent to a schema.
- GaussDB automatically converts uppercase letters in object names to lowercase letters. If a schema name contains uppercase letters, add extra quotation marks, for example, **-n '"Sch1"'** or **-n "\"Sch1\""**.
- When **-n** is specified, pattern "*" cannot be used to match system schemas in the database during export.

Multiple schemas can be dumped. For example, in the following example, both **sch1** and **sch2** are dumped.

```
gs_dump -h host_name -p port_number testdb -f backup/bkp_shl2.sql -n sch1 -n sch2
```

– -N, --exclude-schema=SCHEMA

Does not dump any tables matching the table pattern. The pattern is interpreted according to the same rule as for -n. **-N** can be specified multiple times to exclude schemas matching any of the specified patterns.

When both -n and -N are specified, the schemas that match at least one -n option but no -N is dumped. If -N is specified and -n is not, the schemas matching -N are excluded from what is normally dumped.

Dump allows you to exclude multiple schemas during dumping. For example, in the following example, **sch1** and **sch2** are excluded during the dump.

```
gs_dump -h host_name -p port_number testdb -f backup/bkp_shl2.sql -N sch1 -N sch2
```

NOTE

GaussDB automatically converts uppercase letters in object names to lowercase letters. If a schema name contains uppercase letters, add extra quotation marks, for example, **-N '"Sch1"'** or **-N "\"Sch1\""**.

– -o, --oids

Dumps OIDs as parts of the data in each table. Use this option if your application references the OID columns in some way (for example, in a FOREIGN KEY constraint). If the preceding situation does not occur, do not use this parameter. This option cannot be used together with the **--inserts/--column-inserts** option.

– -O, --no-owner

Do not output commands to set ownership of objects to match the original database. By default, gs_dump issues the ALTER OWNER or SET SESSION AUTHORIZATION command to set ownership of created database objects. These statements will fail when the script is running unless it is started by a system administrator (or the same user who owns all of the objects in the script). To make a script that can be used by any user for restoration and give the user ownership of all objects, specify **-O**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

This parameter is used only for the plain-text format. For the archive format, you can specify the option when using gs_restore.

- -s, --schema-only

  Dumps only the object definition (schema) but not data.

- -q, --target=VERSION

  Exports text files compatible with databases of other versions. Currently, parameters of V1 and V5 are supported. If other parameters are specified, no error is reported but the setting does not take effect. The **v1** value means to export GaussDB v5 data as a text file compatible with GaussDB v1. The **v5** value means to export GaussDB v5 data as a text file, reducing errors that may occur when data is imported to a GaussDB v5 database.

  When **v1** is specified, you are advised to use it along with parameters such as **--exclude-guc="enable_cluster_resize"**, **--exclude-function**, and **--exclude-with**. Otherwise, an error may be reported during data import to a GaussDB v1 database.

- -S, --sysadmin=NAME

  Reserved for function extension. The option is not recommended.

- -t, --table=TABLE

  Specifies a list of tables, views, sequences, or foreign tables not to be dumped. You can use multiple **-t** parameters or wildcard characters to specify tables. For example, in the following example, both **schema1.table1** and **schema2.table2** are dumped.

  ```
  gs_dump -h host_name -p port_number testdb -f backup/bkp_shl2.sql -t schema1.table1 -t schema2.table2
  ```

  When you use wildcard characters, quote patterns to prevent the shell from expanding the wildcard characters.

  The **-n** and **-N** options have no effect when **-t** is used because tables selected by using **-t** are not affected the **-n** and **-N** options.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The number of **-t** parameters must be less than or equal to 100.
- If the number of **-t** parameters is greater than 100, you are advised to use the **--include-table-file** parameter to replace some **-t** parameters.
- If **-t** is specified, gs_dump does not dump any other database objects that the selected tables might depend upon. Therefore, there is no guarantee that the results of a specific-table dump can be automatically restored to an empty database.
- **-t** *tablename* only dumps visible tables in the default search path. **-t** **.tablename* dumps the tables specified by *tablename* in all schemas (except the system schemas) in the database. **-t** *schema.table* dumps tables in a specific schema.
- **-t** *tablename* does not export trigger information from a table.
- If the table name contains uppercase letters, you need to add **\"** to the table name when using the **-t** parameter to specify the export. To export the **"abC"** table, specify **-t \"abC\"**. To export the **schema."abC"** table, specify **-t schema.\"abC\"**.
- **-t ""** does not match any table.
- When the **-t** parameter contains multiple dot (.) separators, only the last two separators are effective. The name between the second-to-last and last separators is the schema name, and the name after the last separator is the table name. For example, running **-t schema_a.schema_b.schema_c.table1** actually exports **table1** from **schema_c** and running **-t schema_a.schema_b.schema_c.*.table1** actually exports **table1** from all schemas.

– --include-table-file=FILENAME

  Specifies the table file to be dumped.

– -T, --exclude-table=TABLE

  Specifies a list of table, view, sequence, or foreign table objects not to be dumped. You can use multiple **-T** parameters or wildcard characters to specify multiple lists.

  When both **-t** and **-T** are specified, it will dump the objects in the **-t** list, but not those in the **-T** list.

  For example, in the following example, **table1** and **table2** are excluded during the dump.

  ```
  gs_dump -h host_name -p port_number testdb -f backup/bkp_shl2.sql -T table1 -T table2
  ```

  NOTE

  For auto-increment columns, the **-T** parameter does not take effect.

– --exclude-table-file=FILENAME

  Specifies the table file not to be dumped.

  NOTE

  Same as **--include-table-file**, the content format of this parameter is as follows:

  schema1.table1

  schema2.table2

  .....

– -x, --no-acl

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Prevents the dumping of access permissions (**GRANT**/**REVOKE** commands). Only ACL objects are affected. Privilege objects are not affected.

- --exclude-function

  Specifies that functions and stored procedures are not exported.

- --exclude-with

  Specifies that the description such as **WITH(orientation=row, compression=on)** is not added to the end of the exported table definition.

- --binary-upgrade

  Reserved for function extension. The option is not recommended.

  NOTE

  M-compatible databases do not support this option.

- --binary-upgrade-usermap="USER1=USER2"

  Reserved for function extension. The option is not recommended.

- --column-inserts|--attribute-inserts

  Exports data by running the INSERT command with explicit column names {INSERT INTO table (column, ...) VALUES ...}. This will cause a slow restoration. However, since this option generates an independent command for each row, an error in reloading a row causes only the loss of the row rather than the entire table content.

  NOTE

  M-compatible databases do not support this option.

- --disable-dollar-quoting

  Disables the use of dollar sign ($) for function bodies, and forces them to be quoted using the SQL standard string syntax.

- --disable-inter-table-parallel

  This option takes effect only during parallel export. If this option is specified, table-level parallel export is used.

- --include-alter-table

  Exports columns that are deleted from a table.

- --disable-triggers

  Reserved for function extension. The option is not recommended.

- --exclude-table-data=TABLE

  Does not dump table data that matches any patterns. The pattern is interpreted according to the same rule as for **-t**.

  --exclude-table-data can be entered more than once to exclude tables matching any of several patterns. When the user needs the specified table definition rather than data in the table, this option is helpful.

  To exclude data of all tables in the database, see **-s, --schema-only**.

- --include-query-file=FILENAME

  When exporting table data, you can use this parameter to configure predicate filter conditions in batches in the file, so as to export specified table data.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Predicate conditions are separated by '\n'. **FILENAME** supports the following formats:

i. [schema_name.][table_name:] query_clause

ii. query=[schema_name.][table_name:] query_clause

NOTE

- The line starting with a number sign (#) in the file is regarded as a comment line.
- **--include-query-file** and **--query** can be used together, but only one predicate condition can be specified for the same table.
- For details about other rules of predicate conditions, see the description of the **--query** parameter.

– --inserts

Dumps data when the **INSERT** command (rather than the **COPY** command) is delivered, resulting in slow recovery.

However, since this option generates an independent command for each row, an error in reloading a row causes only the loss of the row rather than the entire table content. Note that the restoration may fail if you rearrange the column order. The **--column-inserts** option is unaffected against column order changes.

NOTE

M-compatible databases do not support this option.

– --log-level=LOG|DEBUG|DEBUG1|DEBUG2|DEBUG3

Specifies the log level of gs_dump. **LOG** records normal execution information, and **DEBUG** records some debugging information. A larger **DEBUG** value indicates a higher log level and more debugging details.

– --no-security-labels

Reserved for function extension. The option is not recommended.

– --no-tablespaces

Does not select any tablespaces. All the objects will be created during the restoration process, no matter which tablespace is selected when using this option.

This parameter is used only for the plain-text format. For the archive format, you can specify the option when using gs_restore.

– --no-unlogged-table-data

Reserved for function extension. The option is not recommended.

– --non-lock-table

Shields the table locking behavior during the export using gs_dump. This parameter is called only between APIs. You are advised not to call this parameter.

NOTE

When gs_dump is used to export a table, a shared lock is added to the table to be exported in the entire transaction by default. This lock blocks the execution of concurrent DDL statements on the table, ensuring that no error is reported due to the impact on concurrent DDL statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

– --query=[schema_name.][table_name:]query_clause

  Adds predicate conditions to export specified table data.

  query_clause is a standard query clause starting with WHERE.

  NOTE

  - If predicate conditions need to be added for multiple tables, multiple **--query** parameters are required.
  - You can specify multiple **query** parameters at the same time, but only one **query** parameter can be specified for the same table.
  - The **-t** parameter takes precedence over the **--query** parameter. When both the **-t** and **--query** parameters are used, tables to be exported are selected based on the **-t** parameter, and then filtered based on the predicates.
  - The **--query** parameter only concatenates predicates and does not escape them. You need to escape them by yourself if necessary.
  - The used predicates must comply with the SQL standard and can be properly executed. Otherwise, an error will be reported during data export.
  - Predicates cannot be specified for views and materialized views. Otherwise, an error will be reported during data export.
  - The export performance is affected by the query performance of predicates.
  - The WHERE clause in the **--query** parameter must be enclosed in single or double quotation marks. If the WHERE clause contains a filter condition of the character type, the WHERE clause must be enclosed in double quotation marks.

    For example, if the filter condition in the WHERE clause is not of the character type, you can use single or double quotation marks to enclose the filter condition.

    ```
    --query=test_schema.test_table:'where test_attr > 3'
    --query=test_schema.test_table:"where test_attr > 3"
    ```

    For example, if the filter condition in the WHERE clause is of the character type, the filter condition must be enclosed in double quotation marks.

    ```
    --query=test_schema.test_table:"where test_attr = 'a'"
    ```

  - GaussDB automatically converts uppercase letters in object names to lowercase letters. If **schema_name**, **table_name**, or **query_clause** contains uppercase letters or special characters, extra quotation marks are required.

    For example, if the schema name contains uppercase letters or special characters, do as follows:

    ```
    --query="\"TEST_SCHEMA\"".test_table:"where test_attr > 3"
    ```

    For example, if the predicate condition contains uppercase letters (the column name contains uppercase letters), do as follows:

    ```
    --query=test_schema.test_table:"where \"TEST_attr\" > 3"
    ```

– --quote-all-identifiers

  Forcibly quotes all identifiers. In the migration to a later version, additional keywords may be introduced. This option is helpful.

  NOTE

  M-compatible databases do not support this option.

– --section=SECTION

  Dumps specified sections (pre-data, data, and post-data).

– --serializable-deferrable

  Uses a serializable transaction during the dump process to ensure that the snapshot used is consistent with the subsequent database state. To

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

achieve this, it must wait for a specific point in an error-free transaction stream (where all concurrent read/write transactions that might conflict with the dump have either committed or rolled back). This guarantees a successful dump and avoids triggering serialization_failure errors that would force other transactions to retry.

This option provides no benefit for dumps intended solely for DR (as DR only requires a fully usable restored dataset, not a strictly consistent snapshot). It is ideal for the following scenario: generating a backup for reports or other read-only workloads while the primary database is continuously updated by business operations—where the copy requires a strictly consistent state. Without this option, the dump might reflect an anomalous state that "no serial execution could ever produce." For example, in batch processing, the dump might contain contradictory results such as "a batch is marked as closed, but its constituent items have not yet fully appeared."

This option will make no difference if there are no active read/write transactions when gs_dump is started. If the read/write transactions are active, the dump start time will be delayed for an uncertain period.

– --use-set-session-authorization

Specifies that the standard SQL **SET SESSION AUTHORIZATION** command rather than **ALTER OWNER** is generated to determine the object ownership. When the **SET SESSION AUTHORIZATION** command is used to dump data, if the object history in the dump file is incorrect, the data may not be restored correctly. A dump by running **SET SESSION AUTHORIZATION** requires the system administrator permission, whereas running **ALTER OWNER** requires a lower permission. However, the SET SESSION AUTHORIZATION statement supports ciphertext passwords. The script exported using this parameter may not be restored. Therefore, you are advised not to use this parameter to export the script.

NOTE

Application scope of SET SESSION AUTHORIZATION:

- A system administrator can switch to a common user using the SET SESSION AUTHORIZATION statement, but cannot switch to the initial user, SYSADMIN, OPRADMIN, MONADMIN, POLADMIN, or AUDITADMIN.
- Other users cannot switch to another user using the SET SESSION AUTHORIZATION statement.

– --all

Specifies that the gs_dump tool is called by the gs_dumpall tool. You are advised not to use this parameter unless you need to export triggers.

– --with-copy-format=FILE_FORMAT

Specifies the format of the data file exported by gs_dump.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The value of **FILE_FORMAT** is a string. Only the TEXT and CSV formats are allowed. The default value is **TEXT**.
- The value of **FILE_FORMAT** can be **CSV** only in the directory archive format.
- When **FILE_FORMAT** is set to **CSV**, **--inserts**, **--column-inserts**, or **--attribute-inserts** cannot be used at the same time.
- After data is exported in the directory archive format, users must use the gs_restore tool to import and restore the data. In this case, users do not need to pay attention to the format of the exported data file. If users specify CSV as the format of the data file to be exported, **--with-copy-format** can be set to **CSV**.

– --with-encryption=AES128

Specifies that dumping data needs to be encrypted using AES-128.

– --with-key=KEY

The AES-128 key rules are as follows:

Consists of 8 to 16 characters.

Contains at least three of the following character types: uppercase characters, lowercase characters, digits, and special characters (limited to ~!@#$ %^&*()-_=+\|[{}];:,<.>/?).

NOTE

Stored procedures and functions cannot be exported in encrypted mode.

– --with-salt=RANDVALUES

gs_dumpall uses this parameter to transfer a random value.

– --include-nodes

Includes the TO NODE or TO GROUP statement in the dumped CREATE TABLE/CREATE FOREIGN TABLE statement.

– --include-extensions

Includes extensions in the dump.

NOTICE

Extensions are for internal use only. Only advanced packages or extensions in the database can be operated.

– --include-depend-objs

Includes information about the objects that depend on the specified object in the backup result. This parameter takes effect only if the **-t** or **--include-table-file** parameter is specified.

– --exclude-self

Excludes information about the specified object from the backup result. This parameter takes effect only if the **-t** or **--include-table-file** parameter is specified.

– --pipeline

Uses a pipe to transmit the password. This parameter cannot be used on devices.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

– --no-transaction

By default (when this option is disabled), when files in plain-text format are exported, statements such as CREATE, ALTER ... OWNER TO, SET SESSION AUTHORIZATION, and RESET SESSION AUTHORIZATION will be wrapped in a separate transaction block for execution. If this option is enabled, these statements are output directly without being wrapped in BEGIN/COMMIT.

– --dont-overwrite-file

The existing files in plain-text, .tar, and custom formats will be overwritten. This option is not applicable to the directory format.

For example:

Assume that the backup.sql file exists in the current directory. If you specify -f backup.sql in the input command, and the backup.sql file is generated in the current directory, the original file will be overwritten.

If the backup file exists and **--dont-overwrite-file** is specified, an error will be reported with the message that the dump file exists.

```
gs_dump -p port_number testdb -f backup.sql -F plain --dont-overwrite-file
```

NOTE

- **-s/--schema-only** and **-a/--data-only** do not coexist.
- **-c/--clean** and **-a/--data-only** do not coexist.
- **--inserts/--column-inserts** and **-o/--oids** do not coexist because OIDS cannot be set using the INSERT statement.
- **--role** must be used with **--rolepassword**.
- **--binary-upgrade-usermap** must be used with **--binary-upgrade**.
- **--include-depend-objs**/**--exclude-self** takes effect only when **-t**/**--include-table-file** is specified.
- **--exclude-self** must be used with **--include-depend-objs**.
- **--with-encryption=AES-128** supports only **-F p/plain**.
- **--with-key=KEY** supports only **-F p/plain**.
- **--with-salt=RANDVALUES** is called by gs_dumpall and does not require manual input.

Connection parameters:

– -h, --host=HOSTNAME

Specifies the host name. If the value begins with a slash, it is used as the directory for the UDS. By default, the value is obtained from the *PGHOST* environment variable (if set).

This parameter applies only to external connections of a cluster. For the localhost in the cluster, you can use IPv4 address **127.0.0.1** or IPv6 address **::1**.

– -p, --port=PORT

Specifies the host port number. If the thread pool is enabled, you are advised to use the pooler port, that is, the host port number plus 1. By default, the value is obtained from the *PGPORT* environment variable (if set).

– -U, --username=NAME

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the username for connecting to a host. The initial user cannot be used for cross-node execution. By default, the value is obtained from the *PGUSER* environment variable (if set).

NOTE

Before exporting and importing data as a common user, grant the permission to create schemas in the corresponding database to the common user, for example, **GRANT CREATE ON DATABASE db**_*xxx* **TO user**_*xxx*;.

- -w, --no-password

  Never issues a password prompt. The connection attempt fails if the host requires password for authentication and the password is not provided in other ways. This parameter is useful in batch jobs and scripts in which no user password is required.

- -W, --password=PASSWORD

  Specifies the user password for connection. If the host uses the trust authentication policy, administrators do not need to enter the **-W** option. If the **-W** option is not provided and you are not a system administrator, the system will ask you to enter a password. To ensure system security, you are advised to enter the password in interactive mode.

- --role=ROLENAME

  Specifies a role name to be used for creating the dump. If this option is selected, the SET ROLE command will be issued after the database is connected to gs_dump. It is useful when the authenticated user (specified by **-U**) lacks the permissions required by gs_dump. It allows the user to switch to a role with the required permissions. Some installations have a policy against logging in directly as a system administrator. This option allows dumping data without violating the policy.

- --rolepassword=ROLEPASSWORD

  Password of the role

Examples

In the following examples, **backup/MPPDB_backup.sql** indicates an exported file, where **backup** indicates the relative path of the current directory. **37300** indicates the port number of the database server. **testdb** indicates the name of the database to be accessed.

NOTE

Before exporting files, ensure that the directory exists and you have the read and write permissions on the directory.

Example 1

Use gs_dump to export the full information of the **testdb** database. The exported **MPPDB_backup.sql** file is in plain-text format.

```
gs_dump -U omm -f backup/MPPDB_backup.sql -p 37300 testdb -F p
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 09:49:17]: The total objects number is 356.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 09:49:17]: [100.00%] 356 objects have been dumped.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 09:49:17]: dump database testdb successfully
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 09:49:17]: total time: 1274  ms
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example 2

Use gs_dump to export the full information of the **testdb** database. The exported **MPPDB_backup.tar** file is in .tar format.

```
gs_dump -U omm -f backup/MPPDB_backup.tar -p 37300 testdb -F t
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:02:24]: The total objects number is 1369.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:02:53]: [100.00%] 1369 objects have been dumped.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:02:53]: dump database testdb successfully
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:02:53]: total time: 50086  ms
```

Example 3

Use gs_dump to export the full information of the **testdb** database. The exported **MPPDB_backup.dmp** file is in custom format.

```
gs_dump -U omm -f backup/MPPDB_backup.dmp -p 37300 testdb -F c
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:05:40]: The total objects number is 1369.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:06:03]: [100.00%] 1369 objects have been dumped.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:06:03]: dump database testdb successfully
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:06:03]: total time: 36620  ms
```

Example 4

Use gs_dump to export the full information of the **testdb** database. The exported **MPPDB_backup** file is in directory format.

```
gs_dump -U omm -f backup/MPPDB_backup -p 37300  testdb -F d
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:16:04]: The total objects number is 1369.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:16:23]: [100.00%] 1369 objects have been dumped.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:16:23]: dump database testdb successfully
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:16:23]: total time: 33977  ms
```

Example 5

Use gs_dump to export the information of the **testdb** database, excluding the information of the table specified in the **/home/MPPDB_temp.sql** file. The exported MPPDB_backup.sql file is in plain-text format.

```
gs_dump -U omm -p 37300 testdb --exclude-table-file=/home/MPPDB_temp.sql -f backup/MPPDB_backup.sql
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:37:01]: The total objects number is 1367.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:37:22]: [100.00%] 1367 objects have been dumped.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:37:22]: dump database testdb successfully
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:37:22]: total time: 37017  ms
```

Example 6

Use gs_dump to export only the information about the views that depend on the **testtable** table.

```
gs_dump -U omm -s -p 37300 testdb -t PUBLIC.testtable --include-depend-objs --exclude-self -f backup/MPPDB_backup.sql -F p
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-15 14:12:54]: The total objects number is 331.
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-15 14:12:54]: [100.00%] 331 objects have been dumped.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-15 14:12:54]: dump database testdb successfully
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-15 14:12:54]: total time: 327  ms
```

Related Tools

gs_dumpall for Exporting All Database Information, **gs_restore for Importing Data**, and **gsql for Connecting to a Database**

4.3.3 gs_dumpall for Exporting All Database Information

Description

gs_dumpall, provided by GaussDB, is used to export all database information, including the data of the default **postgres** database, data of user-specified databases, and global objects of all databases in a cluster.

When gs_dumpall is used to export data, other users can still access (read or write) the database cluster.

gs_dumpall can export complete, consistent data. For example, if gs_dumpall is started to export database from the entire cluster at T1, data of the cluster databases at that time point will be exported, and modifications on the cluster databases after that time point will not be exported.

gs_dumpall exports all databases from the entire cluster in two parts:

- gs_dumpall exports all global objects, including information about database users and groups, tablespaces, and attributes (for example, global access permissions).
- gs_dumpall calls gs_dump to export SQL scripts from each cluster database, which contain all the SQL statements required to restore databases.

The exported files are both plain-text SQL scripts. Use **gsql for Connecting to a Database** to execute them to restore databases.

gs_dumpall supports SSL-encrypted communication. The method is the same as that of using gsql.

Before using gs_dumpall, ensure that the gs_dumpall version is consistent with the gs_dump version and database version. gs_dumpall of a later version may not be fully compatible with the kernel data of an earlier version.

Because running gs_dumpall will call gs_dump, some limitations on exporting objects within a database are the same as those of gs_dump.

Precautions

- Do not modify any exported file. Otherwise, restoration may fail.
- To ensure the data consistency and integrity, gs_dumpall acquires a shared lock on the table to be dumped. If a shared lock has been set for the table in other transactions, gs_dumpall locks the table after it is released. If the table cannot be locked within the specified time, the dump fails. You can customize the timeout interval to wait for lock release by specifying the **--lock-wait-timeout** parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- During an export, gs_dumpall reads all tables in a database. Therefore, you need to connect to the database as a database cluster administrator to export a complete file. When you use gsql to execute scripts, administrator permissions are also required to add users and user groups, and create databases. Before importing a backup, you need to verify the security to prevent administrator permissions from being exploited.
- If you use gs_dumpall to export all database objects and want to import them to a new instance environment, ensure that the names and permissions of the users used for the export and import are the same. Otherwise, an error message will be displayed, indicating that the names are inconsistent or the permissions are insufficient.
- For scheduled tasks, this tool can export only scheduled tasks created using CREATE EVENT or non-periodic scheduled tasks created using advanced packages from a MySQL-compatible database.
- gs_dumpall does not export user-defined tokenweight dictionaries. You can manually create the corresponding tokenweight dictionary according to the dictionary information displayed in the error message "WARNING: dictionary xx cannot be automatically exported, please create it manually."
- Only one M-compatible database can be created globally. If an M-compatible database is exported, ensure that no other M-compatible database exists in the target instance environment.
- If transcoding is required for the specified dump encoding and data in the table contains invalid characters, the error message "invalid byte sequence" will be displayed during export. You are advised to specify the **-s** parameter when using gs_dump to export only definitions and execute **COPY** with the encoding fault tolerance option separately to export and import data.
- The current design of gs_dumpall is incompatible with separation of duties. In separation of duties, gs_dumpall can be used only by the initial user. The file to be imported must also be executed by the initial user.
- When a system administrator uses gs_dumpall to export data, objects created by the initial user are not exported. The objects created by the initial user can only be exported by the initial user.
- If the exported objects contain views that depend on internal functions of packages, an error indicating that the view dependency object does not exist is generated when the objects are imported to the target end for the first time. In this case, you need to run the **gs_dumpall --schema-only** command to back up metadata and import the metadata to the target end again.
- By default, gs_dumpall exports invalid views and displays a message if any. An error may be reported when the views are imported in versions earlier than 507.0. You can specify the **--ignore-invalid-views** option to skip exporting invalid views.

Syntax

```
gs_dumpall [OPTION]...
```

Parameters

Common parameters:

- -f, --filename=FILENAME

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Sends the output to the specified file. If this option is omitted, the standard output is used.

- -v, --verbose

  Specifies the verbose mode. This will cause gs_dumpall to output detailed object comments and start/stop times to the dump file, and progress messages to standard errors.

- -V, --version

  Prints the gs_dumpall version and exits.

- --lock-wait-timeout=TIMEOUT

  Specifies the timeout period (in milliseconds) for waiting for a shared table lock when the dump starts. If no such lock is obtained in the specified period, the dump fails. The timeout interval can be specified in any of the formats accepted by SET statement_timeout.

- -?, --help

  Displays help about the command line parameters for gs_dumpall and exits.

- --ignore-invalid-views

  If this parameter is specified, invalid views will not be exported. If invalid views are detected during the export, a message will be displayed.

Dump parameters:

- -a, --data-only

  Dumps only the data, not the schema (data definition).

- -c, --clean

  Runs SQL statements to delete databases before rebuilding them. Statements for dumping roles and tablespaces are added.

- -g, --globals-only

  Dumps only global objects (roles and tablespaces) but no databases.

- -o, --oids

  Dumps OIDs as parts of the data in each table. Use this parameter if your application references the OID columns in some way. If the preceding situation does not occur, do not use this parameter.

- -O, --no-owner

  Do not output commands to set ownership of objects to match the original database. By default, gs_dumpall issues the ALTER OWNER or SET SESSION AUTHORIZATION statement to set ownership of created schema elements. These statements will fail when the script is running unless it is started by a system administrator (or the same user who owns all of the objects in the script). By specifying **-O**, you can compile a script that can be stored by any user. The script grants the user the permission to own all objects because the ALTER OWNER or SET SESSION AUTHORIZATION statement is not used and the execute user permission is always used during the import. Therefore, before importing the dump file, check whether there are risks in the dump file. For example, check whether the dump file contains a privilege escalation statement and whether the statement is known to the administrator.

- -r, --roles-only

  Dumps only roles but not databases or tablespaces.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- -s, --schema-only

  Dumps only the object definition (schema) but not data.

- -S, --sysadmin=NAME

  Reserved for function extension. The option is not recommended.

- -t, --tablespaces-only

  Dumps only tablespaces but not databases or roles.

- -x, --no-privileges

  Prevents the dumping of access permissions (**GRANT**/**REVOKE** commands).

- --column-inserts|--attribute-inserts

  Exports data by running the **INSERT** command with column names specified. This will cause a slow restoration. However, since this option generates an independent command for each row, an error in reloading a row causes only the loss of the row rather than the entire table content.

  NOTE

  M-compatible databases do not support this option. This option is skipped when an M-compatible database is exported.

- --disable-dollar-quoting

  Disables the use of dollar sign ($) for function bodies, and forces them to be quoted using the SQL standard string syntax.

- --disable-triggers

  Reserved for function extension. The option is not recommended.

- --inserts

  Dumps data when the INSERT statement (rather than COPY) is issued. This will cause a slow restoration. The restoration may fail if you rearrange the column order. The --column-inserts parameter is safer against column order changes, though even slower.

  NOTE

  M-compatible databases do not support this option. This option is skipped when an M-compatible database is exported.

- --no-security-labels

  Reserved for function extension. The option is not recommended.

- --no-tablespaces

  Do not output statements to create tablespaces or select tablespaces for objects. All the objects will be created during the restoration process, no matter which tablespace is selected when using this option.

- --no-unlogged-table-data

  Reserved for function extension. The option is not recommended.

- --include-alter-table

  Exports information about deleted columns in the table.

- --quote-all-identifiers

  Forcibly quotes all identifiers. This parameter is useful when you dump a database for migration to a later version, in which additional keywords may be introduced.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> M-compatible databases do not support this option. This option is skipped when an M-compatible database is exported.

- --dont-overwrite-file

  Do not overwrite the current file.

- --use-set-session-authorization

  Specifies that the standard SQL **SET SESSION AUTHORIZATION** command rather than **ALTER OWNER** is generated to determine the object ownership. This makes dumping more standard. However, if a dump file contains objects that have historical problems, restoration may fail. A dump by running **SET SESSION AUTHORIZATION** requires the system administrator permissions, whereas running **ALTER OWNER** requires lower permissions. However, the SET SESSION AUTHORIZATION statement supports user and permission switching using a ciphertext password. The script exported using this parameter may not be restored. Therefore, you are advised not to use this parameter to export the script.

  > **NOTE**
  >
  > Application scope of SET SESSION AUTHORIZATION:
  >
  > - A system administrator can switch to a common user using the SET SESSION AUTHORIZATION statement, but cannot switch to the initial user, SYSADMIN, OPRADMIN, MONADMIN, POLADMIN, or AUDITADMIN.
  > - Other users cannot switch to another user using the SET SESSION AUTHORIZATION statement.

- --with-encryption=AES128

  Specifies that dumping data needs to be encrypted using AES-128.

- --with-key=KEY

  The AES-128 key rules are as follows:

  Consists of 8 to 16 characters.

  Contains at least three of the following character types: uppercase characters, lowercase characters, digits, and special characters (limited to ~!@#$ %^&*()-_=+\|[{}];:,<.>/?).

- --include-extensions

  Backs up all CREATE EXTENSION statements if the include-extensions parameter is set.

  NOTICE

  Extensions are for internal use only. Only advanced packages or extensions in the database can be operated.

- --include-templatedb

  Includes template databases during the dump.

- --dump-nodes

  Includes nodes and Node Groups during the dump.

- --include-nodes

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Includes TO NODE statement in the dumped CREATE TABLE statement.

- --include-buckets

  Reserved for function extension. The option is not recommended.

- --dump-wrm

  Includes workload resource manager (resource pool, load group, and load group mapping) during the dump.

  NOTE

  This option is not supported when an M-compatible database is connected for export.

- --binary-upgrade

  Reserved for function extension. The option is not recommended.

  NOTE

  M-compatible databases do not support this option. This option is skipped when an M-compatible database is exported.

- --binary-upgrade-usermap="USER1=USER2"

  Reserved for function extension. The option is not recommended.

- --non-lock-table

  This parameter is used only for inter-software API calling.

- --tablespaces-postfix

  Reserved for function extension. The option is not recommended.

- --parallel-jobs

  Specifies the number of concurrent backup processes. The value range is 1–1000.

- --no-transaction

  By default, when files in plain-text format are exported, statements such as CREATE, ALTER ... OWNER TO, SET SESSION AUTHORIZATION, and RESET SESSION AUTHORIZATION will be wrapped in a separate transaction block for execution. If this option is enabled, these statements are output directly without being wrapped in BEGIN/COMMIT.

- --pipeline

  Uses a pipe to transmit the password. This parameter cannot be used on devices.

NOTE

- The **-g/--globals-only** and **-r/--roles-only** parameters do not coexist.
- The **-g/--globals-only** and **-t/--tablespaces-only** parameters do not coexist.
- The **-r/--roles-only** and **-t/--tablespaces-only** parameters do not coexist.
- The **-s/--schema-only** and **-a/--data-only** parameters do not coexist.
- The **-r/--roles-only** and **-a/--data-only** parameters do not coexist.
- The **-t/--tablespaces-only** and **-a/--data-only** parameters do not coexist.
- The **-g/--globals-only** and **-a/--data-only parameters** do not coexist.
- **--tablespaces-postfix** must be used with **--binary-upgrade**.
- **--binary-upgrade-usermap** must be used with **--binary-upgrade**.
- **--parallel-jobs** must be used with **-f/--file**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Connection parameters:

- -h, --host=HOSTNAME

  Specifies the host name. If the value begins with a slash (/), it is used as the UDS directory. The default value is taken from the *PGHOST* environment variable. If it is not set, a UDS connection is attempted.

  This parameter is used only for defining names of the hosts outside a cluster. For the localhost in the cluster, you can use IPv4 address **127.0.0.1** or IPv6 address **::1**.

  Environment variable: *PGHOST*

- -l, --database=DATABASENAME

  Specifies the name of the database connected to dump all objects and discover other databases to be dumped. If this parameter is not specified, the **postgres** database will be used. If the **postgres** database does not exist, **template1** will be used.

  NOTE

  No matter which database is specified as the database to be connected for export, you must connect to the default database when using gsql to import data to a new instance. Otherwise, syntax incompatibility may occur, resulting in import errors. In particular, if the M-compatible database is incorrectly connected for import, the creation fails because CREATE DATABASE on the M-compatible database is equivalent to CREATE SCHEMA.

- -p, --port=PORT

  Specifies the TCP port listened on by the server or the local UDS file name extension to ensure a correct connection. The default value is the *PGPORT* environment variable.

  If the thread pool is enabled, you are advised to use the pooler port, that is, the listening port number plus 1.

  Environment variable: *PGPORT*

- -U, --username=NAME

  Specifies the name of the connected user. The initial user cannot be used for cross-node execution.

  Environment variable: *PGUSER*

- -w, --no-password

  Never issues a password prompt. The connection attempt fails if the server requires password for authentication and the password is not provided in other ways. This parameter is useful in batch jobs and scripts in which no user password is required.

- -W, --password=PASSWORD

  Specifies the user password for connection. If the authentication policy of the host is **trust**, the password of the system administrator is not verified. That is, you do not need to enter the **-W** option. If this parameter is not specified and you are not a system administrator, the system prompts you to enter the password in interactive mode. To ensure system security, you are advised to enter the password in interactive mode.

- --role=ROLENAME

  Specifies a role name to be used for creating the dump. This option causes gs_dumpall to issue the SET ROLE statement after connecting to the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

database. It is useful when the authenticated user (specified by **-U**) lacks the permissions required by gs_dumpall. It allows the user to switch to a role with the required permissions. Some installations have a policy against logging in directly as a system administrator. This option allows dumping data without violating the policy.

- --rolepassword=ROLEPASSWORD

  Specifies the password of the specific role.

Notes

- gs_dumpall internally calls gs_dump. For details about the diagnosis information, see **gs_dump for Exporting Database Information**.
- After restoration, it is advised to run **ANALYZE** on each database so that the optimizer can provide useful statistics.
- gs_dumpall requires all needed tablespace directories to be empty before the restoration. Otherwise, database creation will fail if the databases are in non-default locations.

Examples

Use gs_dumpall to export all databases from a cluster at a time.

NOTE

gs_dumpall supports only plain-text format export. Therefore, only gsql can be used to restore a file exported using gs_dumpall.

- Export all databases.

  backup/bkp2.sql indicates the exported file, **29900** indicates the port number of the database server, and **omm** indicates the username.

```
gs_dumpall -U omm -f ./backup/bkp2.sql -p 29900
gs_dump[user='omm'][localhost][port='29900'][dbname='postgres'][2024-11-11 11:00:01]: The total objects number is 458.
gs_dump[user='omm'][localhost][port='29900'][dbname='postgres'][2024-11-11 11:00:01]: [100.00%] 458 objects have been dumped.
gs_dump[user='omm'][localhost][port='29900'][dbname='postgres'][2024-11-11 11:00:01]: dump database dbname='postgres' successfully
gs_dump[user='omm'][localhost][port='29900'][dbname='postgres'][2024-11-11 11:00:01]: total time: 3239  ms
gs_dumpall[user='omm'][localhost][port='29900'][2024-11-11 11:00:01]: dumpall operation successful
gs_dumpall[user='omm'][localhost][port='29900'][2024-11-11 11:00:01]: total time: 3305  ms
```

- Specify four jobs to concurrently export all databases.

```
gs_dumpall -U omm -f ./backup/bkp2.sql -p 29900 --parallel-jobs 4
gs_dump[user='omm'][localhost][port='29900'][dbname='postgres'][2024-11-11 11:04:13]: The total objects number is 458.
gs_dump[user='omm'][localhost][port='29900'][dbname='postgres'][2024-11-11 11:04:13]: [100.00%] 458 objects have been dumped.
gs_dump[user='omm'][localhost][port='29900'][dbname='postgres'][2024-11-11 11:04:13]: dump database dbname='postgres' successfully
gs_dump[user='omm'][localhost][port='29900'][dbname='postgres'][2024-11-11 11:04:13]: total time: 3261  ms
gs_dumpall[user='omm'][localhost][port='29900'][2024-11-11 11:04:13]: dumpall operation successful
gs_dumpall[user='omm'][localhost][port='29900'][2024-11-11 11:04:13]: total time: 3320  ms
```

- Export only global objects (roles and tablespaces).

```
gs_dumpall -U omm -f ./backup/bkp2.sql -p 29900 -g
gs_dumpall[user='omm'][localhost][port='29900'][2024-11-11 11:06:00]: dumpall operation successful
gs_dumpall[user='omm'][localhost][port='29900'][2024-11-11 11:06:00]: total time: 45  ms
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Encrypt the exported files using AES-128.

  Replace ******** with the actual password.

```
gs_dumpall -U omm -f ./backup/bkp2.sql -p 29900 --with-encryption=AES128 --with-key='********'
gs_dump[user='omm'][localhost][port='29900'][dbname='postgres'][2024-11-11 11:07:20]: The total objects number is 459.
gs_dump[user='omm'][localhost][port='29900'][dbname='postgres'][2024-11-11 11:07:20]: [100.00%] 459 objects have been dumped.
gs_dump[user='omm'][localhost][port='29900'][dbname='postgres'][2024-11-11 11:07:21]: dump database dbname='postgres' successfully
gs_dump[user='omm'][localhost][port='29900'][dbname='postgres'][2024-11-11 11:07:21]: total time: 3518  ms
gs_dumpall[user='omm'][localhost][port='29900'][2024-11-11 11:07:21]: dumpall operation successful
gs_dumpall[user='omm'][localhost][port='29900'][2024-11-11 11:07:21]: total time: 3764  ms
```

Related Commands

gs_dump for Exporting Database Information and **gs_restore for Importing Data**

4.3.4 gs_restore for Importing Data

Description

gs_restore, provided by GaussDB, is used to import data that was exported using gs_dump. It can also be used to import files exported by gs_dump.

It has the following functions:

- Imports data to the database.

  If a database is specified, data is imported to the database. For parallel import, the password for connecting to the database is required.

- Imports data to an archive file.

  If the **-l** parameter is specified, an archive file containing a brief summary of the data is generated.

gs_restore supports SSL-encrypted communication. The method is the same as that of using gsql.

Prerequisites

Before using gs_restore, ensure that the gs_restore version is consistent with the gs_dump version and database version.

Precautions

- When gs_restore is used to restore data to a new database, the value of **rowid** in the hidden system column can be changed, the value of the sequence associated with the hidden column can be changed, and the user column data of the rowid type remains unchanged.

Syntax

```
gs_restore [OPTION]... FILE
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The **FILE** does not have a short or long option. It is used to specify the location for the archive files.
- The **dbname** or **-l** option is required as prerequisites. Users cannot enter **dbname** and **-l** parameters at the same time.
- gs_restore incrementally imports data by default. To prevent data exception caused by consecutive imports, use the **-e** and **-c** parameters for each import. **-c** indicates that the database objects that already exist in the database to be restored are cleared (deleted) before the database objects are rebuilt. **-e** indicates that if an error occurs when an SQL statement is sent to the database, the system exits. By default, the system continues to import data and displays a series of error information after the import is complete.
- If the owner of the schema object has the OPRADMIN system permission, the initial user is required for the import.
- During data import, if invalid characters are found and do not need to be transcoded, you can set the database compatibility parameter **copy_special_character_version** to **'no_error'** to import the data. Otherwise, an error is reported but the import is not interrupted.
- The current design of gs_restore is incompatible with separation of duties. In separation of duties, gs_restore can be used only by the initial user. The file to be imported must also be executed by the initial user.

Parameters

Common parameters:

- -d, --dbname=NAME

  Connects to the **dbname** database and imports data to the database.

- -f, --file=FILENAME

  Specifies the output file for the generated archive, or the output file in the list specified using **-l**.

  The default is the standard output.

  NOTE

  -f cannot be used with **-d**.

- -F, --format=c|d|t

  Specifies the format of the archive. The format does not need to be specified because the gs_restore determines the format automatically.

  Value range:

  – **c/custom**: The archive form is the customized format in gs_dump.
  – **d/directory**: The archive format is **directory**.
  – **t/tar**: The archive format is **tar**.

- -l, --list

  Lists the formats of the archive. The operation output can be used for the input of the **-L** option.

  NOTE

  If filtering parameters, such as **-n** or **-t**, are used together with **-l**, they will restrict the listed items.

- -v, --verbose

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Specifies the verbose mode.

- -V, --version

  Prints the gs_restore version and exits.

- -?, --help

  Displays help information about gs_restore parameters and exits.

Import parameters:

- -a, --data-only

  Imports only the data, not the schema (data definition). gs_restore incrementally imports data.

- -c, --clean

  Cleans (deletes) existing database objects in the database to be restored before recreating them. If the target database does not contain the objects involved in the deletion operation, some promptive error information may be displayed.

  NOTE

  - None of the commands in the backup file for deleting database objects are cascading DROP commands.
  - At t1, database 1 is connected to export table 1 to the **backup01.dmp** file using the **-c** parameter. At t2, view 1 is created in database 1 and depends on table 1. At t3, **backup01.dmp** is restored to database 1 using the **-c** parameter of gs_restore. Non-cascading DROP is used. When table 1 is deleted, the error message "ERROR: table table cannot be dropped because other objects depend on it." is displayed, and when table 1 is created, the error message "ERROR: Relation "table1" already exists." is displayed. In this case, delete related dependent objects or execute the cascading DROP command after evaluating the impact.

- -C, --create

  Before data is imported to a database, CREATE DATABASE is used to create the database. (After this option is specified, the database specified by **-d** is only used to execute the **CREATE DATABASE** command, and all data is still imported to the created database.)

  NOTE

  This option is not supported in M-compatible databases. You must create a database in M-compatible mode on the target instance, export data from the source instance, and connect to the newly created M-compatible database on the target instance to import data.

- -e, --exit-on-error

  Exits if an error occurs when you send the SQL statement to the database. If you do not exit, the commands will still be sent and error information will be displayed when the import ends.

- -I, --index=NAME

  Imports only the definition of the specified index. Multiple indexes can be imported. Enter *-I index* multiple times to import multiple indexes.

  For example:

  ```
  gs_restore -h host_name -p port_number -d testdb -I Index1 -I Index2 backup/MPPDB_backup.tar
  ```

  In this example, **Index1** and **Index2** will be imported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- -j, --jobs=NUM

  Specifies the number of concurrent, the most time-consuming jobs of gs_restore (such as loading data, creating indexes, or creating constraints). This parameter can greatly reduce the time to import a large database to a server running on a multi-processor machine.

  Each task may be a process or a thread, which is determined by the OS. Each task is connected to the server separately.

  The optimal value for this option depends on the server hardware settings, the client, the network, the number of CPU cores, and disk settings. It is recommended that the parameter be set to the number of CPU cores on the server. In addition, a larger value can also lead to faster import in many cases. However, an overly large value will lead to performance deterioration because of thrashing.

  This option supports the customized archive format only. The input file must be a regular file (not the pipe file). This parameter can be ignored when you select the script method rather than connecting to a database server. In addition, multiple jobs cannot be used with the **--single-transaction** option.

  NOTE

  - This parameter applies to multi-table, multi-index, and multi-constraint scenarios. In practice, the number of created processes (or threads) is related to the number of tables, indexes, and constraints. The maximum number of concurrent jobs does not exceed the specified number of jobs.
  - The specified number of parallel tasks cannot exceed twice the number of CPU cores on the client. When the client runs in a container, the number of CPUs obtained is the number of CPUs on the host, which may be greater than the number of CPUs that can be used by the container. You are advised to set the degree of parallelism to a value less than twice the number of CPUs that can be used by the client. Parallel import is implemented by concurrently executing multiple transactions through multiple processes. The actual degree of parallelism is limited by the thread pool model of the server and increases the pressure on the server. Set the degree of parallelism based on the actual needs. For example, the degree of parallelism cannot exceed the number of idle CPU cores on the client or the maximum number of connections on the server.

- -L, --use-list=FILENAME

  Imports only archive elements that are listed in **list-file** and imports them in the order that they appear in the file. If filtering parameters, such as **-n** or **-t**, are used with **-L**, they will further limit the items to be imported.

  list-file is normally created by editing the output of a previous **-l** parameter. File lines can be modified or removed, and can also be commented out by placing a semicolon (;) at the beginning of the row. An example is provided in this document.

- -n, --schema=NAME

  Restores only objects that are listed in schemas.

  This parameter can be used with the **-t** parameter to import a specific table.

  Entering **-n** *schemaname* multiple times can import multiple schemas.

  For example:

  ```
  gs_restore -h host_name -p port_number -d testdb -n sch1 -n sch2 backup/MPPDB_backup.tar
  ```

  In this example, **sch1** and **sch2** will be imported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> GaussDB automatically converts uppercase letters in object names to lowercase letters. If a schema name contains uppercase letters, add extra quotation marks, for example, -n '"Sch1"' or -n "\"Sch1\"".

- -O, --no-owner

  Do not output commands to set ownership of objects to match the original database. By default, gs_restore issues the ALTER OWNER or SET SESSION AUTHORIZATION statement to set ownership of created schema elements. Unless a system administrator or the user who has all the objects in the script initially connects to the database. Otherwise, the statement will fail. Any username can be used for the initial connection using **-O**, and this user will own all the created objects.

- -P, --function=NAME(args)

  Imports only listed functions. You need to correctly spell the function name and the parameter based on the contents of the dump file in which the function exists.

  Entering **-P** alone means to import all function-name(args) functions in a file. Entering **-P** with **-n** means to import the function-name(args) functions in a specified schema. Entering **-P** multiple times and using **-n** once means that all imported functions are in the **-n** schema by default.

  You can enter **-n schema-name -P 'function-name(args)'** multiple times to import functions in specified schemas.

  For example:

  ```
  ./gs_restore -h host_name -p port_number -d testdb -n test1 -P 'Func1(integer)' -n test2 -P 'Func2(integer)' backup/MPPDB_backup.tar
  ```

  In this example, both **Func1 (i integer)** in the **test1** schema and **Func2 (j integer)** in the **test2** schema will be imported.

- -s, --schema-only

  Imports only schemas (data definitions), instead of data (table content). The current sequence value will not be imported.

- -S, --sysadmin=NAME

  Reserved for function extension. The option is not recommended.

- -t, --table=NAME

  Imports only listed table definitions or data, or both. This parameter can be used with the **-n** parameter to specify a table object in a schema. When **-n** is not entered, the default schema is **PUBLIC**. Entering **-n** *schemaname* **-t** *tablename* multiple times can import multiple tables in a specified schema.

  For example:

  Import **table1** in the **PUBLIC** schema.

  ```
  gs_restore -h host_name -p port_number -d testdb -t table1 backup/MPPDB_backup.tar
  ```

  Import **test1** in the **test1** schema and **test2** in the **test2** schema.

  ```
  gs_restore -h host_name -p port_number -d testdb -n test1 -t test1 -n test2 -t test2 backup/MPPDB_backup.tar
  ```

  Import **table1** in the **PUBLIC** schema and **test1** in the **test1** schema.

  ```
  gs_restore -h host_name -p port_number -d testdb -n PUBLIC -t table1 -n test1 -t table1 backup/MPPDB_backup.tar
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- **-t** does not support the *schema_name.table_name* input format. If this format is specified, no error is reported but the setting does not take effect.
- When **-t** is specified, gs_restore does not import any other database objects that are attached to the selected table. Therefore, there is no guarantee that the results of a specific-table dump can be automatically imported to an empty database.
- **-t tablename** does not import trigger information from a table.

- -T, --trigger=NAME

  This parameter is reserved for extension.

- -x, --no-privileges/--no-acl

  Prevents the import of access permissions (**GRANT**/**REVOKE** commands).

- -1, --single-transaction

  Executes the import as a separate transaction.

  This option ensures that either all the commands are completed successfully, or no application is changed. This option means **--exit-on-error**.

- --disable-triggers

  Reserved for function extension. The option is not recommended.

- --log-level=LOG|DEBUG|DEBUG1|DEBUG2|DEBUG3

  Sets the log level of gs_restore. **LOG** records normal execution information, and **DEBUG** records some debugging information. A larger **DEBUG** value indicates a higher log level and more debugging details.

- --no-data-for-failed-tables

  By default, table data will be imported even if the statement to create a table fails (for example, the table exists). Data in such table is skipped using this parameter. This operation is useful if the target database already contains the desired table contents.

  This parameter takes effect only when you import data directly to a database, not when you output SQL scripts.

- --no-security-labels

  Reserved for function extension. The option is not recommended.

- --no-tablespaces

  Does not select any tablespaces. All the objects will be created during the import process, no matter which tablespace is selected when using this option.

- --no-transaction

  By default, statements such as CREATE, ALTER... OWNER TO, SET SESSION AUTHORIZATION, and RESET SESSION AUTHORIZATION are executed in a separate transaction block. If this option is enabled, these statements are executed directly without being wrapped in BEGIN/COMMIT.

- --section=SECTION

  Imports the listed sections (such as pre-data, data, or post-data).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- --use-set-session-authorization

  This option is used for backing up the plain-text format.

  Generates the **SET SESSION AUTHORIZATION** statement as the output instead of the **ALTER OWNER** statement to determine object ownership. This parameter makes dump more standards-compatible. If the records of objects in exported files are referenced, import may fail. Only administrators can use the **SET SESSION AUTHORIZATION** statement to dump data, and the administrators must manually change and verify the passwords of exported files by referencing the **SET SESSION AUTHORIZATION** statement before import. The **ALTER OWNER** statement requires lower permissions.

- --pipeline

  Uses a pipe to transmit the password. This parameter cannot be used on devices.

NOTICE

- If any local additions need to add to the template1 database during the installation, restore the output of gs_restore into an empty database with caution. Otherwise, you are likely to obtain errors due to duplicate definitions of the added objects. To create an empty database without any local additions, copy data from template0 rather than template1. For example:

```
gaussdb=# CREATE DATABASE foo WITH TEMPLATE template0;
```

NOTE

- The **-d/--dbname** and **-f/--file** options do not coexist.
- The **-s/--schema-only** and **-a/--data-only** options do not coexist.
- The **-c/--clean** and **-a/--data-only** options do not coexist.
- When the **--single-transaction** option is used, **-j/--jobs** must be a single job.
- **--role** must be used with **--rolepassword**.

Connection parameters:

- -h, --host=HOSTNAME

  Specifies the host name. If the value begins with a slash (/), it is used as the UDS directory. The default value is taken from the *PGHOST* environment variable. If it is not set, a UDS connection is attempted.

  This parameter is used only for defining names of the hosts outside a cluster. For the localhost in the cluster, you can use IPv4 address **127.0.0.1** or IPv6 address **::1**.

  Environment variable: *PGHOST*

- -p, --port=PORT

  Specifies the TCP port listened on by the server or the local UDS file name extension to ensure a correct connection. The default value is the *PGPORT* environment variable.

  If the thread pool is enabled, you are advised to use the pooler port, that is, the listening port number plus 1.

  Environment variable: *PGPORT*

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- -U, --username=NAME

  Specifies the name of the connected user. The initial user cannot be used for cross-node execution.

  Environment variable: *PGUSER*

  > NOTE
  >
  > Before exporting and importing data as a common user, grant the permission to create schemas in the corresponding database to the common user, for example, **grant create on database db**_*xxx* **to user**_*xxx*.

- -w, --no-password

  Never issues a password prompt. The connection attempt fails if the server requires password for authentication and the password is not provided in other ways. This parameter is useful in batch jobs and scripts in which no user password is required.

- -W, --password=PASSWORD

  Specifies the user password for connection. If the authentication policy of the host is **trust**, the password of the system administrator is not verified. That is, you do not need to enter the **-W** parameter. If this parameter is not specified and you are not a system administrator, the system prompts you to enter the password in interactive mode. To ensure system security, you are advised to enter the password in interactive mode.

- --role=ROLENAME

  Specifies a role name for the import operation. If this parameter is selected, the SET ROLE statement will be issued after gs_restore connects to the database. It is useful when the authenticated user (specified by **-U**) lacks the permissions required by gs_restore. This parameter allows the user to switch to a role with the required permissions. Some installations have a policy against logging in directly as the initial user. This parameter allows data to be imported without violating the policy.

- --rolepassword=ROLEPASSWORD

  Specifies a role password for the import operation.

Examples

gs_restore is used to import the files exported by gs_dump.

- Example 1: Execute the gs_restore tool to import the exported **MPPDB_backup.dmp** file (custom format) to the **testdb** database.

  ```
  gs_restore backup/MPPDB_backup.dmp -p 8000 -d testdb
  restore operation successful
  total time: 13053  ms
  ```

- Example 2: Execute the gs_restore tool to import the exported **MPPDB_backup.tar** file (.tar format) to the **testdb** database.

  ```
  gs_restore backup/MPPDB_backup.tar -p 8000 -d testdb
  restore operation successful
  total time: 21203  ms
  ```

- Example 3: Execute the gs_restore tool to import the exported **MPPDB_backup** file (directory format) to the **testdb** database.

  ```
  gs_restore backup/MPPDB_backup -p 8000 -d testdb
  restore operation successful
  total time: 21003  ms
  ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Example 4: Execute the gs_restore tool and import the **MPPDB_backup.dmp** file (in custom format). Specifically, import all the object definitions and data in the **PUBLIC** schema. Existing objects are deleted from the target database before the import. If an existing object references to an object in another schema, you need to manually delete the referenced object first.

```
gs_restore backup/MPPDB_backup.dmp -p 8000 -d testdb -e -c -n PUBLIC
Error while PROCESSING TOC:
Error from TOC entry 313; 1259 337399 TABLE table1 gaussdba
could not execute query: ERROR:  cannot drop table table1 because other objects depend on it
DETAIL:  view t1.v1 depends on table table1
HINT:  Use DROP ... CASCADE to drop the dependent objects too.
   Command was: DROP TABLE IF EXISTS public.table1;
```

  Manually delete the referenced object and create it again after the import is complete.

```
gs_restore backup/MPPDB_backup.dmp -p 8000 -d testdb -e -c -n PUBLIC
restore operation successful
total time: 2203  ms
```

- Example 5: Execute the gs_restore tool and import the **MPPDB_backup.dmp** file (in custom format). Specifically, import only the definition of **table1** in the **PUBLIC** schema.

```
gs_restore backup/MPPDB_backup.dmp -p 8000 -d testdb -e -c -s -n PUBLIC -t table1
restore operation successful
total time: 21000  ms
```

- Example 6: Execute the gs_restore tool and import the **MPPDB_backup.dmp** file (in custom format). Specifically, import only the data of **table1** in the **PUBLIC** schema.

```
gs_restore backup/MPPDB_backup.dmp -p 8000 -d testdb -e -a -n PUBLIC -t table1
restore operation successful
total time: 20203  ms
```

- Special case: Execute the gsql tool. Import the **MPPDB_backup.sql** file in the plain-text format exported using gs_dump or gs_dumpall to the **testdb** database.

```
-- In the following example, -f specifies an absolute file path, 8000 specifies the port number of the database server, and testdb specifies the name of the database to be accessed.
gsql -d testdb -p 8000 -f /home/omm/test/MPPDB_backup.sql
SET
SET
SET
SET
SET
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
CREATE INDEX
CREATE INDEX
CREATE INDEX
SET
CREATE INDEX
REVOKE
REVOKE
GRANT
GRANT
total time: 30476  ms
```

Related Commands

gs_dump for Exporting Database Information and **gs_dumpall for Exporting All Database Information**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

5 Log Reference

5.1 Log Overview

During database running, a large number of logs are generated, including write-ahead logs (WALs, also called Xlogs) for ensuring database security and reliability and run logs and operation logs for daily database maintenance. If the database is faulty, you can refer to these logs to locate the fault and restore the database.

Log Type

The following table describes details about log types.

Table 5-1 Log types

Type	Description
System log	Logs generated during database running. They are used to record abnormal process information.
Operation log	Logs generated when a client tool (such as gs_guc) is operating databases.
Audit log	Logs used to record some of the database user operations after the database audit function is enabled.
WAL	Logs used to restore a damaged database. They are also called redo logs. You are advised to routinely back up WALs.
Memory log	Logs used to save the memory context statistics when the dynamic memory usage of the database process exceeds 90%.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Log List

Table 5-2 Log list

Log Category	Log Name	Log Path
System log	gaussdb-*Creation time*.log	*$GAUSSLOG*/gs_log
	system_alarm-*Creation time*-current.log	*$GAUSSLOG*/gs_log
	gtm-*Creation time*.log	*$GAUSSLOG*/gs_log/gtm
	cm_agent-*Creation time*-current.log	*$GAUSSLOG*/cm/cm_agent
	system_alarm-*Creation time*-current.log	*$GAUSSLOG*/cm/cm_agent
	system_call-current.log	*$GAUSSLOG*/cm/cm_agent
	system_stat-current.log	*$GAUSSLOG*/cm/cm_agent
	cm_server-*Creation time*-current.log	*$GAUSSLOG*/cm/cm_server
	key_event-*Creation_time*-current.log	*$GAUSSLOG*/cm/cm_server
	system_alarm-*Creation time*-current.log	*$GAUSSLOG*/cm/cm_server
	cm_ctl-*Creation time*-current.log	*$GAUSSLOG*/cm/cm_ctl
	om_monitor-*Creation time*-current.log	*$GAUSSLOG*/cm/om_monitor
	system_alarm-*Creation time*-current.log	*$GAUSSLOG*/cm/om_monitor
Operation log	gs_install-*Creation time*.log	*$GAUSSLOG*/om
	gs_preinstall-*Creation time*.log	*$GAUSSLOG*/om
	gs_om-*Creation time*.log	*$GAUSSLOG*/om
	gs_expand-*Creation time*.log	*$GAUSSLOG*/om
	gs_replace-*Creation time*.log	*$GAUSSLOG*/om
	gs_local-*Creation time*.log	*$GAUSSLOG*/om

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Log Category	Log Name	Log Path
	checkRunStatus-*Creation time*.log	*$GAUSSLOG*/om
	JsonToDbclusterInfo-*Creation time*.log	*$GAUSSLOG*/om
	killall_log-*Creation time*.log	*$GAUSSLOG*/om
	uploader-*Creation time*.log	*$GAUSSLOG*/om
	cmd_sender-*Creation time*.log	*$GAUSSLOG*/om
	gs_initdb-*Creation time*-current.log	*$GAUSSLOG*/bin/gs_initdb
	gs_guc-*Creation time*-current.log	*$GAUSSLOG*/bin/gs_guc
	gs_cgroup-*Creation time*.log	*$GAUSSLOG*/bin/gs_cgroup
	gs_clean-*Creation time*-current.log	*$GAUSSLOG*/bin/gs_clean
	gs_ctl-*Creation time*-current.log	*$GAUSSLOG*/bin/gs_ctl
	gs_encrypt-*Creation time*-current.log	*$GAUSSLOG*/bin/gs_encrypt
	gs_initgtm-*Creation time*-current.log	*$GAUSSLOG*/bin/gs_initgtm
	gs_obs.run.log	*$GAUSSLOG*/bin/gs_obs
	gtm_ctl-*Creation time*-current.log	*$GAUSSLOG*/bin/gtm_ctl
	gs_redis-*Creation time*-current.log	*$GAUSSLOG*/bin/gs_redis
Audit log	–	The default path is *$GAUSSLOG*/**pg_audit**. (If the OM tool is used for cluster deployment, the audit log path is *$GAUSSLOG*/**pg_audit/***instance name*.) The GUC parameter **audit_directory** can be used to configure the path.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Log Category	Log Name	Log Path
WAL	pg_xlog	*Data directory*/**pg_xlog**, for example, **/gaussdb/data/data_dn/pg_xlog** in a DN instance. **/gaussdb/data/data_dn** indicates the data directory of the cluster node.
Memory log	mem_log-*Creation time*.log	*$GAUSSLOG*/mem_log

5.2 System Logs

System logs include those generated by CNs and DNs, GTMs, and CMs during cluster running, and those generated when the cluster is deployed. If an error occurs during cluster running, you can locate the cause and troubleshoot the error based on system logs.

Log Storage Directory

Run logs of CNs, DNs, and GTMs are stored in the corresponding folders in the *$GAUSSLOG*/**gs_log** directory.

Logs of CMs (including the CM Agent, CM Server, cm_ctl, OM Monitor, and ETCD) are stored in the corresponding folders in the *$GAUSSLOG*/**cm** directory.

Logs generated during OM cluster installation and uninstallation are stored in the *$GAUSSLOG*/**om** directory.

Log Naming Rules

- The name format of CNs and DNs run logs is:

  gaussdb-*Creation time***.log**

  By default, a new log file is generated at 0:00 every day, or when the latest log file exceeds 16 MB or a cluster (CNs and DNs) is restarted.

  system_alarm-*Creation time***-current.log**
- The name format of GTM run logs is:

  gtm-*Creation time***.log**

  By default, a new log file is generated when the latest log file exceeds 16 MB or a GTM is restarted.
- The name formats of CM logs are:
  - CM Agent logs: **cm_agent-***Creation time***.log**, **cm_agent-***Creation time***-current.log**, **system_alarm-***Creation time***-current.log**, **system_call-current.log**, **system_stat-current.log**, and **gaussdb-***Creation time*_**000000.log**.
  - CM Server logs: **cm_server-***Creation time***.log**, **cm_server-***Creation time***-current.log**, **key_event-***Creation time***-current.log**, and **system_alarm-***Creation time***-current.log**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- cm_ctl logs: **cm_ctl-***Creation time***-current.log**.
- OM Monitor logs: **om_monitor-***Creation time***.log**, **om_monitor-***Creation time***-current.log**, and **system_alarm-***Creation time***-current.log**.

Logs whose names do not contain **current** are historical log files. Logs whose names contain **current** are current log files. When a process is called for the first time, a log file whose name contains **current** is created. If the size of this file exceeds 16 MB, the file is renamed in the historical log file name format, and a new log file is generated at the current time point.

NOTE

- Logs with prefixes cm_agent, cm_server, cm_ctl, and om_monitor are run logs of CM Agent, CM Server, cm_ctl, and OM Monitor respectively.
- Logs with the prefix system_alarm record the alarm reporting and clearance of cluster running exceptions in the CM Agent, CM Server, and OM Monitor.
- Logs with the prefix key_event are key event logs of the CM Server that record information about CM Server arbitration.
- Logs with the prefix system_call record the calling of tool commands when the CM Agent is running.
- Logs with the prefix system_stat record the disk I/O performance of the current system when the CM Agent is running.
- Logs with the prefix gaussdb are run logs when the fenced mode is enabled.
- Log names depend on the system timestamp. The database compresses old logs based on the time. If the system time is reversed, the log files that are being written may be compressed in advance. As a result, the log files cannot be written. In this case, you need to clear the logs at future time points for restoration.

Log Content Description

- The default format of each line of log content for CNs and DNs is as follows:

  Date+Time+Node name+Username+Database name+IP address+Session ID +Transaction ID+Application name+Log level+Log content

Table 5-3 Log parameters on CNs and DNs

Column	Description
Date	Date when a log is generated. The format is yyyy-mm-dd.
Time	Time when a log is generated. The format is hh:mm:ss:ms.
Node name	Node to which an error is reported.
Username	Name of the user who triggers log generation.
Database name	Name of the database that triggers log generation.
IP address	IP address of the client that triggers log generation.
Thread ID	Thread ID.
Session ID	ID of the session that triggers log generation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Transaction ID	Transaction ID (**0** indicates that no transaction ID is assigned).
Thread name	Thread name.
Query ID	**query_id** indicates the ID of a query initiated by a user which is recorded in the background.
Module name	Module name.
Log level	Different log levels, such as **FATAL**, **ERROR**, and **LOG**, indicate different severities.
Log content	Log content.

- Content of a line in a GTM log:

  Log line No.+Thread ID+Date+Time+Log level+Log content

  Table 5-4 GTM log parameters

Column	Description
Log line No.	Number of the line where the log content is located in the log, because the current log may contain multiple lines.
Thread ID	ID of the thread that generates the log.
Date	Date when a log is generated. The format is yyyy-mm-dd.
Time	Time when a log is generated. The format is hh:mm:ss:ms.
Log level	Different log levels, such as **FATAL**, **ERROR**, and **LOG**, indicate different severities.
Log content	Log content.

- The content of a line in a CN or DN system_alarm log: Alarm ID+Alarm name +Alarm severity+Host scope list+Alarm source tag+Alarm operation type +Alarm description+Clearance type+Start timestamp+End timestamp. For details about log parameters, see **Table 5-5**.

  Table 5-5 system_alarm log parameters of the CN, DN, CM Agent, CM Server, and OM Monitor

Field Name	Field Name	Description
id	Alarm ID	Alarm ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Field Name	Field Name	Description
name	Alarm name	Alarm name.
level	Alarm severity	Different alarm severities indicate different severities, including **critical**, **major**, **minor**, and **notice**.
scope	Host scope list	Host scope list. Alarm platform that uses NewSQL. The value is fixed to **newsql**.
source_tag	Alarm source tag	Alarm source tag.
op_type	Alarm operation type	Operation type of an alarm. • **firing** indicates that an alarm is reported. If the value is **firing**, the log contains the alarm description and clearance type. • **resolved** indicates that the alarm is cleared. If the value is **resolved**, the log does not contain the alarm description and clearance type.
details	Alarm description	Detailed information about an alarm.
clear_type	Alarm clearance type	Type of alarm clearance. The options are **ADAC** and **ADMC**. • **ADAC**: The system can automatically detect the occurrence and rectification of the fault. • **ADMC**: The system can automatically detect the occurrence of the fault, but cannot detect the rectification of the fault.
start_timestamp	Start timestamp	Time when an alarm starts. • When the alarm is reported, **start_timestamp** has a value, and the value of **end_timestamp** is **0**. • When the alarm is cleared, the value of **start_timestamp** is **0**, and **end_timestamp** has a value.
end_timestamp	End timestamp	Time when an alarm ends.

- The content of the CM Agent, CM Server, cm_ctl, and OM Monitor logs is as follows:
  - The content of a line in a run log of the CM Agent, CM Server, cm_ctl, or OM Monitor: Date+Time+Thread ID+Thread name+Log level+Log content.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The content of a line in a system_alarm log of the CM Agent, CM Server, or OM Monitor: Alarm ID+Alarm name+Alarm severity+Host scope list +Alarm source tag+Alarm operation type+Alarm description+Clearance type+Start timestamp+End timestamp.
- The content of a line in a key_event log of the CM Server: Date+Time +Thread ID+Thread name+Key event type+Arbitration object instance ID +Arbitration details.
- Most contents in a system_call log of the CM Agent: Date+Time+Node name+Username+Database name+IP address+Session ID+Transaction ID +Application name+Log level+Log content. For details about log parameters, see **Table 5-3**.
- The content of a system_stat log of the CM Agent: Time+OS+CPU information+Disk read/write status.

Table 5-6 Run log parameters of the CM Agent, CM Server, cm_ctl, and OM Monitor

Column	Description
Date	Date when a log is generated. The format is yyyy-mm-dd.
Time	Time when a log is generated. The format is hh:mm:ss:ms.
Thread ID	ID of the thread that generates the current log.
Thread name	Name of the thread that generates the current log. (This parameter does not exist in cm_ctl and OM Monitor run logs.)
Log level	Different log levels, such as **LOG**, **WARNING**, **ERROR**, and **FATAL**, indicate different severities.
Log content	Log content.

Table 5-7 key_event log parameters of the CM Server

Column	Description
Date	Date when a log is generated. The format is yyyy-mm-dd.
Time	Time when a log is generated. The format is hh:mm:ss:ms.
Thread ID	ID of the thread that generates the current log.
Thread name	Name of the thread that generates the current log.
Key event type	Type of a key event. The format is [KeyEvent: xxxx].

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Key event object instance ID	Object of a key event. The format is [Instance: xxxx].
Key event details	Details of a key event. The format is [Details: xxxx].

Table 5-8 system_stat log parameters of the CM Agent

Column	Description
Time	Time when a log is generated, in CST format.
OS	OS of a host.
CPU information	CPU usage. It includes the following information: ● **%user**: percentage of the time when the CPU is in user mode. ● **%nice**: percentage of the time when the CPU is in user mode with the NICE value. ● **%system**: percentage of the time when the CPU is in system mode. ● **%iowait**: percentage of the time that the CPU waits for input and output completion. ● **%steal**: percentage of the time that the virtual CPU waits involuntarily for the management program maintaining another virtual processor. ● **%idle**: percentage of CPU idle time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Disk read/write	Disk I/O information. It includes the following information: • **rrqm/s**: number of read requests merged per second, that is, the number of times that read requests are merged into larger blocks. • **wrqm/s**:number of write requests merged per second, that is, the number of times that write requests are merged into larger blocks. • **r/s**: number of read requests completed per second. • **w/s**: number of write requests completed per second. • **rMB/s**: amount of data read per second. The unit is MB/s. • **wMB/s**: amount of data written per second. The unit is MB/s. • **r_await**: average waiting time of each read request, in milliseconds. • **w_await**: average waiting time of each write request, in milliseconds. • **%util**: disk usage, that is, the percentage of the time that the disk is occupied by disk I/O operations. • **%rrqm**: percentage of read requests that are merged before being sent to the device. • **%wrqm**: percentage of write requests that are merged before they are sent to the device. • **rareq-sz**: average size of read requests sent to the device, in KB. • **wareq-sz**: average size of write requests sent to the device, in KB. • **d/s**: number of discard requests processed by the device per second. • **dMB/s**: number of sectors discarded by the device per second. • **drqm/s**: number of discard requests merged into the device queue per second. • **%drqm**: percentage of discard requests that are merged before being sent to the device. • **d_await**: average time for sending discard requests to the device to be served. • **dareq-sz**: average size of discard requests sent to the device, in KB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
	• **f/s**: number of refresh requests processed by the device per second (after merge). • **f_await**: average time for sending refresh requests to the device to be served. • **aqu-sz**: average queue length of requests sent to the device.

Log Examples

- CNs and DNs
  - A typical run log is as follows:
    ```
    2023-09-14 21:28:52.647 dn_6001 rdsAdmin postgres localhost 139983342597888 0[0:0#0] 0 dn_6001 0 [BACKEND] LOG:  will update control file (create checkpoint), shutdown:0
    ```
  - A typical system_alarm log is as follows:
    ```
    {"id":"0000001078919222","name":"AbnormalDataInstConnAuthMethod","level":"critical","scope":["newsql"],"source_tag":"szvphis5033219-cn_5001","op_type":"firing","details":"Data instance cn_5001 remote connection trust method error","clear_type":"ADAC","start_timestamp":1694140757589,"end_timestamp":0}
    ```
- A typical GTM run log is as follows:
  ```
  1:140027660965632:2023-09-05 09:38:51.267 -LOG:  switched to running as GTM-primary.
  ```
- cm_agent
  - A typical run log is as follows:
    ```
    2023-09-05 05:59:59.845 tid=13690 CheckSysStatus ASYN LOG: [MEMORY STAT] memory system usage: 91.60%, application usage: 97.42%.
    ```
  - A typical system_alarm log is as follows:
    ```
    {"id":"0000001078919202","name":"DatanodeFailOver","level":"major","scope":["newsql"],"source_tag":"pekphis351066-dn_6001","op_type":"firing","details":"Datanode standby dn_6001 fail over","clear_type":"ADMC","start_timestamp":1693035351709,"end_timestamp":0}
    ```
  - A typical system_call log is as follows:
    ```
    2023-09-04 15:37:07.509 dn_6001_6002_6003 [unknown] [unknown] localhost 140252967524864 0[0:0#0] 0 [unknown] 0 [BACKEND] LOG:  create gaussdb state file success: db state(STARTING_STATE), server mode(Pending), connection index(1)
    ```
  - A typical system_stat log is as follows:
    ```
    Fri Aug 25 16:58:19 CST 2023
    Linux 3.10.0-862.14.1.5.h687.eulerosv2r7.x86_64 (pekphis351066)         08/25/2023      _x86_64_        (8 CPU)

    avg-cpu:  %user   %nice %system %iowait  %steal   %idle
              18.02    0.00   10.07    0.41    0.00   71.50

    Device:         rrqm/s   wrqm/s     r/s     w/s    rMB/s    wMB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
    vda               0.01     7.89    0.08    3.67     0.00     0.05    26.96     0.01    2.57    3.41    2.55    0.18   0.07
    vdb               0.04    51.39    0.53  103.88     0.09     0.65    14.63     0.09    0.84    4.85    0.82    0.39   4.06
    ```
  - A typical gaussdb log in the **gs_log** directory is as follows:
    ```
    [UDF] LOG:  Fenced master process start OK.
    ```
- cm_server
  - A typical run log is as follows:
    ```
    2023-09-04 16:03:50.693 tid=95109 MAIN LOG: process startup packet, remote_type 7, nodeid 1, node name pekphis351066, postmaster is false.
    ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - A typical system_alarm log is as follows:
    ```
    {"id":"0000001078919258","name":"ServerSwitchOver","level":"minor","scope":["newsql"],"source_tag":"pekphis351066-server_1","op_type":"firing","details":"Server standby server_1 switch over","clear_type":"ADMC","start_timestamp":1693031871104,"end_timestamp":0}
    ```
  - A typical key_event log is as follows:
    ```
    2023-09-04 15:45:10.468 tid=95440 AGENT_WORKER: [KeyEvent: KEY_EVENT_SWITCHOVER] [Instance: 6001] [Details: send switchover message, node=1, instance=6001]
    ```
- cm_ctl
  - A typical run log is as follows:
    ```
    2023-08-28 16:39:54.113 tid=72942  LOG: set cm_server.conf success.
    ```
- om_monitor
  - A typical run log is as follows: (The CM Agent startup check is complete, the files **cluster_manual_start**, **agent_config_file_r**, and **agent_binary_file_x** exist, the **config_change_flag** file does not exist, the CM Agent process did not exist, and the number of times that the CM Agent is started is 0.)
    ```
    2023-09-05 10:50:41.044 tid=100802  LOG: The CM Agent startup check is complete: cluster_manual_start=1, agent_config_file_r=1, agent_binary_file_x=1, config_change_flag=0, previous_status=0, start_count=0.
    ```
  - A typical system_alarm log is as follows:
    ```
    {"id":"0000001078919226","name":"AbnormalCMAProcess","level":"critical","scope":["newsql"],"source_tag":"pekphis351066-cma","op_type":"resolved","start_timestamp":0,"end_timestamp":1692953836127}
    ```

Log Discarding Mechanism

When an exception occurs during system log writing (for example, the log disk is faulty or overloaded), the system retries to write the log (the default retry timeout interval is 60 seconds). If the retry still fails, the system discards the log to prevent services from being affected by log writing blocking. When a log is discarded, the following alarm information is recorded in the OS log: "WARNING: logger pipe write failed, dropped a log message."

5.3 Operation Logs

Operation logs are generated when database tools are used by a database administrator or called by a cluster. If the cluster is faulty, you can backtrack user operations on the database and reproduce the fault based on the operation logs.

Log Storage Directory

The default paths are *$GAUSSLOG*/**om** and *$GAUSSLOG*/**bin**. If the environment variable *$GAUSSLOG* does not exist or its value is empty, the logs generated for a tool will be displayed, but not recorded in the log file of the tool.

Log Naming Rules

The log file name format is as follows:

- *tool name-log creation time*.**log**
- *tool name-log creation time*-**current.log**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

tool name-log creation time.**log** is a historical log file, and *tool name-log creation time*-**current.log** is a current log file.

If the size of a log file exceeds 16 MB, the next time the tool is called, the log file is renamed in the historical log file name format, and a new log file is generated at the current time point. For example, **gs_guc-2015-01-16_183728-current.log** is renamed as **gs_guc-2015-01-16_183728.log**, and **gs_guc-2015-01-17_142216-current.log** is generated.

Log Content Description

- Logs of OM tools, including gs_install, gs_preinstall, gs_om, gs_expand, gs_replace, gs_local, checkRunStatus (API for checking O&M operating status), JsonToDbclusterInfo (a tool used to convert cluster topology files in JSON format to XML files), killall_log, uploader (PSCP tool), and cmd_sender (PSSH tool). The content of each log line is as follows:

  Call chain ID+Date and time+Thread ID+Location where the log is reported +Thread name+Log level+[Step*]+Log content

  Table 5-9 Log parameters of tools or components contained in the OM

Column	Description
Call chain ID	ID of the call chain.
Date and time	Date and time when a log is generated. The format is yyyy-mm-dd hh:mm:ss.ms.
Thread ID	ID of the thread that generates the log.
Location where the log is reported	Number of the line where the log content is located in the executed file. The current log may contain multiple lines.
Thread name	Name of the thread that generates the log.
Log level	Different log levels, such as **LOG**, **DEBUG**, **WARNING**, and **ERROR**, indicate different severities.
Step*	ID of the step executed at the current time.
Log content	Log content.

- Content of a line in a gs_initdb log:

  Date+Time+Log level+Log content

  Table 5-10 gs_initdb log parameters

Column	Description
Date	Date when a log is generated. The format is yyyy-mm-dd.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Time	Time when a log is generated. The format is hh:mm:ss.
Log level	Different log levels, such as **LOG**, **WARNING**, **ERROR**, and **FATAL**, indicate different severities.
Log content	Log content.

- Content of a line in a gs_guc log:
  Date+Time+Log content

  Table 5-11 gs_guc log parameters

Column	Description
Date	Date when a log is generated. The format is yyyy-mm-dd.
Time	Time when a log is generated. The format is hh:mm:ss.
Log content	Log content.

- Content of a line in a gs_cgroup log:
  Call chain ID+Date and time+Thread ID+Location where the log is reported +Thread name+Log level+Log content

  Table 5-12 gs_cgroup log parameters

Column	Description
Call chain ID	ID of the call chain.
Date and time	Date and time when a log is generated. The format is yyyy-mm-dd hh:mm:ss.ms.
Thread ID	ID of the thread that generates the log.
Location where the log is reported	Number of the line where the log content is located in the executed file. The current log may contain multiple lines.
Thread name	Name of the thread that generates the log.
Log level	Different log levels, such as **LOG**, **DEBUG**, **WARNING**, and **ERROR**, indicate different severities.
Log content	Log content.

- Content of a line in a gs_clean log:
  Date+Time+Log content

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 5-13 gs_clean log parameters

Column	Description
Date	Date when a log is generated. The format is yyyy-mm-dd.
Time	Time when a log is generated. The format is hh:mm:ss.
Log content	Log content.

- Content of a line in a gs_ctl log:
  Date+Time+Thread ID+*node_name*+Thread name+Log content

  Table 5-14 gs_ctl log parameters

Column	Description
Date	Date when a log is generated. The format is yyyy-mm-dd.
Time	Time when a log is generated. The format is hh:mm:ss.ms.
Thread ID	ID of the thread that generates the log.
node_name	Instance name.
Thread name	Name of the thread that generates the log.
Log content	Log content.

- Content of a line in a gs_initgtm log:
  Date+Time+Log content

  Table 5-15 gs_initgtm log parameters

Column	Description
Date	Date when a log is generated. The format is yyyy-mm-dd.
Time	Time when a log is generated. The format is hh:mm:ss.
Log content	Log content.

- Content of a line in a gs_obs log:
  Date+Time+Log level+Thread ID+Log content

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 5-16 gs_obs log parameters

Column	Description
Date	Date when a log is generated. The format is yyyy-mm-dd.
Time	Time when a log is generated. The format is hh:mm:ss.ms.
Log level	Different log levels, such as **LOG**, **WARNING**, **ERROR**, and **FATAL**, indicate different severities.
Thread ID	ID of the thread that generates the log.
Log content	Log content.

- Content of a line in a gtm_ctl log:

  Date+Time+Log content

  Table 5-17 gtm_ctl log parameters

Column	Description
Date	Date when a log is generated. The format is yyyy-mm-dd.
Time	Time when a log is generated. The format is hh:mm:ss.
Log content	Log content.

- Content of a line in a gs_redis log:

  Date+Time+Timestamp+Thread name+Thread ID+Location where the log is reported+step[*]+Log content

  Table 5-18 gs_redis log parameters

Column	Description
Date	Date when a log is generated. The format is yyyy-mm-dd.
Time	Time when a log is generated. The format is hh:mm:ss:ms.
Timestamp	Timestamp of the current time.
Thread name	Name of the thread that generates the log.
Thread ID	ID of the thread that generates the log.
step[*]	ID of the step executed at the current time.
Log content	Log content.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Log Examples

- A typical OM tool log is as follows:
  ```
  [220822164bc111ee9466fa163e8dd000][2023-09-05 08:02:11.098083][11214]
  [InstallImpl.py(doStart:864)][gs_install][DEBUG][Step7]:Successfully started the cluster.
  ```
- A typical gs_initdb log is as follows:
  ```
  [2023-09-05 07:57:01]

  WARNING: enabling "trust" authentication for local connections
  You can change this by editing gs_hba.conf or using the option -A, or
  --auth-local and --auth-host, the next time you run gs_initdb.
  ```
- A typical gs_guc log is as follows:
  ```
  [2023-09-05 09:05:06]
  Begin to perform the total nodes: 3.
  Popen count is 3, Popen success count is 3, Popen failure count is 0.
  Begin to perform gs_guc for datanodes.
  [2023-09-05 09:05:07]
  expected instance path: [/var/lib/engine/data1/data/dn_6001/gaussdb.conf]
  expected instance path: [/var/lib/engine/data1/data/dn_6005/gaussdb.conf]
  gs_guc reload: key=value: [/var/lib/engine/data1/data/dn_6001/gaussdb.conf]
  server signaled
  gs_guc reload: key=value: [/var/lib/engine/data1/data/dn_6005/gaussdb.conf]
  server signaled

  Total instances: 2. Failed instances: 0.
  Success to perform gs_guc!

  Command count is 3, Command success count is 3, Command failure count is 0.

  Total instances: 6. Failed instances: 0.
  ALL: Success to perform gs_guc!
  ```
- A typical gs_cgroup log is as follows:
  ```
  [de86987e4bc011eeb3befa163e8dd000][2023-09-05 15:51:49.372997][947]
  [PreInstallUtility.py(setCgroup:1793)][gs_cgroup][DEBUG]:The gs_cgroup run with the following
  arguments: [/var/chroot/usr/local/package/dependency/bin/gs_cgroup -U Ruby --upgrade -c -H /var/
  chroot/usr/local/tool/Ruby ].
  '/sys/fs/cgroup/cpu,cpuacct/CM:Ruby' exists, omit to create this cgroup.
  '/sys/fs/cgroup/cpu,cpuacct/CM:Ruby' exists, omit to create this cgroup.
  ```
- A typical gs_clean log is as follows:
  ```
  2023-09-05 09:06:57  gs_clean: connected successfully
  ```
- A typical gs_ctl log is as follows:
  ```
  [2023-09-05 08:00:59.538][4053][dn_6004_6005_6006][gs_ctl]: finish fsync all files.
  ```
- A typical gs_initgtm log is as follows:
  ```
  [2023-09-05 09:05:06] gs_initgtm: input file location must be an absolute path.
  ```
- A typical gs_obs log is as follows:
  ```
  2025-01-21 11:06:42.867| INFO|[140565224322816]|Enter complete_multi_part_upload successfully !
  ```
- A typical gtm_ctl log is as follows:
  ```
  [2023-09-05 08:00:40]
   Set GTM syncmode succeeded.
  ```
- A typical gs_redis log is as follows:
  ```
  2023-09-05 08:53:01  1693903981339 worker 140463824475904
  [snapshot.snap_summary_stat_all_indexes] step[2]: END successfully
  ```

Maintenance Suggestion

You are advised to dump expired logs periodically to save disk space and prevent important logs from being lost.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

5.4 Audit Logs

CREATEROLE can use the audit logs to reproduce a series of events that cause faults in the database and identify unauthorized users, unauthorized operations, and the time when these operations are performed. After the audit function is enabled, a large number of audit logs will be generated, which occupy large storage space. You can customize an audit log maintenance policy based on the size of available storage space.

Log Storage Directory

Audit logs are stored in *$GAUSSLOG*/**pg_audit/** by default. If **om** is used for cluster deployment, they are stored in *$GAUSSLOG*/**pg_audit/***Instance name*. The log storage path can be specified by the GUC parameter **audit_directory**.

Log Query Methods

Only users with the AUDITADMIN permission can view audit records. The query methods are as follows:

- Query traditional audit records.
  ```
  gaussdb=# SELECT * FROM gs_query_audit('2021-03-03 21:30:00', '2021-03-03 22:00:00') where type = 'login_success' and username = 'user1';
  ```
- Query traditional audit records of all CNs.
  ```
  gaussdb=# SELECT * FROM gs_query_audit_dist('2021-03-03 21:30:00', '2021-03-03 22:00:00') where type = 'login_success' and username = 'user1';
  ```
- Query unified audit records.

```
gaussdb=# SELECT * FROM gs_query_unified_audit('2025-04-26 08:00:00','2025-04-26 17:00:00')where type = 'login_success' and username = 'user1';
```

The difference between the values of the **starttime** and **endtime** parameters indicates the query period, which can be any value ranging from 00:00:00 of the **starttime** parameter to 23:59:59 of the **endtime** parameter. Therefore, the **startime** and **endtime** parameters must be properly set to ensure that the required audit information is displayed.

Log Content Description

For details about the columns in the log, see the definitions of the gs_query_audit function in **Table 5-19**.

Table 5-19 Traditional audit log parameters

Parameter	Description
time	Operation time.
type	Operation type.
result	Operation result.
userid	User ID

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
username	Name of the user who performs the operation.
database	Database name.
client_conninfo	Client connection information.
object_name	Operation object name.
detail_info	Operation details.
node_name	Node name.
thread_id	Thread ID
local_port	Local port.
remote_port	Remote port.

For details about the columns in the log, see the definitions of the gs_query_unified_audit function in **Table 5-20**.

Table 5-20 Unified audit log parameters

Parameter	Description
time	Operation time.
type	Operation type.
result	Operation result.
userid	User ID.
username	Name of the user who performs the operation.
database	Database name.
client_conninfo	Client connection information.
object_name	Operation object name.
detail_info	Operation details.
node_name	Node name.
thread_id	Thread ID.
local_port	Local port.
remote_port	Remote port.
policy_id	Unified audit policy ID.
unified_audit_type	Unified audit policy type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
unified_audit_policy	Unified audit policy information.

5.5 WALs

In a system using Write-ahead logs (WALs or Xlogs), all data file modifications are written to a log before they are applied. That is, corresponding logs must be written into a permanent device before a data file is modified. You can use WALs to restore the cluster if the system crashes. WALs can be recorded at different levels, which are specified by the **wal_level** parameter.

Log Running Mechanism

When GaussDB is started, a memory segment is allocated to cache logs generated during database running. The buffer size is specified by the **wal_buffers** parameter. To ensure transaction persistence, logs in **wal_buffers** must be flushed to disks.

WALs are written to disks in the following scenarios:

- Commit a transaction using COMMIT. The system calls the fdatasync function to write WAL data in **wal_buffers** to WAL files. (**wal_sync_method** specifies the function called during commit. The default value is **fdatasync**.)
- WAL writer periodically writes data. The WAL writer process periodically writes WAL data in **wal_buffers** to WAL files. The **wal_writer_delay** parameter specifies the interval at which the WAL writer process works. The default value is **200**, in ms. A longer delay might lead to insufficient WAL buffer and a shorter delay leads to continuously writing of the WALs, thereby increasing the load of disk I/O.

Log Storage Directory

The path is *Data directory*/**pg_xlog**.

- The WAL storage path of CNs is **/***<CN data directory>*/**pg_xlog**. The default size of a WAL file is 16 MB, which is specified by the **wal_segment_size** parameter. Generally, you do not need to change the size.

```
gaussdb=# SHOW wal_segment_size;
wal_segment_size
-----------------
16MB
(1 row)
```

- The following uses one DN as an example. You can run the following command to query the data directory: **/opt/gaussdb/data/dn/db_6001** indicates the data directory of the cluster node. Xlogs on the current node are stored in the **/opt/gaussdb/data/dn/db_6001/pg_xlog** directory.

```
$ cm_ctl query -Cvd
[    CMServer State    ]
node                        instance              state
--------------------------------------------------------------------------
1    192.168.10.182    1    /opt/gaussdb/data/cm/cm_server    Primary
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
[   Cluster State   ]

cluster_state  : Normal
redistributing : No
balanced       : Yes
current_az : AZ_ALL

[    Datanode State    ]

node                        instance               state
--------------------------------------------------------------------------
1    192.168.10.182 6001    /opt/gaussdb/data/dn/dn_6001 P Primary Normal
```

Log Naming Rules

WALs are saved as segment files. Each segment is 16 MB and is divided into multiple 8 KB pages. The name of a segment file consists of 24 hexadecimal characters. The name of a WAL file consists of the timeline, log file identifier, and log file segment identifier. For details, see **Table 5-21**.

The upper limit of the segment identifier of a WAL file is **FF**. If there is another WAL file, add 1 to the file identifier and reset the segment identifier to **0**. For example, the next file of 000000010000000000000FF is 000000010000000100000000. The timeline starts from 1, and the log file identifier and the segment identifier of the file both start from 0.

Table 5-21 WAL file naming rules

Column	Description
Timeline	Time line The value is an 8-digit hexadecimal number.
wal_file_num	Identifier of a WAL file The value is an 8-digit hexadecimal number.
wal_segment_num	Identifier of a WAL file segment The value is an 8-digit hexadecimal number.

The size of each WAL file is 16 MB. Typical WAL file names are as follows:

```
-rW----- 1 gaussdb d gaussdb d 16777216 Jun 11   12:33 000000010000000001000000F2
-rW----- 1 gaussdb d gaussdb d 16777216 Jun 11   18:10 000000010000000001000000F3
-rW----- 1 gaussdb d gaussdb d 16777216 Jun 11   22:18 000000010000000001000000F4
-rW----- 1 gaussdb d gaussdb d 16777216 Jun 12  03:58 000000010000000001000000F5
-rW----- 1 gaussdb d gaussdb d 16777216 Jun 12  07:59 000000010000000001000000F6
-rW----- 1 gaussdb d gaussdb d 16777216 Jun 12  13:45 000000010000000001000000F7
-rW----- 1 gaussdb d gaussdb d 16777216 Jun 12  17:30 000000010000000001000000F8
-rW----- 1 gaussdb d gaussdb d 16777216 Jun 12  23:10 000000010000000001000000F9
-rW----- 1 gaussdb d gaussdb d 16777216 Jun 13  03:10 000000010000000001000000FA
-rW----- 1 gaussdb d gaussdb d 16777216 Jun 13  08:39 000000010000000001000000FB
-rW----- 1 gaussdb d gaussdb d 16777216 Jun 13  12:43 000000010000000001000000FC
-rW----- 1 gaussdb d gaussdb d 16777216 Jun 13  17:08 000000010000000001000000FD
-rW----- 1 gaussdb d gaussdb d 16777216 Jun 13  16:32 000000010000000001000000FE
```

NOTE

The numbers in the WAL file name are used in ascending order in succession. Exhausting all available numbers takes a long time, and the numbers will start from zero again after they reach the maximum.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Log Content Description

The content of WALs depends on the types of recorded transactions. WALs can be used to restore a system after the system breaks down.

By default, GaussDB reads WALs for system restoration during each startup.

Log Application Scenario

Database recovery: When a database crashes unexpectedly, database changes have been recorded in Xlogs. After the database is restarted, the Xlogs after the checkpoint can be replayed to flush the modified data pages (recorded in the Xlogs but not modified in the data files) to disks, ensuring data integrity. This process is automatically completed after the database is started.

Data synchronization between the primary and standby DNs: The standby DN of GaussDB receives Xlogs from the primary DN and applies Xlogs to ensure data consistency between the primary and standby DNs. Data synchronization between the primary and standby databases can also be implemented using Xlogs.

Database backup and restoration: When GaussRoach is used to back up GaussDB, you need to copy data files and Xlogs generated during the backup to form a complete backup. You can use a full backup or an incremental backup and Xlogs to restore data to a specified point in time, that is, perform point-in-time recovery (PITR).

Logical replication: Logical replication tools, such as DRS, parse Xlogs to obtain data streams for incremental data synchronization.

Xlog Recycling

During database running, Xlogs are continuously generated. GaussDB clears Xlogs using the checkpoint mechanism (also called Xlog recycling). For details about the parameters related to Xlog recycling, see **Table 5-22**. For details about how to set parameters, see **Setting Parameters**.

Table 5-22 Parameters related to Xlog recycling

Parameter	Description
checkpoint_timeout	Sets the maximum time between automatic checkpoints. Specifically, this is the interval for the next automatic checkpoint after the last automatic checkpoint. The default value is 15 minutes. If the number of Xlogs accumulated between two automatic checkpoints is greater than the value of **checkpoint_segments**, an automatic checkpoint is triggered.
checkpoint_segments	Maximum number of WAL files that can be retained within the interval specified by **checkpoint_timeout**. The size of each log file is 16 MB.
enable_incremental_checkpoint	Specifies whether to enable incremental checkpointing. The default value is **on**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
incremental_checkpoint_timeout	Specifies the maximum interval between automatic checkpoints when the incremental checkpoint (enable_incremental_checkpoint) is enabled. The default value is 1 minute. That is, an incremental checkpoint is started every minute, and whether Xlogs need to be cleared is checked.
wal_keep_segments	Specifies the minimum number of Xlog files that can be retained in the **pg_xlog** directory. The standby node obtains the Xlogs from the primary node to perform streaming replication. The default value is **128**. Generally, when executing a checkpoint, the database checks whether the number of Xlogs in the **pg_xlog** directory is greater than the value of **wal_keep_segments**. If the number is greater than the value of **wal_keep_segments**, the database deletes the earliest Xlogs until the number of Xlogs is equal to the value of **wal_keep_segments**.
enable_xlog_prune	Specifies whether the primary node recycles logs when any standby node is disconnected and the size of Xlogs exceeds the value of max_size_for_xlog_prune. The value **on** indicates that the functionality is enabled. When a standby node is disconnected, Xlogs between the primary and standby DNs are not cleared to ensure that Xlogs are continuous. However, if **enable_xlog_prune** is enabled and the accumulated Xlog size is greater than the value of **max_size_for_xlog_prune** (generally 256 GB), Xlogs are cleared.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
max_size_for_xlog_prune	Maximum number of Xlogs retained on the primary node when the standby node is faulty. This parameter takes effect when **enable_xlog_prune** is enabled. The mechanism is as follows: 1. If all standby nodes specified by the replconninfo series GUC parameters are connected to the primary node, this parameter does not take effect. 2. This parameter takes effect when one or more standby nodes specified by the replconninfo series GUC parameters are disconnected from the primary node. If the number of Xlogs on the primary node is greater than the value of this parameter, the Xlogs will be forcibly recycled. Exception: In synchronous commit mode (that is, the **synchronous_commit** parameter is not set to **local** or **off**), if there are standby nodes connecting to the primary node, the primary node retains the logs after the minimum log receiving location in the majority of standby nodes. In this case, the number of retained logs may be greater than the value of **max_size_for_xlog_prune**. 3. If a standby node is being built, this parameter does not take effect. All logs on the primary node are retained to prevent failures caused by log recycling during the build operation.

Maintenance Suggestion

WALs are important for database restoration. You are advised to routinely back up WALs.

5.6 Memory Logs

Memory logs record memory context statistics generated when the dynamic memory usage of a database process exceeds the specified threshold, the dynamic memory fluctuation exceeds the specified threshold, or the statistical period reaches the specified value. The GUC parameter **memory_trace_level** determines whether to record memory logs. By default, level-1 logs are recorded. The GUC parameter **memory_log_config** is used to set more functions, such as the dynamic memory usage percentage threshold, memory fluctuation percentage threshold, statistical period, number of top memory contexts, and number of top SQL statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- An anti-oscillation mechanism is provided for generating memory logs. When the dynamic memory usage exceeds the threshold (specified by the first part of the **memory_log_config** parameter), the dynamic memory used for generating memory logs is recorded as the trigger memory. If the dynamic memory usage is continuously greater than the trigger memory minus 5% of dynamic memory upper limit, no memory log is generated because the dynamic memory usage exceeds the threshold. This prevents repeated log generation when the dynamic memory usage fluctuates slightly around the threshold.
- The memory context snapshot generation based on the memory fluctuation percentage (set in part 2 of the **memory_log_config** parameter) is based on a 3-second detection period. If the snapshot generation conditions are met within the 3-second detection period, no memory context snapshot is generated.
- Memory threshold triggering, memory fluctuation triggering, and periodic triggering cannot be performed together.
- Only the memory contexts in the format of *{Module name}_{Number}* can be combined by module name. If the memory contexts do not meet the requirement, they are not combined. For example, **pooler_memory_context**, **pooler_core_context**, and **pooler_agent_context** are three different memory contexts and are not combined.

Log Storage Directory

The memory logs of the database node are stored in the *$GAUSSLOG*/**mem_log** directory.

Log Naming Rules

The name format of database node memory logs is: **mem_log-***creation time*.**log**

Log Content Description

Detailed usage statistics of the global memory context, session-level memory context, and thread-level memory context of the database process, including the top 20 session-level and thread-level memory usage, corresponding client types, and SQL statements. If parallel logical decoding tasks are running, the log also contains the memory usage information of parallel logical decoding.

Top *n* memory contexts of each type, top *n* memory contexts of each query (specified by the fourth part of the **memory_log_config** parameter), and top *n* queries (specified by the fifth part of the **memory_log_config** parameter) are displayed based on the memory usage.

Table 5-23 Log parameters

Column	Description
Max_dynamic_memory	Maximum dynamic memory that can be used by the database process.
Dynamic_used_memory	Size of the dynamic memory used by the current process.
Dynamic_peak_memory	Historical maximum dynamic memory usage of the current process.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Dynamic_thread_cxt_used_memory	Size of the memory used by the thread-level memory context of the current process.
Dynamic_session_cxt_used_memory	Size of the memory used by the session-level memory context of the current process. This parameter is displayed only when the thread pool is enabled.
Dynamic_used_shrctx	Size of the memory occupied by the global memory context of the current process.
Dynamic_peak_shrctx	Historical maximum size of the memory occupied by the global memory context of the current process.
Max_backend_memory	Maximum memory size requested by the session connected to the HA port.
Backend_used_memory	Size of the memory that is being occupied by the session connected to the HA port.
other_used_memory	Size of other memory that is not applied for in the memory context.
unit	Unit of the memory size at the same level.
Context type	Memory context type, including **Shared Memory Context**, **Thread Memory Context**, and **Session Memory Context**.
Context	Name of the memory context. Memory contexts with the same name and module name (in the format of *{Module name}_{Number}*) are displayed together. Only the module name is displayed.
freeSize	Size of the idle memory cached in the memory context.
totalSize	Total size of the memory requested by the memory context.
File	File name of the code where the memory allocation is located. It is recorded only when **memory_trace_level** is set to **level2**.
Line	File line number of the code where the memory allocation is located. It is recorded only when **memory_trace_level** is set to **level2**.
Size	Accumulated size of the memory allocated for in a line of a file. It is recorded only when **memory_trace_level** is set to **level2**.
count	Number of memory contexts with the same name and module name.
query	Executed statement.
unique query id	Unique ID of a statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
debug query id	Debugging ID of a statement.
session id	Session ID.
thread id	Thread ID.
mem size	Size of the memory allocated by a statement.
application name	Application name.

Log Examples

- When **memory_trace_level** is set to **level1**, the following logs are recorded:

```
{
  "Global Memory Statistics": {
  "Max_dynamic_memory": 34012,
  "Dynamic_used_memory": 3645,
  "Dynamic_peak_memory": 3664,
  "Dynamic_thread_cxt_used_memory": 383,
  "Dynamic_session_cxt_used_memory": 2582,
  "Dynamic_used_shrctx": 401,
  "Dynamic_peak_shrctx": 401,
  "Max_backend_memory": 648,
  "Backend_used_memory": 1,
  "other_used_memory": 0,
  "unit": "MB"
 },
 "Memory Context Info": {
  "Memory Context Detail": {
   "Context Type": "Shared Memory Context",
   "Memory Context": {
    "context": "IncreCheckPointContext",
    "freeSize": 2864,
    "totalSize": 263066352,
    "unit": "Byte",
    "count": 1
   },
   ...
  },
  "Memory Context Detail": {
   "Context Type": "Session Memory Context",
   "Memory Context": {
    "context": "CachedPlan",
    "freeSize": 68041368,
    "totalSize": 235937792,
    "unit": "Byte",
    "count": 1
   },
   ...
  },
  "Memory Context Detail": {
   "Context Type": "Thread Memory Context",
   "Memory Context": {
    "context": "LocalSysCacheShareMemoryContext",
    "freeSize": 60431360,
    "totalSize": 644141760,
    "unit": "Byte",
    "count": 1
   },
   ...
  }
 "Query Info": {
  "Query Context": {
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    "query":  "select distinct a from t3 order by a;",
    "unique query id":  "2286703632",
    "debug query id":  "1407374883554034",
    "session id":  "2",
    "thread id":  "140188551411456",
    "mem size":  1,
    "unit":  "MB",
    "application name":  "gsql",
    "Memory Context Detail":  {
       "Context Type":  "Session Memory Context",
       "Memory Context":  {
       "context":  "SessionTopMemoryContext",
       "freeSize":  97048,
       "totalSize":  460520,
       "unit":  "Byte",
       "count":  1
    },
    ...
   },
  "Query Context":  {
      "query":  "WLM fetch collect info from data nodes",
      "unique query id":  "0",
      "debug query id":  "0",
      "session id":  "140186948200192",
      "thread id":  "140186948200192",
      "mem size":  1,
      "unit":  "MB",
      "application name":  "workload",
      "Memory Context Detail":  {
         "Context Type":  "Thread Memory Context",
         "Memory Context":  {
         "context":  "LocalSysCacheShareMemoryContext",
         "freeSize":  150720,
         "totalSize":  1872096,
         "unit":  "Byte",
         "count":  1
      },
      ...
    },
   ...
 },
 "parallel logical decode Info":  [{
  "slot name":  "slot1",
  "memory info":  "memory usage: reader toast memory size: 0, sender out buf memory size: 0, [xid,
first_lsn, memory, entry num] : NULL"
 }]
}
```


- When **memory_trace_level** is set to **level2**, the following logs are recorded:


```
{
  "Global Memory Statistics":  {
  "Max_dynamic_memory":  34012,
  "Dynamic_used_memory":  3645,
  "Dynamic_peak_memory":  3664,
  "Dynamic_thread_cxt_used_memory":  383,
  "Dynamic_session_cxt_used_memory":  2582,
  "Dynamic_used_shrctx":  401,
  "Dynamic_peak_shrctx":  401,
  "Max_backend_memory":  648,
  "Backend_used_memory":  1,
  "other_used_memory":  0,
  "unit":  "MB"
 },
 "Memory Context Info":  {
  "Memory Context Detail":  {
    "Context Type":  "Shared Memory Context",
    "Memory Context":  {
      "context":  "IncreCheckPointContext",
      "freeSize":  2864,
      "totalSize":  263066352,
```


 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
    "unit":  "Byte",
    "count":  1,
    "MemoryDetail":  {
        "file":"xxx.cpp",
        "line":411,
        "size":211131,
     }
     ...
  },
 ...
},
"Memory Context Detail":       {
  "Context Type": "Session Memory Context",
  "Memory Context":       {
    "context":      "CachedPlan",
    "freeSize":     68041368,
    "totalSize":    235937792,
    "unit":    "Byte",
    "count":    1,
    "MemoryDetail":  {
        "file":"xxx.cpp",
        "line":411,
        "size":211131,
     }
     ...
  },
  ...
},
"Memory Context Detail":  {
  "Context Type":  "Thread Memory Context",
  "Memory Context":  {
    "context":  "LocalSysCacheShareMemoryContext",
    "freeSize":  60431360,
    "totalSize":  644141760,
    "unit":  "Byte",
    "count":  1,
    "MemoryDetail":  {
        "file":"xxx.cpp",
        "line":411,
        "size":211131,
     }
     ...
  },
  ...
}
"Query Info":  {
 "Query Context":  {
    "query":    "select distinct a from t3 order by a;",
    "unique query id":  "2286703632",
    "debug query id":  "1407374883554034",
    "session id":  "2",
    "thread id":  "140188551411456",
    "mem size":  1,
    "unit":  "MB",
    "application name":  "gsql",
    "Memory Context Detail":  {
       "Context Type":  "Session Memory Context",
       "Memory Context":  {
       "context":  "SessionTopMemoryContext",
       "freeSize":  97048,
       "totalSize":  460520,
       "unit":  "Byte",
       "count":  1,
       "MemoryDetail":  {
          "file":  "xxx.cpp",
          "line":  461,
          "size":  62976
       },
       ...
```

```
      },
      ...
    },
  "Query Context": {
      "query": "WLM fetch collect info from data nodes",
      "unique query id": "0",
      "debug query id": "0",
      "session id": "140186948200192",
      "thread id": "140186948200192",
      "mem size": 1,
      "unit": "MB",
      "application name": "workload",
      "Memory Context Detail": {
          "Context Type": "Thread Memory Context",
          "Memory Context": {
          "context": "LocalSysCacheShareMemoryContext",
          "freeSize": 150720,
          "totalSize": 1872096,
          "unit": "Byte",
          "count": 1,
          "MemoryDetail": {
              "file":    "xxx.cpp",
              "line":    352,
              "size":    40153344
          },
          ...
      },
      ...
    },
   ...
 },
 "parallel logical decode Info": [{
  "slot name": "slot1",
  "memory info": "memory usage: reader toast memory size: 0, sender out buf memory size: 0, [xid,
first_lsn, memory, entry num] : NULL"
 }]
}
```

Graphical Memory Information (HTML File)

When a .log file in JSON format is generated, a .gz package of the HTML file with the same name is generated in the same directory. Decompress the package, export the HTML file and the **echarts.min.js** file in the same directory, and use a browser to open the HTML file. You can view the graphical memory information.

The memory information corresponds to the memory log in JSON format and consists of the following parts:

- **Global Memory Statistics**: displays the ring chart of the proportion of dynamic memory of each category to the maximum dynamic memory.
- **Shared Memory Context**: displays the memory bar chart of the shared memory context. If the fourth part of the GUC parameter **memory_log_config** is set to **0**, it is not displayed.
- **Session Memory Context**: displays the memory bar chart of the session memory context. If the fourth part of the GUC parameter **memory_log_config** is set to **0**, or the GUC parameter **enable_thread_pool** is set to **off**, it is not displayed.
- **Thread Memory Context**: displays the memory bar chart of the thread memory context. If the fourth part of the GUC parameter **memory_log_config** is set to **0**, it is not displayed.
- **SQL Details**: displays the memory information of the executed statement or thread. If the fifth part of the GUC parameter **memory_log_config** is set to **0**,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

it is not displayed. If there are memory contexts, you can click each statement to display the memory bar chart of the top *n* memory contexts.

NOTE

Compared with **level1**, when the GUC parameter **memory_trace_level** is set to **level2**, the memory location and size information about the memory context is displayed. You can click the bar chart of each memory context to display the memory location and size.

The following table describes the columns of the information displayed in each part.

Table 5-24 Display information

Column	Description
Dynamic used share ctx memory	Size of the dynamic memory used by the current process.
Dynamic thread cxt used memory	Size of the memory used by the thread-level memory context of the current process.
Dynamic session cxt used memory	Size of the memory used by the session-level memory context of the current process. This parameter is displayed only when the thread pool is enabled.
Dynamic unused memory	Size of the memory not used by the current process.
Context type	Memory context type, including **Shared Memory Context**, **Thread Memory Context**, and **Session Memory Context**.
Context	Name of the memory context. Memory contexts with the same name and module name (in the format of *{Module name}_{Number}*) are displayed together. Only the module name is displayed.
totalSize	Total size of the memory requested by the memory context.
File	File name of the code where the memory allocation is located. It is recorded only when **memory_trace_level** is set to **level2**.
Line	File line number of the code where the memory allocation is located. It is recorded only when **memory_trace_level** is set to **level2**.
Size	Accumulated size of the memory allocated for in a line of a file. It is recorded only when **memory_trace_level** is set to **level2**.
query	Executed statement.
unique query id	Unique ID of a statement.
debug query id	Debugging ID of a statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
session id	Session ID.
thread id	Thread ID.
mem size	Size of the memory allocated by a statement.
application name	Application name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6 Database GUC Parameters

6.1 Viewing Parameters

Viewing GUC Parameters

GaussDB uses a set of default GUC parameters after it is installed. You can modify the GUC parameters to enable GaussDB to better fit your service scenarios and data volume.

Procedure

Step 1 Connect to the database. For details, see "Database Quick Start > Connecting to a Database > Using gsql to Connect to a Database" in *Developer Guide*.

Step 2 Check the value of the GUC parameter.

- Method 1: Run the **SHOW** command.
  - View the value of a certain parameter.
    ```
    gaussdb=# SHOW client_encoding;
    ```
    The **client_encoding** parameter is used to check the character encoding type of the client.
  - View values of all parameters.
    ```
    gaussdb=# SHOW ALL;
    ```
- Method 2: Query the pg_settings view.
  - View the value of a certain parameter.
    ```
    gaussdb=# SELECT * FROM pg_settings WHERE NAME='client_encoding';
    ```
  - View values of all parameters.
    ```
    gaussdb=# SELECT * FROM pg_settings;
    ```

----End

Examples

Check the character encoding type of the client.

```
gaussdb=# SHOW client_encoding;
 client_encoding
-----------------
 UTF8
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Viewing CM Parameters

You can view CM parameters in the configuration files or by running commands.

- Viewing parameters in the configuration files

  You can view CM Agent parameters in the **cm_agent.conf** file in the CM Agent data directory and CM Server parameters in the **cm_server.conf** file in the CM Server data directory.

- Viewing parameters by running commands

  ```
  cm_ctl list --param {--agent | --server} [-n NODEID]
  ```

  Parameters

  – **--agent**: lists CM Agent parameters.

  – **--server**: lists CM Server parameters.

  – **-n** *NODEID*: lists the CM parameters of a specified node. *NODEID* indicates the node name, which can be queried by running the **cm_ctl query -Cv** command.

  Examples

  – List CM Agent parameters.

  ```
  cm_ctl list --param --agent
  ```

  – List CM Server parameters.

  ```
  cm_ctl list --param --server
  ```

6.2 Setting Parameters

Setting GUC Parameters

GaussDB provides six types of GUC parameters and multiple methods for modifying GUC parameters. You can set GUC parameters for databases, users, and sessions. For details, see **Table 6-1**.

- Parameter names are case-insensitive.
- The parameter values can be integers, floating points, strings, Boolean values, or enumerated values.
  – The Boolean values can be (**on**, **off**), (**true**, **false**), (**yes**, **no**), or (**1**, **0**). The Boolean values (**on**, **off**) are recommended for GaussDB.
  – The enumerated value range is specified in the **enumvals** column of the system catalog pg_settings.
  – For parameters of the floating-point (real) type, the decimal point is required. For example, 1.0e-3 is valid, but 1e-3 is invalid.
  – According to the IEEE 754 standard, +0 and –0 are defined for floating-point numbers. In this case, the value **–0** can be verified for numeric parameters.
  – gs_guc has a precision threshold of 1.0e-9 for checking the range of floating-point numbers.
- For parameters using units, specify their units during the setting. Otherwise, default units are used.
  – The default units are specified in the **unit** column of pg_settings.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - The unit of memory can be KB, MB, or GB.
  - The unit of time can be ms, s, min, h, or d.
- You can set parameters related to CNs and DNs at a time, but cannot do the same to other parameters.

For details about parameters, see **GUC Parameters**.

Table 6-1 GUC parameter types

Parameter Type	Description	Setting Method
INTERNAL	Fixed parameters. They are set during database creation and cannot be modified. Users can only view the parameters by running the **SHOW** command or in the pg_settings view.	None
POSTMASTER	Database server parameters. They can be set when the database is started or in the configuration file.	Method 1 in **Table 6-2**.
SIGHUP	Global database parameters. They can be set during database startup or be modified later.	Method 1 or 2 in **Table 6-2**.
BACKEND	Session connection parameters. They are specified during session connection creation and cannot be modified after that. The parameter setting becomes invalid when the session is disconnected. The parameters of this type are internal parameters and not recommended for users to set it.	Method 1 or 2 in **Table 6-2**. **NOTE** The parameter setting takes effect when the next session is created.
SUSET	Database administrator parameters. They can be set by common users during database startup or after the database is started. They can also be set by database administrators using SQL statements.	Method 1 or 2 by a common user, or method 3 by a database administrator in **Table 6-2**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter Type	Description	Setting Method
USERSET	Common user parameters. They can be set by any user at any time.	Method 1, 2, or 3 in Table 6-2. **NOTE** When you set parameters of the USERSET type, the parameter value set using **ALTER DATABASE** takes precedence over that set using gs_guc. To make the parameter settings of gs_guc take effect, run the **alter database** *xxx* **reset** *xxx* command to reset the parameters.

You can set GUC parameters in GaussDB using the methods listed in Table 6-2.

Table 6-2 Methods for setting GUC parameters

No.	Setting Method
Method 1	1. Log in to the management console. 2. On the **Instances** page, click the name of the target instance to go to the **Basic Information** page. 3. In the navigation pane on the left, choose **Parameters**. On the displayed page, modify parameters. If the parameters cannot be modified on the console, evaluate the modification risks and contact customer service for modification. 4. Reboot the DB instance to make the modifications take effect. **NOTE** Rebooting instances will interrupt user operations. Plan a proper execution window before the reboot.
Method 2	1. Log in to the management console. 2. On the **Instances** page, click the name of the target instance to go to the **Basic Information** page. 3. In the navigation pane on the left, choose **Parameters**. On the displayed page, modify parameters. If the parameters cannot be modified on the console, evaluate the modification risks and contact customer service for modification.
Method 3	Modify a session-level parameter. • Set a session-level parameter. gaussdb=# SET *paraname* TO *value*; The parameter value in the current session is changed. After you exit the session, the setting becomes invalid.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If you use method 1 or 2 to set a parameter that does not belong to the current node, the database displays a message indicating that the parameter is not supported.
- You cannot set a parameter in the unit of pages by using method 1 or method 2.
- When you use method 3 to set a parameter, if the parameter value is an integer, leading zeros will be filtered out. For example, **SET** *paraname* **TO** *008192* and **SET** *paraname* **TO** *8192* have the same effect.

Setting CM Parameters

GaussDB CM parameters fall into two categories: those that take effect only after a cluster restart, and those take effect immediately after a reload. For details, see **Table 6-3**.

Table 6-3 Methods for setting CM parameters

No.	Setting Method

Method 1	1. Run the following command to modify parameters:
	`cm_ctl set --param {--agent | --server} [-n NODEID] -k "PARAMETER=value"`
	NOTE
	Parameters
	• **--agent**: sets CM Agent parameters.
	• **--server**: sets CM Server parameters.
	• **-n** *NODEID*: sets the CM parameters of a specified node. *NODEID* indicates the node name, which can be queried by running the **cm_ctl query -Cv** command. If this parameter is not set, all nodes are set by default.
	• **-k "***PARAMETER***=***value***"**: specifies the parameters to be set and values of the parameters. Only existing parameters can be set. Parameters cannot be added or deleted.
	Examples
	• Set the CM Agent parameter **incremental_build** to **on**.
	`cm_ctl set --param --agent -k "incremental_build=on"`
	• Set the CM Server parameter **switch_rto** to **600**.
	`cm_ctl set --param --server -k "switch_rto=600"`
	2. Restart the cluster for the parameters to take effect.
	`cm_ctl stop && cm_ctl start`
	NOTE
	Restarting the cluster results in operation interruption. Properly plan the restart to avoid affecting users.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Setting Method

Method 2	1. Run the following command to modify parameters:
	`cm_ctl set --param {--agent | --server} [-n NODEID] -k "PARAMETER=value"`
	NOTE
	Parameters
	• **--agent**: sets CM Agent parameters.
	• **--server**: sets CM Server parameters.
	• **-n** *NODEID*: sets the CM parameters of a specified node. *NODEID* indicates the node name, which can be queried by running the **cm_ctl query -Cv** command. If this parameter is not set, all nodes are set by default.
	• **-k "***PARAMETER***=***value***"**: specifies the parameters to be set and values of the parameters. Only existing parameters can be set. Parameters cannot be added or deleted.
	Examples
	• Set the CM Agent parameter **incremental_build** to **on**.
	`cm_ctl set --param --agent -k "incremental_build=on"`
	• Set the CM Server parameter **switch_rto** to **600**.
	`cm_ctl set --param --server -k "switch_rto=600"`
	2. Dynamically load CM parameters.
	`cm_ctl reload --param {--agent | --server} [-n NODEID]`
	NOTE
	Parameters
	• **--agent**: dynamically loads CM Agent parameters.
	• **--server**: dynamically loads CM Server parameters.
	• **-n** *NODEID*: dynamically loads the CM parameters of a specified node. *NODEID* indicates the node name, which can be queried by running the **cm_ctl query -Cv** command.
	Examples
	• Dynamically load CM Agent parameters.
	`cm_ctl reload --param --agent`
	• Dynamically load CM Server parameters.
	`cm_ctl reload --param --server`

6.3 GUC Parameters

6.3.1 GUC Parameter Usage

A database provides many GUC parameters. Changing these parameters will affect the behavior of the database system. Before modifying these parameters, learn the specifications of these parameters and the impact of the modification on the database. Otherwise, unexpected results may occur.

- Some internal database parameters are related to the environment and have been specified during database installation. You are advised not to modify them. If the modification is required, ensure that the content and format are correct. Otherwise, the instance will be abnormal. For example, set the **listen_addresses**, **port**, and **local_bind_address** (IP address) parameters, and various parameters for setting the file output path.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Due to the size of hardware resources, the following parameters need to be modified with caution: memory, number of connections, and parameters that affect disk space usage. Before modifying the parameters, check the hardware specifications to ensure that the modified parameters do not exceed the hardware size limit. Otherwise, the database instance may fail to apply for sufficient resources and the instance status may be abnormal.
- Due to the association between parameters, you need to read the specifications of the parameters before modifying the parameters. For example, the value of **max_wal_senders** must be smaller than that of **max_connections** so that the database can be started properly.
- If this parameter is set to a large value or a small value, services may be affected. Therefore, you need to learn the meaning of this parameter and modify it with caution. For example, if the **session_timeout** parameter is set to **1** (in seconds) and no operation is performed within 1 second after the service is connected to the server, the service is disconnected. As a result, the service execution is affected. **log_min_messages** specifies the log level. If the log level is too low, a large number of logs will be recorded, affecting the performance.

Precautions

- If the value range of a parameter is a string, the string should comply with the path and file naming conventions of the OS running the target database.
- If the maximum value of a parameter is *INT_MAX*, the maximum parameter value varies by OS. *INT_MAX* indicates the maximum value of the INT data type. The value is **2147483647**.
- If the maximum value of a parameter is *DBL_MAX*, the maximum parameter value varies by OS. The value is the maximum value of the FLOAT data type.
- Some GUC parameters affect the selection of functions and operators, compilation of stored procedures, and generation of execution plans. As a result, views, default values of function parameters, compilation products of stored procedures, and plan cache are affected. Due to these mechanisms, subsequent GUC parameter changes may not affect these behaviors.
  - View: When defining a view, the database generates rewriting rules based on the GUC parameter status. Subsequent operations on the view directly follow the corresponding rules without being affected by GUC parameters (which affect the generation of rewriting rules).
  - Default value of a function parameter: For a function parameter, a function is selected as the default parameter value based on the GUC parameter status when the function parameter is created and the OID of the function that is used as the default value is recorded in the corresponding system catalog (pg_proc). When this function parameter is executed later, the function OID recorded in the system catalog will be used as the default value, and the value is not affected by GUC parameters.
  - Compilation product of a stored procedure: When compiling a stored procedure, the database generates a compilation product based on the GUC parameter status. When executing the stored procedure, the database directly uses the compilation product without being affected by GUC parameters (which affect the generation of compilation products).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

– Plan cache: When executing an SQL statement, the database generates an execution plan based on the GUC parameter status. If the plan is cached, the SQL statement will be executed according to the plan without being affected by GUC parameters (which affect the generation of execution plans).

The following table lists the GUC parameters that exert influence. You can view the function of each GUC parameter.

GUC Parameter	Enabled	Disabled	Impact	Example

convert_string_to_digit	Character strings are preferentially converted to numbers.	Character strings cannot be preferentially converted to numbers.	Views, default values of function parameters, stored procedure compilation products, and plan cache.	CREATE or REPLACE FUNCTION test(c numeric) RETURN text package AS
				BEGIN
				RETURN 'test(c numeric)';
				END;
				/
				CREATE or REPLACE FUNCTION test(c varchar2) RETURN text package AS
				BEGIN
				RETURN 'test(c varchar2)';
				END;
				/
				
				SET convert_string_to_digit=on;
				CREATE or REPLACE VIEW test_view AS SELECT test('123'::text);
				
				SELECT test('123'::text);
				test
				----------------- 	
				test(c numeric)
				(1 row)
				
				SELECT * FROM test_view;
				test
				----------------- 	
				test(c numeric)
				(1 row)
				
				-- Disable the parameter.
				SET convert_string_to_digit=off;
				
				SELECT test('123'::text);
				test
				----------------- 	
				test(c varchar2)
				(1 row)
				
				-- The view behavior is inconsistent with the direct function calling behavior.
				SELECT * FROM test_view;
				test
				----------------- 	
				test(c numeric)
				(1 row))

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example
set behavior_compat_options = 'current_sysdate';	The SYSDATE function obtains the current OS time. (The current_sysdate function is used at the bottom layer.)	The SYSDATE function obtains the current database time. (The sysdate function is used at the bottom layer.)	Views, default values of function parameters, stored procedure compilation products, and plan cache.	SET behavior_compat_options = 'current_sysdate'; CREATE or REPLACE VIEW test_view_SYSDATE AS SELECT SYSDATE AS test; SELECT * FROM test_view_SYSDATE;-- Returns the current OS time. SELECT SYSDATE;-- Returns the current OS time. -- Disable the parameter. SET behavior_compat_options = ''; -- The view behavior is inconsistent with the direct function calling behavior. SELECT * FROM test_view_SYSDATE;-- Returns the current OS time. SELECT SYSDATE;-- Returns the current database time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example
set a_format_version='10c'; set a_format_dev_version='s1';	• NVL2 is supported. • Some system functions are added. • The CASE WHEN return type is changed. • Constructors of the set type are prior to functions. • Implicit conversion from time	The functions supported when the parameter is enabled are not supported.	Views, default values of function parameters, stored procedure compilation products, and plan cache.	SET a_format_version='10c'; SET a_format_dev_version='s1'; CREATE OR REPLACE VIEW test_view_nvl2 AS SELECT nvl2(1,2,3) AS test; SELECT * FROM test_view_nvl2; test ------ 2 (1 row) RESET a_format_dev_version; SELECT * FROM test_view_nvl2; test ------ 2 (1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example
	stamp to timestampz is supported.			

GUC Parameter	Enabled	Disabled	Impact	Example
set a_format_version='10c'; set a_format_dev_version='s2';	• CURRENT_TIMESTAMP '('FCONST')' is supported. • DBTIMEZONE is supported. • LNNVL is supported. • CURRENT_DATE If both **a_format_date_timestamp** and **mapping_date_to_datea** are set to	The functions supported when the parameter is enabled are not supported.	Views, default values of function parameters, stored procedure compilation products, and plan cache.	SET a_format_version='10c'; SET a_format_dev_version='s2'; CREATE or REPLACE VIEW test_view_LNNVL AS SELECT LNNVL(123=123) AS test; SELECT * FROM test_view_LNNVL; RESET a_format_dev_version; SELECT * FROM test_view_LNNVL;

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example
	off, the return value column name is **timestamp**. If **a_format_date_timestamp** is set to **on** and **mapping_date_to_datea** is set to **off**, the return value column name is **current_date**. If **map**			

GUC Parameter	Enabled	Disabled	Impact	Example
	ping_date_to_datea is set to **on**, **a_format_date_timestamp** does not affect the column name and the return value column name is **current_date**. • Some system functions are added.			

GUC Parameter	Enabled	Disabled	Impact	Example
set a_format_version='10c'; set a_format_dev_version='s4';	Some system functions are added.	The functions supported when the parameter is enabled are not supported.	Views, default values of function parameters, stored procedure compilation products, and plan cache.	-

GUC Parameter	Enabled	Disabled	Impact	Example

set a_format_version='10c';
set a_format_dev_version='s5';
	• Some system functions are added.
	• Composite constructors are prior to functions.
		The functions supported when the parameter is enabled are not supported.	Views, default values of function parameters, stored procedure compilation products, and plan cache.	-- The composite type constructor is prior to functions.
				CREATE TYPE tt AS (
				val1 int,
				val2 int
				);
				CREATE OR REPLACE FUNCTION tt(va int, vb int)
				RETURN int IS
				ret int;
				BEGIN
				ret := va;
				RETURN ret;
				END;
				/
				
				SET a_format_version='10c';
				SET a_format_dev_version='s5';
				
				-- Assign a value to the *tt* variable.
				CREATE OR REPLACE FUNCTION test0
				RETURN int IS
				va tt;
				ret int;
				BEGIN
				va := tt(1,2);
				RAISE INFO 'tt: %' ,va;
				ret := 9;
				RETURN ret;
				END;
				/
				
				SELECT test0();
				INFO: tt: (1,2)
				test0
				------- 	
				9
				(1 row)
				
				SET a_format_version='';
				SET a_format_dev_version='';
				
				SELECT test0();
				INFO: tt: (1,2)
				test0
				------- 	
				9
				(1 row)
				
				-- Assign a value to the *tt* variable.
				CREATE OR REPLACE FUNCTION test0
				RETURN int IS
				va tt;
				ret int;
				BEGIN
				va := tt(1,2);
				RAISE INFO 'tt: %' ,va;
				ret := 9;
				RETURN ret;
				END;
				/
				
				select test0();
				ERROR: cannot assign non-composite value to a row variable

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example
set behavior_compat_options = 'enable_bpcharlikebpchar_compare';	The bpcharlikebpchar and bpcharnlikebpchar operators are enabled.	The bpcharlikebpchar and bpcharnlikebpchar operators are disabled.	Views, stored procedure compilation products, and plan cache	CREATE TABLE op_test (col BPCHAR(2) DEFAULT NULL); CREATE INDEX op_index ON op_test(col); INSERT INTO op_test VALUES ('a'); INSERT INTO op_test VALUES ('1'); INSERT INTO op_test VALUES ('11'); INSERT INTO op_test VALUES ('12'); INSERT INTO op_test VALUES ('sd'); INSERT INTO op_test VALUES ('aa'); SET behavior_compat_options = 'enable_bpcharlikebpchar_compare'; EXPLAIN (COSTS OFF) SELECT * FROM op_test WHERE col LIKE col::BPCHAR ORDER BY col; QUERY PLAN ------------------------------ Sort Sort Key: col -> Seq Scan on op_test Filter: (col ~~ col) (4 rows) CREATE OR REPLACE VIEW test_view_bpchar AS SELECT * FROM op_test WHERE col LIKE col::BPCHAR ORDER BY col; EXPLAIN SELECT * FROM test_view_bpchar; QUERY PLAN -- Sort (cost=34.48..34.50 rows=10 width=12) Sort Key: op_test.col -> Seq Scan on op_test (cost=0.00..34.31 rows=10 width=12) Filter: (col OPERATOR(pg_catalog.~~) col) (4 rows) SET behavior_compat_options = ''; EXPLAIN (COSTS OFF) SELECT * FROM op_test WHERE col LIKE col::BPCHAR ORDER BY col; QUERY PLAN --------------------------------------- Sort Sort Key: col -> Seq Scan on op_test Filter: (col ~~ (col)::text) (4 rows) EXPLAIN SELECT * FROM test_view_bpchar; QUERY PLAN -- Sort (cost=34.48..34.50 rows=10

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example
				width=12) Sort Key: op_test.col -> Seq Scan on op_test (cost=0.00..34.31 rows=10 width=12) Filter: (col OPERATOR(pg_catalog.~~) col) (4 rows) -- Rebuild a view when the parameter is disabled. CREATE OR REPLACE VIEW test_view_bpchar AS SELECT * FROM op_test WHERE col LIKE col::BPCHAR ORDER BY col; explain select * from test_view_bpchar; QUERY PLAN ---------- ---------- Sort (cost=39.34..39.37 rows=10 width=12) Sort Key: op_test.col -> Seq Scan on op_test (cost=0.00..39.17 rows=10 width=12) Filter: (col ~~ (col)::text) (4 rows)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example
set b_format_version='5.7';set b_format_dev_version='s1';	• CURDATE, CURRENT_TIME, CURRENT_TIMESTAMP, LOCALTIME, and LOCALTIMESTAMP NOW return the statement execution timestamp. • SYSDATE returns the function execution time	• The functions CURDATE, CURRENT_TIME, CURRENT_TIMESTAMP, LOCALTIME, LOCALTIMESTAMP NOW, and SYSDATE return the timestamp of the transaction start time. • The function	Views, default values of function parameters, stored procedure compilation products, and plan cache.	-

GUC Parameter	Enabled	Disabled	Impact	Example
	stamp. • The behaviors of the DB_JSOBJ, LAST_DAY_FUNC, EXTRACT, TIMESTAMPDIFF, and SUBSTRING functions are changed to be the same as those of MySQL.	s DB_JSOBJ, LAST_DAY_FUNC, EXTRACT, TIMESTAMPDIFF, and SUBSTRING are restored to the default behavior.		

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example
set behavior_compat_options = 'bind_procedure_searchpath';	• The search path of the database objects in a stored procedure without specifying the schema name is specified. • If the schema name is implicitly specified for a stored procedure, the schema	The default behavior is restored.	Stored procedure compilation products.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example
	to which the stored procedure belongs is searched preferentially.			

GUC Parameter	Enabled	Disabled	Impact	Example

set behavior_compat_options='skip_insert_gs_source';	The default behavior is restored.	When creating PL/SQL objects (stored procedures, functions, and packages), do not insert PL/SQL object creation information into gs_source.	Products of PL/SQL objects (stored procedures, functions, and packages).	CREATE OR REPLACE PROCEDURE proc_no_parameter
				AS
				ret1 int;
				BEGIN
				ret1 := 0;
				dbe_output.print_line('procedure has no parameter is 1w111.');
				END;
				/
				SELECT * FROM DBE_PLDEVELOPER.gs_source;
				id | owner | nspid | name | type | status | src
				-------+-------+-------+-------------------+-----------+-------
				+---
				24593 | 10 | 2200 | proc_no_parameter | procedure | t
				| CREATE OR REPLACE PROCEDURE proc_no_parameter +
				| | | | | |
				AS
				+
				| | | | | |
				| ret1 int;
				+
				| | | | | |
				BEGIN
				+
				| | | | | |
				| ret1 := 0;
				+
				| | | | | |
				| dbe_output.print_line('procedure has no parameter is 1w111.');+
				| | | | | |
				| END;
				(1 row)

set behavior_compat_options = 'enable_out_param_override';	Determines the overloading of output parameters of a stored procedure.	The default behavior is restored.	Stored procedure compilation products.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example

enable_bitmapscan	The bitmapscan operator is selected.	The bitmapscan operator is not selected.	Plan cache.	SET enable_auto_explain=true;
				SET auto_explain_level=notice;
				CREATE TABLE t1(a int, b int, c int,d varchar)WITH(storage_type=ustore);
				CREATE TABLE t2(a int, b int, c int,d varchar)WITH(storage_type=ustore);
				INSERT INTO t1 SELECT i,10*random()*i,i%20,'atr'||i from generate_series(1,1000) i;
				CREATE INDEX index1 ON t1(a);
				
				SET enable_bitmapscan=on;
				SET enable_seqscan=off;
				SET enable_indexscan=off;
				CREATE OR REPLACE PROCEDURE proc_while_loop()
				AS
				DECLARE
				i int :=1;
				BEGIN
				WHILE i < 5 LOOP
				INSERT INTO t2 SELECT * FROM t1 WHERE a = i ;
				i:=i+1;
				END LOOP;
				END;
				/
				CALL proc_while_loop();
				SET enable_bitmapscan=off;
				CALL proc_while_loop();
				SET enable_bitmapscan=on;
				CALL proc_while_loop();

enable_indexscan	The indexscan operator is selected.	The indexscan operator is not selected.	Plan cache.	SET enable_auto_explain=true;
				SET auto_explain_level=notice;
				CREATE TABLE t1(a int, b int, c int,d varchar)WITH(storage_type=ustore);
				CREATE TABLE t2(a int, b int, c int,d varchar)WITH(storage_type=ustore);
				INSERT INTO t1 SELECT i,10*random()*i,i%20,'atr'||i from generate_series(1,1000) i;
				CREATE INDEX index1 on t1(a);
				SET enable_bitmapscan=off;
				SET enable_seqscan=off;
				CREATE OR REPLACE PROCEDURE proc_while_loop()
				AS
				DECLARE
				i int :=1;
				BEGIN
				WHILE i < 5 LOOP
				insert into t2 select * from t1 where a = i ;
				i:=i+1;
				END LOOP;
				END;
				/
				CALL proc_while_loop();
				SET enable_indexscan=off;
				CALL proc_while_loop();

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example

query_Dop	Users can specify the degree of query parallelism. If the SMP function is enabled, the system uses the specified degree of parallelism.	The default value is **1**, which indicates that parallel query is disabled.	Plan cache.	SET enable_auto_explain=true;
				SET auto_explain_level=notice;
				CREATE TABLE t1(a int, b int, c int,d varchar)WITH(storage_type=ustore);
				CREATE TABLE t2(a int, b int, c int,d varchar)WITH(storage_type=ustore);
				INSERT INTO t1 SELECT i,10*random()*i,i%20,'atr'||i from generate_series(1,1000) i;
				CREATE INDEX index1 ON t1(a);
				SET enable_bitmapscan=f;
				CREATE OR REPLACE PROCEDURE proc_while_loop()
				AS
				DECLARE
				i int :=1;
				BEGIN
				WHILE i < 5 LOOP
				INSERT INTO t2 SELECT /*+ indexscan(t1 index1) */* FROM t1 WHERE a = i ;
				i:=i+1;
				END LOOP;
				END;
				/
				SET enable_auto_explain=true;
				SET auto_explain_level=notice;
				SET client_min_messages=log;
				SET enable_stream_operator=on;
				SET query_dop=4;
				CALL proc_while_loop();
				SET query_Dop=6;
				CALL proc_while_loop();

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example

set plan_cache_mode = force_generic_plan	In PBE mode, the gplan is forcibly executed.	In PBE mode, the gplan is not forcibly executed.	Plan cache.	SET enable_auto_explain=true;
				SET auto_explain_level=notice;
				CREATE TABLE t1(a int, b int, c int,d varchar)WITH(storage_type=ustore);
				CREATE TABLE t2(a int, b int, c int,d varchar)WITH(storage_type=ustore);
				INSERT INTO t1 SELECT i,10*random()*i,i%20,'atr'||i from generate_series(1,1000) i;
				CREATE INDEX index1 ON t1(a);
				SET enable_bitmapscan=on;
				SET enable_seqscan=off;
				SET enable_indexscan=off;
				CREATE OR REPLACE PROCEDURE proc_while_loop()
				AS
				DECLARE
				i int :=1;
				BEGIN
				WHILE i < 5 LOOP
				insert into t2 select * from t1 where a = i ;
				i:=i+1;
				END LOOP;
				END;
				/
				CALL proc_while_loop();
				SET plan_cache_mode = force_generic_plan;
				PREPARE aa AS SELECT proc_while_loop();
				EXECUTE aa;
				SET enable_bitmapscan=off;
				EXECUTE aa;

a_format_date_timestamp	CURRENT_DATE returns the timestamp when the current SQL statement is started.	CURRENT_DATE returns the date or date and time when the transaction is started.	Views, default values of function parameters, stored procedure compilation products, and plan cache.	SET a_format_date_timestamp=on;
				CREATE OR REPLACE PROCEDURE proc_while_loop()
				AS
				DECLARE
				i date;
				BEGIN
				i:=current_date;
				raise info 'step2:%',TO_CHAR(current_date, 'MM-DD-YYYY HH24:MI:SS');
				END;
				/
				CALL proc_while_loop();
				PREPARE p1 AS SELECT current_timestamp;
				EXECUTE p1;
				SET a_format_date_timestamp=off;
				CALL proc_while_loop();
				EXECUTE p1;

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example
adjust_systemview_priority	Views with the same name preferentially access user-defined views.	Views with the same name preferentially access system views.	Default value of parameters in the views	CREATE DATABASE database_test DBCOMPATIBILITY = 'ORA'; SELECT datname,datcompatibility,dattimezone FROM pg_database; \c database_test CREATE TABLE t1(c int); CREATE TABLE CREATE VIEW dual AS SELECT * FROM t1; CREATE VIEW SET adjust_systemview_priority = off; SET SELECT * FROM dual; dummy ------- X (1 row) PREPARE p1 as SELECT * FROM dual; PREPARE EXECUTE p1; dummy ------- X (1 row) SET adjust_systemview_priority = on; SET SELECT * FROM dual; c --- (0 rows) EXECUTE p1; dummy ------- X (1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example
mapping_date_to_datea	Replaces the date type with the datea type in ORA-compatible mode.	The default behavior is restored.	Views, default values of function parameters, stored procedure compilation products, and plan cache.	-- By default, the function is enabled in the newly installed database. SHOW mapping_date_to_datea; mapping_date_to_datea ---------- on (1 row) CREATE TABLE t1(c1 date); CREATE TABLE \d+ t1; Table "public.t1" Column | Type | Modifiers | Storage | Stats target | Description --------+-------+-----------+---------+--------------+------------- c1 | datea | | plain | | Has OIDs: no Options: orientation=row, compression=no, storage_type=USTORE, segment=off -- Set the parameter to **off** in the **gaussdb.conf** file. After the database is restarted, the switch is turned off. SHOW mapping_date_to_datea; mapping_date_to_datea ---------- off (1 row) CREATE TABLE t2(c1 date); CREATE TABLE \d+ t2; Table "public.t2" Column | Type | Modifiers | Storage | Stats target | Description --------+-----------------------------+-----------+---------+--------------+------------- c1 | timestamp(0) without time zone | | plain | | Has OIDs: no Options: orientation=row, compression=no, storage_type=USTORE, segment=off

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example
nls_date_format	In ORA-compatible mode, it is used for format control during the conversion between the datea type and the character type. 1. The time format set in **nls_date_format** must be met when the character type is converted to the date type. 2. When the date type is converted to the	The default value of **nls_date_format** is **"DD-MON-RR"**.	Views, default values of function parameters, and plan cache.	-- The default value is **'DD-MON-RR'**. SHOW nls_date_format; nls_date_format ----------------- DD-MON-RR (1 row) -- An error is reported when the input date type does not comply with the format specified by **nls_date_format**. SELECT '2020-10-10'::date; ERROR: invalid value "20-10-10" for "MON" LINE 1: select '2020-10-10'::date; ^ DETAIL: The given value did not match any of the allowed values for this field. CONTEXT: referenced column: datea SET nls_date_format='YYYY-MM-DD'; SET SELECT '2020-10-10'::date; datea ----------------------- 2020-10-10 00:00:00 AD (1 row) -- The input of the to_date function must be in the format specified by **nls_date_format**. Otherwise, an error is reported. SELECT to_date('10-10-2020'); ERROR: Invalid data exists in a string for matching date. CONTEXT: referenced column: to_date SELECT to_date('2020-10-10'); to_date ----------------------- 2020-10-10 00:00:00 AD (1 row) -- When the to_char function is used to convert a date type to a string, the string result is in the format specified by **nls_date_format**. SELECT to_char(to_date('05-2020-10', 'MM-YYYY-DD')); to_char ------------ 2020-05-10 (1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example
	character type, the date type is converted based on the format specified by **nls_date_format**. 3. By default, **to_char** and **to_date** are converted based on the format specified by **nls_date_format**.			

GUC Parameter	Enabled	Disabled	Impact	Example

fixed_date	In ORA-compatible mode, the value of **sysdate** is fixed, and **sysdate** returns the time specified by **fixed_date**.	**sysdate** returns the system time.	Value of **sysdate**.	-- Result returned when **fixed_date** is enabled.
				gaussdb=# SET fixed_date='2024-05-15 11:47:02';
				SET
				gaussdb=# SELECT sysdate;
				timestamp
				--------------------- 	
				2024-05-15 11:47:02
				(1 row)
				
				gaussdb=# PREPARE p1 AS SELECT sysdate;
				PREPARE
				gaussdb=# EXECUTE p1;
				datea
				------------------------ 	
				2024-05-15 11:47:02 AD
				(1 row)
				
				gaussdb=# SET fixed_date='2024-05-16 11:47:02';
				SET
				gaussdb=# SELECT sysdate;
				datea
				------------------------ 	
				2024-05-16 11:47:02 AD
				(1 row)
				
				gaussdb=# EXECUTE p1;
				datea
				------------------------ 	
				2024-05-15 11:47:02 AD
				(1 row)

a_format_func_col_name	In ORA-compatible mode, the projection column name returned by **select user** is changed from **current_user** to **user**.	In ORA-compatible mode, the projection column name returned by **select user** is **current_user**.	Plan cache.	gaussdb=# PREPARE p1 AS SELECT user FROM dual;
				PREPARE
				gaussdb=# EXECUTE p1;
				current_user
				-------------- 	
				omm
				(1 row)
				gaussdb=# SET a_format_func_col_name = 'user';
				SET
				gaussdb=# EXECUTE p1;
				current_user
				-------------- 	
				omm
				(1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example

disable_keyword_options	The behaviors of keywords in each compatibility mode are affected. For details, see **Table 6-38**.	The behaviors of keywords in each compatibility mode are affected. For details, see **Table 6-38**.	Plan cache.	gaussdb=# set disable_keyword_options = 'dbtimezone';
				SET
				gaussdb=# prepare p1 as select 1 dbtimezone;
				PREPARE
				gaussdb=# execute p1;
				dbtimezone
				------------ 	
				1
				(1 row)
				
				gaussdb=# set disable_keyword_options = '';
				SET
				gaussdb=# execute p1;
				dbtimezone
				------------ 	
				1
				(1 row)
				
				gaussdb=# deallocate p1;
				DEALLOCATE

backslash_quote	Enabling the parameter allows single quotation marks in the string text to be represented by \'.	Disabling the parameter does not allow single quotation marks in the string text to be represented by \'.	Plan cache.	gaussdb=# set backslash_quote = 'safe_encoding';
				SET
				gaussdb=# prepare p1 as select text E'Don \'t do that.';
				PREPARE
				gaussdb=# execute p1;
				text
				----------------- 	
				Don 't do that.
				(1 row)
				
				gaussdb=# set backslash_quote = off;
				SET
				gaussdb=# execute p1;
				text
				----------------- 	
				Don 't do that.
				(1 row)
				
				gaussdb=# deallocate p1;
				DEALLOCATE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example

set sql_mode='ansi_quotes'	In M-compatible mode, identifiers support both backquotes (``) and double quotation marks ("").	In M-compatible mode, identifiers support only backquotes (``).	Plan cache.	-- The option of **sql_mode** is changed, which does not cause the plan to be regenerated.
				m_db=# CREATE TABLE t1(a int);
				NOTICE: The 'DISTRIBUTE BY' clause is not specified. Using 'a' as the distribution column by default.
				HINT: Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
				CREATE TABLE
				m_db=# SET sql_mode = 'ansi_quotes';
				SET
				m_db=# PREPARE p1 FROM 'INSERT INTO "t1" VALUES(1)';
				PREPARE
				m_db=# EXECUTE p1;
				INSERT 0 1
				
				m_db=# SET sql_mode='';
				SET
				m_db=# EXECUTE p1;
				INSERT 0 1
				m_db=# DROP TABLE t1;
				DROP TABLE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example

standard_conforming_strings	Backslashes are regarded as ordinary texts as specified in the SQL standard.	Backslashes are regarded as escape characters in the string literal value.	Plan cache.	gaussdb=# SET standard_conforming_strings = on;
				SET
				gaussdb=# PREPARE p1 AS SELECT u&'d\0061t\+000061' AS u&"d\0061t\+000061";
				PREPARE
				gaussdb=# SELECT u&'d\0061t\+000061' AS u&"d\0061t\+000061";
				data
				------ 	
				data
				(1 row)
				
				gaussdb=# EXECUTE p1;
				data
				------ 	
				data
				(1 row)
				
				gaussdb=# SET standard_conforming_strings = off;
				SET
				gaussdb=# SELECT u&'d\0061t\+000061' AS u&"d\0061t\+000061";
				ERROR: unsafe use of string constant with Unicode escapes
				LINE 1: select u&'d\0061t\+000061' as u&"d\0061t\+000061";
				^
				DETAIL: String constants with Unicode escapes cannot be used when standard_conforming_strings is off.
				gaussdb=# EXECUTE p1;
				data
				------ 	
				data
				(1 row)
				
				gaussdb=# DEALLOCATE p1;
				DEALLOCATE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

GUC Parameter	Enabled	Disabled	Impact	Example

set behavior_compat_options = 'ignore_like_default_escape';	This option can be enabled when **standard_conforming_strings** is set to **on**. After this option is enabled, the backslash in the pattern string can be regarded as a common character and be used for fuzzy match.	In LIKE fuzzy match, the backslash in the pattern string is used as an escape character to remove the special meaning of the wildcard.	Plan cache.	gaussdb=# SET standard_conforming_strings = on;
				SET
				gaussdb=# SET behavior_compat_options = 'ignore_like_default_escape';
				SET
				gaussdb=# CREATE TABLE test(c1 text);
				CREATE TABLE
				gaussdb=# INSERT INTO test values('aa\a');
				INSERT 0 1
				gaussdb=# PREPARE p1 AS SELECT * FROM test WHERE c1 LIKE 'aa\a';
				PREPARE
				gaussdb=# EXECUTE p1;
				  c1
				------ 	
				 aa\a
				(1 row)
				
				gaussdb=# SELECT * FROM test WHERE c1 LIKE 'aa\a';
				  c1
				------ 	
				 aa\a
				(1 row)
				
				gaussdb=# SET behavior_compat_options = '';
				SET
				gaussdb=# EXECUTE p1;
				  c1
				------ 	
				 aa\a
				(1 row)
				
				gaussdb=# SELECT * FROM test WHERE c1 LIKE 'aa\a';
				 c1
				---- 	
				(0 rows)
				
				gaussdb=# DEALLOCATE p1;
				DEALLOCATE

6.3.2 File Location

After a database has been installed, three configuration files (**gaussdb.conf**, **gs_hba.conf**, and **gs_ident.conf**) are automatically generated and saved in the data directory. You can use the methods described in this section to change the names and save paths of these configuration files.

When changing the storage directory of a configuration file, set **data_directory** in **gaussdb.conf** to the actual data directory.

NOTE

If a configuration file is incorrectly modified, the database will be seriously affected. Do not modify the configuration files mentioned in this section after installation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

data_directory

Parameter description: Specifies the GaussDB **data** directory. Only the SYSADMIN user can access this parameter.

Parameter type: string.

Unit: none

Value range: valid path of a directory.

Default value: Specify this parameter during installation. If this parameter is not specified during installation, the database is not initialized by default.

Setting method: This parameter can be set as follows:

- Set it when GaussDB is installed.
- This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

config_file

Parameter description: Specifies the configuration file (**gaussdb.conf**) of the primary node.

Parameter type: string.

Unit: none

Value range: a string of more than 0 characters.

Default value: **gaussdb.conf** (The absolute directory of this file may be displayed in the actual situation.)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

hba_file

Parameter description: Specifies the configuration file (**gs_hba.conf**) for host-based authentication (HBA). This parameter can be specified only in the **gaussdb.conf** file and can be accessed only by the SYSADMIN user.

Parameter type: string.

Unit: none

Value range: a string of more than 0 characters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **gs_hba.conf** (The absolute directory of this file may be displayed in the actual situation.)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised not to modify the parameters in the configuration file.

Risks and impacts of improper settings: The cluster upgrade and backup functions may be affected.

ident_file

Parameter description: Specifies the name of the configuration file (**gs_ident.conf)** for client authentication. Only the SYSADMIN user can access this parameter.

Parameter type: string.

Unit: none

Value range: a string of more than 0 characters.

Default value: **gs_ident.conf** (The absolute directory of this file may be displayed in the actual situation.)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised not to modify the parameters in the configuration file.

Risks and impacts of improper settings: The cluster upgrade and backup functions may be affected.

external_pid_file

Parameter description: Specifies the extra PID file that can be used by the server management program. Only the SYSADMIN user can access this parameter.

Parameter type: string.

Unit: none

Value range: valid file name.

Default value: **""**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

NOTE

This parameter takes effect only after the database restarts.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3.3 Connection and Authentication

6.3.3.1 Connection Settings

This section describes parameters related to client-server connection modes.

light_comm

Parameter description: Specifies whether the server uses the lightweight communications mode.

This parameter specifies whether the server uses the communication mode based on lightweight locks and non-blocking sockets.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The lightweight communications mode is used.
- **off**: The lightweight communications mode is not used.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

listen_addresses

Parameter description: Specifies the TCP/IP address of the client for a server to listen on.

This parameter specifies the IP address used by the GaussDB server for listening, for example, an IPv4 address or IPv6 address. Multiple NICs may exist on the host and each NIC can be bound to multiple IP addresses. This parameter specifies the IP addresses to which GaussDB is bound. The client can use the IP addresses specified by this parameter to connect to GaussDB or send requests to GaussDB.

Parameter type: string.

Unit: none

Value range:

- Host name or IP address. Multiple values are separated with commas (,).
- Asterisk (*), **::**, or **0.0.0.0**, indicating that all IP addresses will be listened on, which is not recommended due to potential security risks.
- If the parameter is not specified, the server does not listen on any IP address. In this case, only UDSs can be used for database connections.

Default value:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

After the cluster is installed, configure different default values based on the IP addresses of different instances in the **public_cloud.conf** file. The default value of CN is **"localhost,***IP address of the mgr.net NIC,IP address of the data.net NIC,IP address of the virtual.net NIC***"**. The default value of DN is **"***IP address of the data.net NIC***"**.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

NOTE

localhost indicates that only local loopback is allowed.

The **public_cloud.conf** file contains the following NIC information: **mgr.net** (management NIC), **data.net** (data NIC), and **virtual.net** (virtual NIC).

If an IPv6 address is used and the IP address is a fe80 address block, add '%zone index' during the configuration.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

listen_address_ext

Parameter description: Specifies the extended TCP/IP address of the client for a server to listen on. This parameter specifies the IP addresses used by the GaussDB server as the extended listening IP address.

Parameter type: string.

Unit: none

Value range: an IPv4 or IPv6 address. Multiple IP addresses are not supported.

Default value: **"localhost"**.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: In BMS deployment mode, this parameter is automatically configured when the cluster is installed and started. You are advised not to manually configure this parameter.

NOTE

- The setting of the **listen_address_ext** parameter involves the network topology and configuration of the extended IP addresses of the entire cluster. Therefore, it must be set and reloaded by using the OM tool.
- **listen_address_ext** is a network channel used to extend specific IP addresses. If this parameter is set to **"*"**, **"0.0.0.0"**, **"localhost"**, **"::-1"**, or **"127.0.0.1"**, the setting does not take effect. If required, you are advised to set these addresses in **listen_addresses**.
- **listen_address_ext** is used to configure distributed DNs. It does not take effect if it is configured for CNs.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

local_bind_address

Parameter description: Specifies the host IP address bound to the current node for connecting to other nodes in the cluster.

Parameter type: string.

Unit: none

Value range: an IPv4 or IPv6 address. Multiple IP addresses are not supported.

Default value:

After the cluster is installed, configure different default values based on the IP addresses of different instances in the **public_cloud.conf** file. The default value of CN/DN is "*IP address of the data.net NIC*".

NOTE

The **public_cloud.conf** file contains the following NIC information: **mgr.net** (management NIC), **data.net** (data NIC), and **virtual.net** (virtual NIC).

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impact of improper settings: This parameter is specified in the configuration file during installation. Do not modify this parameter unless necessary. Otherwise, database communications will be affected after the modification.

port

Parameter description: Specifies the TCP port listened on by the GaussDB.

Parameter type: integer.

Unit: none

Value range: 1 to 65535

NOTE

- When setting the port number, ensure that the port number is not in use. When setting the port numbers of multiple instances, ensure that the port numbers do not conflict.
- Ports 1 to 1023 are reserved for the OS. Do not use them.
- When the cluster is installed using the configuration file, pay attention to the ports reserved in the communication matrix in the configuration file. For example, the DN needs to reserve **dataPortBase** + 1 as the port used by internal tools. Therefore, during cluster installation, the maximum value of **port** is **65532** for CNs, **65529** for DNs, and **65534** for GTMs. Ensure that the port number does not conflict with each other.

Default value: **5432** (The actual value is specified in the configuration file during installation.)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impact of improper settings: This parameter is specified in the configuration file during installation. Do not modify this parameter unless necessary. Otherwise, database communications will be affected after the modification.

max_connections

Parameter description: Specifies the maximum number of concurrent connections to the database. This parameter influences the concurrent processing capability of the cluster.

Parameter type: integer.

Unit: none

Value range: The minimum value is **10** (greater than the value of **max_wal_senders**). The theoretical maximum value is **262143**. The actual maximum value is a dynamic value, which is calculated using the formula: 262143 – **job_queue_processes** – **autovacuum_max_workers** – **max_inner_tool_connections** – **max_concurrent_autonomous_transactions** – **AUXILIARY_BACKENDS** – **AV_LAUNCHER_PROCS** – *min(max(newValue/4,64),1024)*. The values of **job_queue_processes**, **autovacuum_max_workers**, **max_inner_tool_connections**, and **max_concurrent_autonomous_transactions** depend on the settings of the corresponding GUC parameters. **AUXILIARY_BACKENDS** indicates the number of reserved auxiliary threads, which is fixed to **20**. **AV_LAUNCHER_PROCS** indicates the number of reserved autovacuum launcher threads, which is fixed to **2**. In *min(max(newValue/4,64),1024)*, **newValue** indicates the new value.

For different memory specifications, value range of this parameter in different instances is as follows:

Table 6-4 Value ranges of memory specifications for different instances in independent deployment mode

Memory Specifications	Value Range for CNs	Value Range for DNs
< 32 GB	[10,100]	[10,100]
[32GB,64GB)	[10,200]	[10,200]
[64GB,128GB)	[10,1000]	[10,2500]
[128GB,256GB)	[10,2000]	[10,6000]
[256GB,480GB)	[10,4000]	[10,12000]
≥ 480 GB	[10,8000]	[10,24000]

Default value:

- Independent deployment:

  CN: **8000** (128-core CPU/1024 GB memory, 60-core CPU/480 GB memory); **4000** (32-core CPU/256 GB memory); **2000** (16-core CPU/128 GB memory);

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

1000 (8-core CPU/64 GB memory); **100** (4-core CPU/32 GB memory, 4-core CPU/16 GB memory)

DN: **24000** (128-core CPU/1024 GB memory, 60-core CPU/480 GB memory); **12000** (32-core CPU/256 GB memory); **6000** (16-core CPU/128 GB memory); **2500** (8-core CPU/64 GB memory); **100** (4-core CPU/32 GB memory, 4-core CPU/16 GB memory)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value of this parameter in the primary database node file. If this parameter is set to a smaller value, the number of available connections decreases. Before changing the value of this parameter, understand the parameter description and exercise caution to avoid risks caused by misoperations.

Risks and impacts of improper settings:

- If the value of **max_connections** exceeds the maximum dynamic value, the CN and DN fail to be started and the error message "invalid value for parameter "max_connections"" is displayed. Alternatively, the memory fails to be allocated during the node startup and the error message "Cannot allocate memory" is displayed.
- If the value of **max_connections** is not increased based on the external export specifications, the memory parameter is not adjusted in the same proportion. When the service load is heavy, the memory may be insufficient and the error message "memory is temporarily unavailable" is displayed.
- In the hybrid deployment scenario, if the value of **max_connections** on the CN is greater than the value of **max_connections** divided by the number of CNs on the DN and the total pressure on the client exceeds the value of **max_connections** on the DN, the CN may fail to connect to the DN, and the error message "pooler... Too many clients already" is displayed.
- If the value of **thread_pool_attr** is greater than or equal to that of **max_connections**, all proc resources are occupied by the thread pool. As a result, proc resources are insufficient.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If the number of connections of the administrator exceeds the value of **max_connections**, the administrator can still connect to the database after the connections are used up by common users. If the number of connections exceeds the value of **sysadmin_reserved_connections**, an error is reported. That is, the maximum number of connections of an administrator is equal to the value of **max_connections** plus the value of **sysadmin_reserved_connections** plus the value of **sysadmin_reserved_connections** (number of dedicated reserved connections logged in through the port whose port number is the master port number plus 1).
- For common users, internal jobs use some connections. Therefore, the value of this parameter is slightly less than that of **max_connections**. The value depends on the number of internal connections.
- After the thread pool is enabled, the maximum number of stream threads is determined by the value of **max_connections**. If the number of stream threads reaches the upper limit, the error message "Exceed stream thread pool limitation..." is reported. In this case, you can increase the value of **max_connections** to increase the upper limit. This is a POSTMASTER parameter. Therefore, you can estimate the number of stream threads based on service requirements. The formula is as follows: Total number of stream threads = Number of concurrent services x Number of stream threads consumed by each concurrently executed statement (which can be viewed in the execution plan).
- **max_connections** indicates the number of node connections, including external and internal connections.
  - External connection: connection of a user application.
  - Internal connection: connection generated by an internal job.
- When DDL operations are used in a transaction, the CN establishes connections with other nodes in the cluster for synchronization until the transaction ends. The connections established by the CN occupy the number of connections on the corresponding node. If *N* concurrent DDL operations are performed in this scenario, the number of internal connections on a node increases by *N*. As a result, the number of connections available to external clients on the node decreases by *N*. For example, if gsql is used to create three connections on CN1 and DDL operations are performed in a transaction, CN1 establishes three internal connections on other nodes, that is, CN2 has three new internal connections from CN1. As a result, the number of connections available to external clients on CN2 decreases by three.

max_inner_tool_connections

Parameter description: Specifies the maximum number of concurrent connections of a tool which is allowed to connect to the database. This parameter influences the concurrent connection capability of the GaussDB tool.

Parameter type: integer.

Unit: none

Value range: The minimum value is **1** and the calculation formula of the maximum value is 262143 – **job_queue_processes** – **autovacuum_max_workers** – **max_connections** – **max_concurrent_autonomous_transactions** – **AUXILIARY_BACKENDS** – **AV_LAUNCHER_PROCS** – **min(max(max_connections/4,64),1024)**. The values of **job_queue_processes**, **autovacuum_max_workers**, **max_connections**, **max_concurrent_autonomous_transactions** depend on the settings of the corresponding GUC parameters. **AUXILIARY_BACKENDS** indicates that the number of reserved auxiliary threads is fixed to **20**. **AV_LAUNCHER_PROCS** reserves two launcher threads for autovacuum.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **50**. If the default value is greater than the maximum value supported by the kernel (determined when the **gs_initdb** command is executed), an error message is displayed.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value of this parameter in the primary database node file.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

sysadmin_reserved_connections

Parameter description: Specifies the minimum number of connections reserved for an administrator, which is also the minimum number of connections reserved for the administrator to log in through the port whose port number is the master port plus 1. You are advised not to set this parameter to a large value. This parameter is used together with **max_connections**. The maximum number of connections for an administrator is equal to the value of **max_connections** plus the value of **sysadmin_reserved_connections** plus the value of **sysadmin_reserved_connections** (number of dedicated reserved connections logged in through the port whose port number is the master port plus 1).

NOTE

- When the thread pool function is enabled, if the thread pool is fully occupied, a processing bottleneck occurs. As a result, connections reserved by the administrator cannot be established. In this case, you can use gsql to establish connections through the port specified by the master port number plus 1 and clear useless sessions.
- When the reserved connections are used up, new connections will fail. In this case, you can only restart the cluster to restore the connections. Therefore, exercise caution when using the reserved connections.

Parameter type: integer.

Unit: none

Value range: 0 to *MIN(262143, max_connections)* (the smaller value between **262143** and **max_connections**). For details about how to calculate the value of **max_connections**, see **max_connections**.

Default value: **3**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

service_reserved_connections

Parameter description: Specifies the minimum number of connections reserved for background O&M users (with the persistence attribute). A large value is not recommended. This parameter is used together with **max_connections**. The

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

maximum number of connections of an O&M user can be calculated as follows: **max_connections** + **service_reserved_connections**.

Parameter type: integer.

Unit: none

Value range: 0 to 262143

Default value: **10**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a small value, the O&M user (with the persistence attribute) cannot connect to the database and jobs cannot be executed when the number of connections reaches the maximum (**max_connections**). If the value is too large, there are too many idle resources.

extip_reserved_connections

Parameter description: Specifies the minimum number of common gsql service connections reserved for a user on the extended IP address specified by **listen_address_ext**. This parameter is used together with **max_connections**. The maximum number of connections of a corresponding service can be calculated as follows: **max_connections** + **service_reserved_connections**.

NOTE

This parameter is valid only when a specific IP address of **listen_address_ext** is configured for a distributed DN.

Parameter type: integer.

Unit: none

Value range: 0 to *MIN(262143, max_connections)* (the smaller value between **262143** and **max_connections**). For details about how to calculate the value of **max_connections**, see **max_connections**.

Default value: **5**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

unix_socket_directory

Parameter description: Specifies the UDS directory for the GaussDB server to listen to connections from the client. Only the SYSADMIN user can access this parameter.

Parameter type: string.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range: valid path of a directory.

NOTE

The parameter length limit varies by the maximum length of a path of a directory in the OS. In the Linux OS, the length of a socket path name (combination of a socket directory and a socket file name) cannot exceed 107 bytes, and the length of a directory cannot exceed 92 bytes. If this limit is exceeded, the error "Unix-domain socket path "xxx" is too long" will be reported.

Default value: **""**. The actual value is specified by **tmpMppdbPath** in the configuration file during installation.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impact of improper configuration: If the configuration is incorrect (for example, the length exceeds the limit or the directory is invalid), the process cannot be started properly. You can retrieve the system_call log in the CM Agent directory to locate configuration issues.

unix_socket_group

Parameter description: Specifies the group of the UDS (the user of a socket is the user who starts the server). This parameter can work with **unix_socket_permissions** to control socket access.

Parameter type: string.

Unit: none

Value range: existing user groups in the system.

Default value: **""**, indicating the default group of the current user.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

unix_socket_permissions

Parameter description: Specifies access permissions on the UDS.

The UDS uses the usual permission set of the Unix file system. The value of this parameter should be a number (acceptable for the **chmod** and **umask** commands). If a user-defined octal format is used, the number must start with 0.

Parameter type: integer.

Unit: none

Value range: an octal integer ranging from 0000 to 0777 (that is, 0 to 511 in the decimal format)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> In the Linux OS, a document has one document attribute and nine permission attributes, which are the read (r), write (w), and execute (x) permissions of the **owner**, **group**, and **others** groups.
>
> The r, w, and x permissions are represented by the following numbers:
>
> r: 4
>
> w: 2
>
> x: 1
>
> -: 0
>
> The three attributes in a group are accumulative.
>
> For example, **-rwxrwx---** indicates the following permissions:
>
> owner = rwx = 4+2+1 = 7
>
> group = rwx = 4+2+1 = 7
>
> others = --- = 0+0+0 = 0
>
> The permission of the file is 0770.

Default value: **0700** in octal (which is **448** in decimal)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised to set it to **0770** (only allowing access from users connecting to the database and users in the same group as them) or **0700** (only allowing access from users connecting to the database).

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

application_name

Parameter description: Specifies the client name used in the current connection request.

Parameter type: string.

Unit: none

Value range: valid client name. The actual query result depends on the client used for queries or user configurations.

> **NOTE**
>
> When a standby node requests to replicate logs on the primary node, if this parameter is not an empty string, it is used as the name of the streaming replication slot of the standby node on the primary node. In this case, if the length of this parameter exceeds 61 bytes, only the first 61 bytes are used as the streaming replication slot name.

Default value: **""** (The actual value is the name of the application connected to the backend.)

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. Note that after the client is connected, this parameter is set to the client name and the client level. Therefore, only the session-level parameters take effect. Other methods do not take effect because they are lower than the client level.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The database depends on **application_name** to identify some internal processes. If you set **application_name** to the same name as an internal process, unexpected behavior may occur.

connection_info

Parameter description: Specifies the database connection information, including the driver type, driver version, driver deployment path, and process owner.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **""**, indicating that the driver connected to the database does not support automatic setting of the **connection_info** parameter or the parameter is not set by users in applications.

NOTE

The following is an example of the concatenated value of **connection_info**:
{"driver_name":"ODBC","driver_version": "(GaussDB XXX.XXX.XXX build 290d125f) compiled at 2020-05-08 02:59:43 last mr 131 release","driver_path":"/usr/local/lib/gsqlodbcw.so","os_user":"omm"}

By default, **driver_name** and **driver_version** are displayed. The display of **driver_path** and **os_user** is controlled by the connection parameter **connectionExtraInfo** in the connection string. For details, see "Application Development Guide > Development Based on JDBC > Development Procedure > Connecting to a Database > Connection Parameter Reference" in *Developer Guide* and "Application Development Guide > Appendix > ODBC > Configuration Parameters" in *Developer Guide*.

Setting method: This is a USERSET parameter used for O&M. You are advised not to set the parameter. This parameter can be set only in a session. To change the parameter value, see "Method 3" in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

backend_version

Parameter description: Specifies the version number of the synchronous connection between CNs or between a CN and a DN. This parameter involves the version number and cannot be set randomly.

Parameter type: integer.

Unit: none

Value range: 0 to 100000

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the settings are incorrect, the versions of different nodes are inconsistent.

check_disconnect_query

Parameter description: Specifies whether to terminate the execution of statements on the GaussDB server after the client is disconnected abnormally (for example, socketTimeout is triggered by JDBC, rwtimeout is triggered by libpq and the connection is closed, or the client process is terminated during service running).

Parameter type: Boolean.

Unit: none

Value range:

- **on**: indicates that the GaussDB server stops running the corresponding statements after the client is disconnected unexpectedly.
- **off**: indicates that the GaussDB server does not stop running the corresponding statements after the client is disconnected unexpectedly.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

connection_reject_nodes

Parameter description: Specifies the blacklist of CNs/DNs that reject connections from other CNs. The blacklists are separated by commas (,), for example, **'cn_5001,cn_5002'**. When a CN (for example, **cn_5001**) in the blacklist attempts to connect to the local node, the connection will be rejected and the error message "Reject cn_5001 connection" will be displayed.

Parameter type: string.

Unit: none

Value range: a character string consisting of several node names and commas (,).

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a node name, other nodes will reject the connection request. As a result, a connection error is reported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_server_listener_thread

Parameter description: Specifies whether to enable the independent listening thread. When this function is enabled, an independent thread listens to the TCP port (**port**) listened by the GaussDB Kernel service. To enable the independent listening thread, set the GUC parameters **enable_thread_pool** and **enable_connect_thread_pool** to **on**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: An independent thread listens to the TCP port (**port**) listened by the GaussDB Kernel service.
- **off**: The postmaster thread of the service instance process listens to the TCP port (**port**) listened by the GaussDB Kernel service.

Default value:

- For a newly installed database: **on**.
- For a database upgraded from a version earlier than 507.0.0: **off**.
- For an upgrade from 507.0.0 or later: original parameter value.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Enabling the independent listening thread will add running threads. If hardware resources are limited, you are advised to determine whether to enable this function based on the actual test result.

Risks and impacts of improper settings: If this function is not enabled, when the postmaster thread competes with other threads, the efficiency of the service listening port may decrease, affecting the time required for establishing a connection with the client. Before the modification, fully verify the change in a test environment.

6.3.3.2 Security and Authentication

This section describes parameters about client-to-server authentication.

authentication_timeout

Parameter description: Specifies the longest duration to wait before the client authentication times out. If a client is not authenticated by the server within the period, the server automatically disconnects from the client so that the client does not occupy connection resources.

Parameter type: integer.

Unit: seconds

Value range: 1–600

Default value: **60**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **600** without a unit, **authentication_timeout** indicates 600s. If the value is **10min**, **authentication_timeout** indicates 10 minutes. If the unit is required, the value must be **s**, **min**, **h**, or **d**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, connection failures may occur frequently, affecting user experience. If the value is too large, security risks increase and resources are wasted.

auth_iteration_count

Parameter description: Specifies the number of iterations during the generation of encryption information for authentication.

Parameter type: integer.

Unit: none

Value range: 2048–134217728

Default value: **10000**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set the number of iterations based on actual hardware conditions. You are advised to retain the default value.

Risks and impacts of improper settings: If the number of iterations is too small, the password storage security is reduced. If the number of iterations is too large, the performance deteriorates in scenarios involving password encryption, such as authentication and user creation.

session_authorization

Parameter description: Specifies the user ID of the current session. The value of this parameter can be queried only by running the **SHOW** statement.

Parameter type: string.

Unit: none

Value range: a valid user ID.

Default value: **""**

Setting method: This is a USERSET parameter. Set it based on the syntax provided in **SET SESSION AUTHORIZATION**. If you directly set the value, it can only be set to **DEFAULT**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

session_timeout

Parameter description: Specifies the longest duration allowed when no operations are performed on a client after it is connected to the server. If this parameter is not set to **0** and a user does not perform any operation for a period longer than the value of this parameter, the server is disconnected.

Parameter type: integer

Unit: seconds

Value range: 0–86400 (1 day)

- The value **0** indicates that the timeout setting is disabled.
- A positive number indicates the maximum duration in which no operation is performed after the connection to the server is set up. When the value of this parameter is exceeded, the client is disconnected from the server.

Default value: **1800**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **600** without a unit, **session_timeout** indicates 600s. If the value is **10min**, **session_timeout** indicates 10 minutes. If the unit is required, the value must be **s**, **min**, **h**, or **d**.

Setting suggestion: Set this parameter based on service security requirements. A smaller value indicates higher security but more inconvenient use.

Risks and impacts of improper settings: If the value is too small, persistent connections in the service connection pool may become invalid, causing service failures.

NOTE

The gsql client of GaussDB has an automatic reconnection mechanism. For local connection of initialized users, the client reconnects to the server if the connection breaks after the timeout.

idle_in_transaction_timeout

Parameter description: Specifies the maximum idle duration of a transaction after the connection to the server is established and the transaction is started. If this parameter is not set to **0**, the transaction is idle, and no operation is performed for a period longer than the value of this parameter, the server is disconnected.

Parameter type: integer

Unit: seconds

Value range: 0–86400 (1 day)

- The value **0** indicates that the timeout setting is disabled.
- The positive number indicates the maximum idle duration of a transaction after the connection to the server is established and the transaction is started. When the value of this parameter is exceeded, the client is disconnected from the server.

Default value: **300** (5 minutes)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **600** without a unit, **idle_in_transaction_timeout** indicates 600s. If the value is **10min**, **idle_in_transaction_timeout** indicates 10 minutes. If the unit is required, the value must be **s**, **min**, **h**, or **d**.

Setting suggestion: Set this parameter based on service security requirements. A smaller value indicates higher security but more inconvenient use.

Risks and impacts of improper settings: If the value is too small, the client application may not send new requests in a timely manner after a transaction is started. As a result, the session times out and exits, affecting user experience.

NOTE

- The gsql client of GaussDB has an automatic reconnection mechanism. For local connection of initialized users, the client reconnects to the server if the connection breaks after the timeout.
- If both **idle_in_transaction_timeout** and **session_timeout** are enabled, the timeout interval takes effect regardless of whether the transaction is idle. The smaller timeout interval prevails.
- For a newly delivered instance, the value of this parameter is **300**. For an upgraded instance, the value of this parameter is **0** by default.
- After an idle transaction times out, the transaction start time, thread ID, application name, and query information is recorded in logs. This log file is used to locate the problem that an idle transaction exits due to timeout. The log file does not contain the prefix format **log_line_prefix**.

ssl

Parameter description: Specifies whether to enable the SSL connection on the server. Before using this option, read "Database Quick Start > Connecting to a Database > Using gsql to Connect to a Database" in *Developer Guide*.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The SSL connection is enabled on the server. Whether SSL encrypted transmission is used during actual communication depends on the client configuration.
- **off**: The SSL connection is disabled on the server.

Default value: **on** (for CNs) or **off** (for DNs)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If high security is required, you are advised to enable this function. If high performance is required, you are advised to disable this function. GaussDB supports SSL when a client connects to a CN. You are advised to enable SSL connections only on CNs. The default value is **off** on DNs.

Risks and impacts of improper settings: Before enabling this parameter, ensure that the parameters **ssl_cert_file**, **ssl_key_file**, and **ssl_ca_file** are correctly set. If SM cryptographic algorithms are used, ensure that the parameters

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

ssl_enc_cert_file and **ssl_enc_key_file** are correctly set. Incorrect settings may cause cluster startup failures.

NOTE

The SSL communication between the primary and standby nodes is also affected by the GUC parameter **wal_replication_ssl**.

comm_ssl

Parameter description: Specifies whether to enable the SSL connection between primary DNs. Before using this option, read "Database Quick Start > Connecting to a Database > Using gsql to Connect to a Database" in *Developer Guide*.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The SSL connection is enabled between primary DNs.
- **off**: The SSL connection is disabled between primary DNs.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised to enable this parameter only on DNs. The default value is **off** on CNs.

Risks and impacts of improper settings: To enable SSL connections, you also need to ensure that parameters such as **ssl_cert_file**, **ssl_key_file**, and **ssl_ca_file** are configured correctly. Incorrect configurations may cause startup failure of the cluster.

require_ssl

Parameter description: Specifies whether the server requires SSL connections. This parameter is valid only when **ssl** is set to **on**. Before using this option, read "Database Quick Start > Connecting to a Database > Using gsql to Connect to a Database" in *Developer Guide*.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The server requires SSL connections.
- **off**: The server does not require SSL connections.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: GaussDB supports SSL when a client connects to a CN. It is recommended that the SSL connection be enabled only on CNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper setting: If SSL-related parameters are incorrectly configured, enabling this parameter may cause communication connection setup failures or even cluster startup failures.

ssl_ciphers

Parameter description: Specifies the list of algorithms supported by SSL. Only the SYSADMIN user can access the list.

Parameter type: string.

Unit: none

Value range: a string. Separate multiple algorithms by semicolons (;).

Default value: **"ALL"**, indicating that all supported algorithms (excluding ADH, LOW, EXP, and MD5) can be used for the peer end.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised not to specify the insecure algorithms DHE-RSA-AES128-GCM-SHA256 and DHE-RSA-AES256-GCM-SHA384.

Risks and impacts of improper setting: If **ssl_ciphers** is incorrectly set, the database cannot be started.

ssl_renegotiation_limit

Parameter description: Specifies the allowed traffic volume over an SSL-encrypted channel before the session key is renegotiated. The renegotiation mechanism reduces the probability that attackers use the password analysis method to crack the key based on a huge amount of data but causes big performance losses. The traffic indicates the sum of transmitted and received traffic.

Parameter type: integer.

Unit: KB

Value range: 0 to 2147483647. **0** indicates that the renegotiation mechanism is disabled.

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1024** without a unit, **ssl_renegotiation_limit** indicates 1024 KB. If the value is **1MB**, **ssl_renegotiation_limit** indicates 1 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Retain the default value. The SSL renegotiation mechanism has been disabled because of potential risks. This parameter is reserved for version compatibility and does not take effect.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

ssl_cert_file

Parameter description: Specifies the name of the file that contains the SSL server certificate. The path is relative to the data directory.

Parameter type: string.

Unit: none

Value range: valid file name, which must end with .crt.

Default value: **"server.crt"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: When configuring the file path, ensure that the file path matches other SSL-related certificate and private key configuration files.

Risks and impacts of improper settings: If the file path is incorrect, the SSL connection cannot be established between the client and the database node, and the database cluster nodes cannot communicate with each other in SSL mode. As a result, data leakage may occur.

ssl_key_file

Parameter description: Specifies the name of the file that contains the SSL private key. The path is relative to the data directory.

Parameter type: string.

Unit: none

Value range: valid file name, which must end with .key.

Default value: **"server.key"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: When configuring the file path, ensure that the file path matches other SSL-related certificate and private key configuration files.

Risks and impacts of improper settings: If the file path is incorrect, the SSL connection cannot be established between the client and the database node, and the database cluster nodes cannot communicate with each other in SSL mode. As a result, data leakage may occur.

ssl_enc_cert_file

Parameter description: Specifies the name of the SSL server certificate file that is encrypted using SM cryptographic algorithms. The path is relative to the data directory.

Parameter type: string.

Unit: none

Value range: valid file name, which must end with .crt.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: ""

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: When configuring the file path, ensure that the file path matches other certificates and private key configuration files related to SSL that uses SM cryptographic algorithms.

Risks and impacts of improper settings: If the file path is incorrect, the SSL connection that uses SM cryptographic algorithms cannot be established between the client and the database node, and the cluster nodes of the database instance cannot communicate with each other in SSL that uses SM cryptographic algorithms. As a result, data leakage may occur.

ssl_enc_key_file

Parameter description: Specifies the name of the SSL server certificate file (with the private key) that is encrypted using SM cryptographic algorithms. The path is relative to the data directory.

Parameter type: string.

Unit: none

Value range: valid file name, which must end with .key.

Default value: ""

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: When configuring the file path, ensure that the file path matches other certificates and private key configuration files related to SSL that uses SM cryptographic algorithms.

Risks and impacts of improper settings: If the file path is incorrect, the SSL connection that uses SM cryptographic algorithms cannot be established between the client and the database node, and the cluster nodes of the database instance cannot communicate with each other in SSL that uses SM cryptographic algorithms. As a result, data leakage may occur.

ssl_ca_file

Parameter description: Specifies the name of the root certificate that contains CA information. Its path is relative to the data directory.

Parameter type: string.

Unit: none

Value range: valid file name, which must end with .pem. An empty character string indicates that no CA file is loaded and client certificate verification is not performed.

Default value: **"cacert.pem"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: If client certificate verification is required, configure the file path correctly.

Risks and impacts of improper settings: If the file path is incorrect, the database cannot verify the client certificate, reducing communication security.

ssl_crl_file

Parameter description: Specifies the name of the file that contains the certificate revocation list (CRL). If a client certificate is in the list, the certificate is invalid. The path is relative to the data directory.

Parameter type: string.

Unit: none

Value range: valid file name. An empty string indicates that there is no CRL.

Default value: **""**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the file name in the CRL is incorrectly configured, invalid client certificates may be used to establish SSL connections, causing data leakage risks. Valid client certificates are considered revoked and SSL connections cannot be established.

enable_shared_sslctx

Parameter description: Specifies whether to enable global SSL certificate loading and SSL context sharing. This parameter is valid only when **SSL** is set to **on**.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Global SSL certificate loading is enabled and the SSL context is initialized only once when the database is started.
- **off**: Global SSL certificate loading is disabled and the SSL context is initialized each time a session is established.

Default value: **off**. If the glibc version of an OS is earlier than 2.28, the value is **on**.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised to disable this function. You are advised to enable this function only when high performance is required in SSL non-two-way authentication scenarios.

Risks and impacts of improper settings: After the global SSL certificate loading function takes effect, the new certificate cannot take effect in real time. You need to restart the database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If the glibc version of an OS is earlier than 2.28, the certificate loading efficiency is low. Therefore, the default value of this parameter is **on** in these OSs.
- If the SSL global context fails to be initialized during database startup, this parameter is automatically set to **off**, and the session-level SSL certificate is loaded.

ssl_cert_notify_time

Parameter description: Specifies the number of days prior to SSL server certificate expiration that a user will receive a reminder. When the SSL certificate is initialized during connection establishment, if the duration from the current time to the certificate expiration time is shorter than the specified value, an expiration notification is recorded in the log.

Parameter type: integer.

Unit: day

Value range: 7–180

Default value: **90**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The unit must be **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the server certificate expiration may not be detected in advance. As a result, the certificate cannot be replaced in a timely manner. If the value is too large, alarms may be frequently generated.

krb_server_keyfile

Parameter description: Specifies the location of the main configuration file of the Kerberos service. Only the SYSADMIN user can access the file.

Parameter type: string.

Unit: none

Value range: valid file path.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Contact technical support for assistance.

Risks and impacts of improper settings: If this parameter is incorrectly set, nodes in the cluster cannot use the Kerberos authentication function.

krb_srvname

Parameter description: Specifies the Kerberos service name.

Parameter type: string.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range: a string.

Default value: **"postgres"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

krb_caseins_users

Parameter description: Specifies whether the Kerberos username is case-sensitive.

Parameter type: Boolean

Unit: none

Value range:

- **on**: It is case-insensitive.
- **off**: It is case-sensitive.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

modify_initial_password

Parameter description: After GaussDB is installed, there is only one initial user account (whose UID is 10) in the database. When a user logs in to the database using this initial account for the first time, this parameter specifies whether the password of the initial account needs to be modified.

NOTE

If the initial user password is not specified during the installation, the initial user password is empty by default after the installation. Before performing other operations, you need to set the initial user password using the gsql client. This parameter no longer takes effect and is reserved only for compatibility with upgrade scenarios.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The password of the initial account needs to be modified upon the first login after the cluster is installed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **off**: The password of the initial account does not need to be modified after the cluster is installed.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

password_policy

Parameter description: Specifies whether to check the password complexity when you run the **CREATE ROLE/USER** or **ALTER ROLE/USER** command to create or modify the GaussDB account.

Parameter type: integer.

Unit: none

Value range: **0** or **1**

- **0**: No password complexity policy is enabled.
- **1**: The password complexity policy is enabled.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: For security purposes, do not disable the password complexity policy.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

password_reuse_time

Parameter description: Specifies whether to check the reuse interval of the new password when you run the **ALTER USER** or **ALTER ROLE** command to change a user password.

NOTE

When you change the password, the system checks the values of **password_reuse_time** and **password_reuse_max**.

- If the values of **password_reuse_time** and **password_reuse_max** are both positive numbers, an old password can be reused when it meets either of the reuse conditions
- If the value of **password_reuse_time** is **0**, password reuse is restricted based on the number of reuse times, and not on the reuse interval.
- If the value of **password_reuse_max** is **0**, password reuse is restricted based on the reuse interval, and not on the number of reuse times.
- If the values of both **password_reuse_time** and **password_reuse_max** are **0**, password reuse is not restricted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: floating-point number

Unit: day

Value range: 0 to 3650

- **0** indicates that the password reuse interval is not checked.
- A positive number indicates that a new password cannot be chosen from passwords in history that are newer than the specified number of days.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service security requirements. A larger value indicates higher security but more inconvenient use.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

password_reuse_max

Parameter description: Specifies whether to check the reuse times of the new password when you run the **ALTER USER** or **ALTER ROLE** command to change a user password. Only the SYSADMIN user can access this parameter.

NOTE

When you change the password, the system checks the values of **password_reuse_time** and **password_reuse_max**.

- If the values of **password_reuse_time** and **password_reuse_max** are both positive numbers, an old password can be reused when it meets either of the reuse conditions
- If the value of **password_reuse_time** is **0**, password reuse is restricted based on the number of reuse times, and not on the reuse interval.
- If the value of **password_reuse_max** is **0**, password reuse is restricted based on the reuse interval, and not on the number of reuse times.
- If the values of both **password_reuse_time** and **password_reuse_max** are **0**, password reuse is not restricted.

Parameter type: integer.

Unit: none

Value range: 0 to 1000

- **0** indicates that the password reuse times are not checked.
- A positive number indicates that the new password cannot be the one whose reuse times exceed the specified number.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service security requirements. A larger value indicates higher security but more inconvenient use.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

password_lock_time

Parameter description: Specifies the duration before a locked account is automatically unlocked.

NOTE

The account locking and unlocking functions take effect only when the values of **password_lock_time** and **failed_login_attempts** are positive numbers.

Parameter type: floating-point number

Unit: day

Value range: 0 to 365. The integer part of the value indicates the number of days, and the decimal part can be converted into hours, minutes, and seconds. For example, **password_lock_time=1.5** indicates one day and 12 hours.

- **0** indicates that an account is not automatically locked if the password verification fails.
- A positive number indicates the duration after which a locked account is automatically unlocked.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The unit must be **d** if required.

Setting suggestion: Retain the default value. The automatic unlock time should not be too long. This parameter and **failed_login_attempts** should be set within a proper range.

Risks and impacts of improper settings: If the value of **failed_login_attempts** is too small and the automatic unlock time is too long, the account cannot be used for a long time due to incorrect input, affecting user experience.

failed_login_attempts

Parameter description: If the number of incorrect password attempts reaches the value of **failed_login_attempts**, the current account is locked. The account is automatically unlocked after the time specified by **password_lock_time**. During this period, only the SYSADMIN user can access the account. If you fail to enter the password during login or fail to change the password when running the **ALTER USER** command, the system considers that the entered password is incorrect.

NOTE

The account locking and unlocking functions take effect only when the values of **failed_login_attempts** and **password_lock_time** are positive numbers.

Parameter type: integer.

Unit: none

Value range: 0 to 1000.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **0** indicates that the automatic locking function does not take effect.
- A positive number indicates that an account is locked when the number of incorrect password attempts reaches the specified number.

Default value: **10**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set it to a value ranging from 5 to 10.

Risks and impacts of improper settings: If this parameter is set to a large value that allows too many incorrect password attempts (less than the maximum value of 1000), but no security measure is taken, security risk occurs. If this parameter is set to a small value (greater than 0 and less than 10), the account may be locked due to failed password attempts, affecting normal use.

enable_lock_account

Parameter description: If **failed_login_attempts** is set to **0**, the automatic lock function is disabled. In this setting, the system cannot record the number of login failures of an account. If **enable_lock_account** is set to disable the automatic lock function, the account will not be automatically locked after the number of consecutive login failures using incorrect passwords reaches the value of **failed_login_attempts**. In this case, the system reports an alarm to ensure account security. Manual locking by an administrator is not controlled by the parameter.

Parameter type: Boolean

Unit: none

Value range:

- **on**: An account is automatically locked when the number of consecutive login failures using incorrect passwords reaches the value of **failed_login_attempts**.
- **off**: An account is not automatically locked when the number of consecutive login failures using incorrect passwords reaches the value of **failed_login_attempts**.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: For security purposes, do not disable the automatic account locking function if the alarm function is not enabled.

Risks and impacts of improper settings: If it is set to **off**, account security deteriorates.

password_encryption_type

Parameter description: Specifies the encryption type of a user password. Changing the value of this parameter does not automatically trigger the change of the password encryption type of an existing user. Only the password of a new

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

user or the password changed by an existing user is encrypted using the new encryption type.

Parameter type: integer.

Unit: none

Value range: **0**, **1**, **2**, or **3**

- **0** indicates that passwords are encrypted with MD5.
- **1** indicates that passwords are encrypted with SHA-256 and MD5.
- **2** indicates that passwords are encrypted with SHA-256.
- **3** indicates that the passwords are encrypted with SM3.

Default value: 2

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impact of improper settings: The MD5 algorithm is not recommended and has low security. If this parameter is set to **0** or **1**, security risks exist.

NOTE

The FIPS mode does not support the MD5 and SM3 algorithms. If **password_encryption_type** is set to **0**, **1**, or **3**, the MD5 and SM3 algorithms do not take effect and the SHA-256 algorithm is forcibly used to encrypt user passwords.

password_min_length

Parameter description: Specifies the minimum length of an account password. Only the SYSADMIN user can access this parameter.

Parameter type: integer.

Unit: none

Value range: 6 to 999

Default value: **8**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter according to the actual situations. You are advised to use the default value.

Risks and impact of improper settings: If this parameter is set to a value greater than **password_max_length**, accounts cannot be created.

password_max_length

Parameter description: Specifies the maximum length of an account password. Only the SYSADMIN user can access this parameter.

Parameter type: integer.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 6 to 999

Default value: **32**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter according to the actual situations. You are advised to use the default value.

Risks and impact of improper settings: If this parameter is set to a value smaller than **password_min_length**, accounts cannot be created.

password_min_uppercase

Parameter description: Specifies the minimum number of uppercase letters that an account password must contain. Only the SYSADMIN user can access this parameter.

Parameter type: integer.

Unit: none

Value range: 0 to 999.

- **0**: no limit.
- 1 to 999: minimum number of uppercase letters required in a password when you create an account.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter according to the actual situations. You are advised to use the default value.

Risks and impact of improper settings: If this parameter is set to a value greater than **password_max_length**, accounts cannot be created.

password_min_lowercase

Parameter description: Specifies the minimum number of lowercase letters that an account password must contain. Only the SYSADMIN user can access this parameter.

Parameter type: integer.

Unit: none

Value range: 0 to 999.

- **0**: no limit.
- 1 to 999: minimum number of lowercase letters required in a password when you create an account.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Set this parameter according to the actual situations. You are advised to use the default value.

Risks and impact of improper settings: If this parameter is set to a value greater than **password_max_length**, accounts cannot be created.

password_min_digital

Parameter description: Specifies the minimum number of digits that an account password must contain. Only the SYSADMIN user can access this parameter.

Parameter type: integer.

Unit: none

Value range: 0 to 999.

- **0**: no limit.
- 1 to 999: minimum number of digits required in a password when you create an account.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter according to the actual situations. You are advised to use the default value.

Risks and impact of improper settings: If this parameter is set to a value greater than **password_max_length**, accounts cannot be created.

password_min_special

Parameter description: Specifies the minimum number of special characters that must be contained in an account password. Only the SYSADMIN user can access this parameter.

Parameter type: integer.

Unit: none

Value range: 0 to 999.

- **0**: no limit.
- 1 to 999: minimum number of special characters required in a password when you create an account.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter according to the actual situations. You are advised to use the default value.

Risks and impact of improper settings: If this parameter is set to a value greater than **password_max_length**, accounts cannot be created.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

password_effect_time

Parameter description: Specifies the validity period of an account password.

Parameter type: floating-point number

Unit: day

Value range: 0 to 999. The integer part indicates the number of days, and the decimal part can be converted into hours, minutes, and seconds. For example, **password_effect_time=0.5** indicates 0 days and 12 hours.

- **0**: The validity period restriction of an account password is disabled.
- 1 to 999: number of days for which an account password is valid. When the password is about to expire or has expired, the system prompts the user to change the password.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised to set the password validity period based on service security requirements. A smaller value indicates higher security.

Risks and impact of improper settings: If this parameter is set to **0**, security risks exist if the password is not changed for a long time.

password_notify_time

Parameter description: Specifies how many days in advance a user is notified before a password expires.

Parameter type: integer.

Unit: day

Value range: 0 to 999.

- **0**: The reminder is disabled.
- 1 to 999: number of days prior to password expiration that a user will receive a reminder.

Default value: **7**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impact of improper settings: If this parameter is set to **0**, password expiration is not reminded in advance.

enable_innertool_cert

Parameter description: Specifies whether internal tools use certificate authentication.

Parameter type: Boolean

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range:

- **on**: Internal tools use certificate-based authentication.
- **off**: Internal tools do not use certificate-based authentication.

Default value: **off**

NOTE

- This parameter takes effect only when both this parameter and **ssl** are enabled.
- By default, the certificate in the *$GAUSSHOME*/**share/sslcert/gsql** directory is used. **Common Name** of the certificate must be the same as the initial user. Otherwise, the authentication fails.
- After this parameter is enabled, the initial user uses certificate-based authentication, and other users use password-based authentication.
- After this parameter is enabled, the initial user can remotely connect to the database using certificate-based authentication.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: The default value is recommended. If you have high security requirements, you are advised to enable this parameter and correctly configure related certificates.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ldap_ca_file

Parameter description: Specifies the LDAP client certificate path.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **"/etc/openldap/certs/cacert.pem"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the actual certificate location.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.3.3 Communications Library Parameters

This section describes parameter settings and value ranges for communications libraries.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

tcp_keepalives_idle

Parameter description: Specifies the interval for transmitting keepalive signals on an OS that supports the **TCP_KEEPIDLE** socket option. If no keepalive signal is transmitted, the connection is in idle mode.

NOTE

- If the OS does not support **TCP_KEEPIDLE**, set this parameter to **0**.
- The parameter is ignored on an OS where connections are established using the UDS.
- If this parameter is set to **0**, the system value is used.
- This parameter is not shared among different sessions. That is, different session connections may have different values.
- The parameter value in the current session connection, not the value of the GUC copy, is displayed.

Parameter type: integer

Unit: seconds

Value range: 0 to 3600

Default value: **60**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **60** without a unit, **tcp_keepalives_idle** indicates 60s. If the value is **1min**, **tcp_keepalives_idle** indicates 1 minute. If the unit is required, the value must be **s**, **min**, **h**, or **d**.

Setting suggestion: Before setting this parameter, ensure that the OS supports the **TCP_KEEPIDLE** socket option.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

tcp_keepalives_interval

Parameter description: Specifies the response time before retransmission on an OS that supports the **TCP_KEEPINTVL** socket option.

NOTE

- If the OS does not support **TCP_KEEPINTVL**, set this parameter to **0**.
- The parameter is ignored on an OS where connections are established using the UDS.
- If this parameter is set to **0**, the system value is used.
- This parameter is not shared among different sessions. That is, different session connections may have different values.
- The parameter value in the current session connection, not the value of the GUC copy, is displayed.

Parameter type: integer

Unit: seconds

Value range: 0 to 180

Default value: **30**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **60** without a unit, **tcp_keepalives_interval** indicates 60s. If the value is **1min**, **tcp_keepalives_interval** indicates 1 minute. If the unit is required, the value must be **s**, **min**, **h**, or **d**.

Setting suggestion: Before setting this parameter, ensure that the OS supports the **TCP_KEEPINTVL** socket option.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

tcp_keepalives_count

Parameter description: Specifies the number of keepalive signals that can be waited before the GaussDB server is disconnected from the client on an OS that supports the **TCP_KEEPCNT** socket option.

NOTE

- If the OS does not support **TCP_KEEPCNT**, set this parameter to **0**.
- The parameter is ignored on an OS where connections are established using the UDS.
- If this parameter is set to **0**, the system value is used.
- This parameter is not shared among different sessions. That is, different session connections may have different values.
- The parameter value in the current session connection, not the value of the GUC copy, is displayed.

Parameter type: integer

Unit: none

Value range: 0 to 100. **0** indicates that the connection is immediately broken if GaussDB does not receive a keepalive signal from the client.

Default value: **20**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Before setting this parameter, ensure that the OS supports the **TCP_KEEPCNT** socket option.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

tcp_user_timeout

Parameter description: Specifies the maximum duration for which the transmitted data can remain in the unacknowledged state before the TCP connection is forcibly closed when the GaussDB sends data on the OS that supports the **TCP_USER_TIMEOUT** socket option.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If the OS does not support the **TCP_USER_TIMEOUT** option, the value of this parameter does not take effect. The default value is **0**.
- The parameter is ignored on an OS where connections are established using the UDS.

Parameter type: integer

Unit: milliseconds

Value range: 0 to 3600000

Default value: **0**, indicating that the value is set based on the OS.

Note that the effective result of this parameter varies according to the OS kernel.

- For AArch64 EulerOS (Linux kernel version: 4.19), the timeout interval is the value of this parameter.
- For x86 Euler 2.5 (Linux kernel version: 3.10), the timeout interval is not the value of this parameter but the maximum value in different ranges. That is, the timeout interval is the maximum upper limit of the total Linux TCP retransmission duration to which the value of **tcp_user_timeout** belongs. For example, if **tcp_user_timeout** is set to **40000**, the total retransmission duration is 51 seconds.

  Table 6-5 Value of tcp_user_timeout for x86 Euler 2.5 (Linux kernel version: 3.10)

Number of Linux TCP Retransmission Times	Total Linux TCP Retransmission Duration Range (s)	Example of tcp_user_timeout (ms)	Actual Linux TCP Retransmission Duration (s)
1	(0.2,0.6]	400	0.6
2	(0.6,1.4]	1000	1.4
3	(1.4,3]	2000	3
4	(3,6.2]	4000	6.2
5	(6.2,12.6]	10000	12.6
6	(12.6,25.4]	20000	25.4
7	(25.4,51]	40000	51
8	(51,102.2]	80000	102.2
9	(102.2,204.6]	150000	204.6
10	(204.6,324.6]	260000	324.6
11	(324.6,444.6]	400000	444.6

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Note: The duration of each TCP retransmission increases exponentially with the number of retransmission times. When the duration of a TCP retransmission reaches 120 seconds, the duration of each subsequent retransmission does not change.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Before setting this parameter, ensure that the OS supports the **TCP_USER_TIMEOUT** socket option.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

tcp_syn_retries

Parameter description: Specifies the number of SYN packet retransmissions due to SYN packet transmission failures in the three-way handshake for connecting to other internal nodes on the GaussDB that supports the **TCP_SYNCNT** socket option.

Parameter type: integer

Unit: none

Value range: 0–127. The value **0** indicates OS-based.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a small value (greater than **0**) and the network fluctuates, the number of SYN retransmissions is small. If the retransmission duration is shorter than the network recovery duration, the connection setup fails. If this parameter is set to a large value and the network is disconnected, the number of SYN retransmissions is large and the retransmission takes a long time. As a result, the overall error reporting time is prolonged.

NOTICE

- If the OS does not support the **TCP_SYNCNT** option, setting this parameter to any value does not take effect but the value is **0** by default.
- The parameter is ignored on an OS where connections are established using the UDS.

comm_tcp_mode

Parameter description: Specifies whether the communications library uses the TCP protocol to set up a data channel. The parameter setting takes effect after you restart the cluster.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> SCTP is no longer supported. This parameter is provided for compatibility, but its value is fixed at **on**.

Parameter type: Boolean

Unit: none

Value range: If this parameter is set to **on** for a CN, it connects to a DN using TCP. If this parameter is set to **on** for a DN, the DN communicates with each other using TCP.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

comm_sctp_port

Parameter description: Specifies the TCP port used to listen on data packet channels by the TCP proxy communications library.

Parameter type: integer

Unit: none

Value range: 0–65535

Default value: **25110** (The actual value is the value of the GUC parameter **port** plus 2.)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised not to set this parameter. Related configurations are automatically delivered during cluster installation.

Risks and impacts of improper settings: This port number is automatically allocated during cluster deployment. Do not change the parameter. If the port number is incorrectly configured, the database communication fails.

comm_control_port

Parameter description: Specifies the TCP listening port used by the TCP proxy communications library.

Parameter type: integer

Unit: none

Value range: 0–65535

Default value: **25111** (The actual value is the value of the GUC parameter **port** plus 3.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised not to set this parameter. Related configurations are automatically delivered during cluster installation.

Risks and impacts of improper settings: This port number is automatically allocated during cluster deployment. Do not change the parameter. If the port number is incorrectly configured, the database communication fails.

comm_max_datanode

Parameter description: Specifies the maximum number of DNs supported by the TCP proxy communications library.

Parameter type: integer

Unit: none

Value range: 1 to 8192

Default value: maximum number of primary DNs supported by each node.

Setting method: This parameter is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. During scale-in, this parameter takes effect only after the cluster is restarted.

Setting suggestion: The recommended value is **256**.

Risks and impacts of improper settings: If the parameter value is less than the number of DNs in the current cluster, the connection fails to be established. If the value is too large, too much memory is occupied.

comm_max_stream

Parameter description: Specifies the maximum number of concurrent data streams supported by the TCP proxy communications library. The value of this parameter must be greater than: Number of concurrent data streams x Number of operators in each stream x Square of **query_dop**.

Parameter type: integer

Unit: none

Value range: 2 to 60000

Default value: **1024**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- You are advised not to set this parameter to a large value because this will cause high memory usage, which equals to 380 bytes x **comm_max_stream** x (**comm_max_datanode** + **max_coordinators**). If the number of concurrent data streams is large, the query is complex and the SMP is large, resulting in insufficient memory.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If the process memory is sufficient, you can properly increase the value of **comm_max_stream**.

Risks and impacts of improper settings: If the value is too small, large-concurrency distributed stream queries will be restricted, and the error message "Cannot get stream index, maybe comm_max_stream is not enough" will be displayed. If the value is too large, too much memory will be occupied, and more system resources will be consumed when services are running in parallel. Before changing the values, check the specifications of the parameters and ensure that the hardware resources (such as CPU and memory) are sufficient. Otherwise, the cluster may be abnormal. Before service rollout, you need to complete distributed service reconstruction and perform pressure tests based on the actual service scenario to avoid errors caused by improper configuration.

comm_max_receiver

Parameter description: Specifies the maximum number of receiver threads for the TCP proxy communications library.

Parameter type: integer

Unit: none

Value range: 1 to 50

Default value: **4**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value to avoid thread switchover scheduling caused by excessive thread resources.

Risks and impacts of improper settings: If the value is too large, too many data receiving threads are started, wasting thread resources.

comm_quota_size

Parameter description: Specifies the maximum size of packets that can be consecutively sent by the TCP proxy communications library.

Parameter type: integer

Unit: KB

Value range: 0 to 2048000

Default value: **1024**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1024** without a unit, **comm_quota_size** indicates 1024 KB. If the value is **1MB**, **comm_quota_size** indicates 1 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Retain the default value. When you use a 1GE NIC, a small value ranging from 20 KB to 40 KB is recommended.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this parameter is set to a small value, the TCP communications library may wait for the quota during the sending and receiving phases.

comm_usable_memory

Parameter description: Specifies the maximum memory available for buffering on the TCP proxy communications library on a DN.

Parameter type: integer

Unit: KB

Value range: 102400 to 1073741823

Default value: **4096000** (that is, 4000 MB)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **102400** without a unit, **comm_usable_memory** indicates 102400 KB. If the value is **100MB**, **comm_usable_memory** indicates 100 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Retain the default value. Set this parameter based on the environment memory and deployment mode.

Risks and impacts of improper settings: If the value is too large, OOM occurs. If the value is too small, the performance of the TCP proxy communications library deteriorates.

comm_memory_pool

Parameter description: Specifies the size of the memory pool resources that can be used by the TCP proxy communications library on a DN.

Parameter type: integer

Unit: KB

Value range: 102400 to 1073741823

Default value: **2048000** (that is, 2000 MB)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **102400** without a unit, **comm_memory_pool** indicates 102400 KB. If the value is **100MB**, **comm_memory_pool** indicates 100 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Retain the default value. If the memory used by the communications library is small, set this parameter to a small value. Otherwise, set it to a large value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing. Before changing the values, check the specifications of the parameters and ensure that the hardware resources are sufficient. Otherwise, the cluster may be abnormal.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

comm_memory_pool_percent

Parameter description: Specifies the percentage of the memory pool resources that can be used by the TCP proxy communications library on a DN. This parameter is used to adaptively reserve memory used by the communications libraries.

Parameter type: integer

Unit: none

Value range: 0 to 100

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If the memory used by the communications library is small, set this parameter to a small value. Otherwise, set it to a large value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

comm_client_bind

Parameter description: Specifies whether to bind the client of the communications library to a specified IP address when the client initiates a connection.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The client is bound to a specified IP address.
- **off**: The client is not bound to a specified IP address.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If multiple IP addresses of a node in the cluster are on the same network segment, set this parameter to **on**. In this case, the client is bound to the IP address specified by **listen_addresses**. The concurrency performance of the cluster depends on the number of random ports because a port can be used by only one client at a time.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

comm_no_delay

Parameter description: Specifies whether to use the **NO_DELAY** attribute of a communications library connection. The modification of this parameter takes effect only after the cluster is restarted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The NO_DELAY attribute is used.
- **off**: The NO_DELAY attribute is not used.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Enable this parameter for delay-sensitive services and disable it for high throughput requirements. In addition, if packet loss occurs in the cluster because a large number of packets are received per second, set this parameter to **off** so that small packets are combined into large packets for transmission to reduce the total number of packets.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

comm_debug_mode

Parameter description: Specifies whether to enable the debug mode of the TCP proxy communications library, that is, whether to print logs about the communication layer.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The debug logs of the communications library are printed.
- **off**: The debug logs of the communications library are not printed.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If this parameter is set to **on**, a large number of logs will be printed, adding extra overhead and reducing database performance. Therefore, set it to **on** only in debugging scenarios.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

comm_ackchk_time

Parameter description: Specifies the duration after which the communications library server automatically triggers ACK when no data packet is received.

Parameter type: integer

Unit: milliseconds

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 0 to 20000. **0**: disabled.

Default value: **2000**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **60** without a unit, **comm_ackchk_time** indicates 60 ms. If the value is **1min**, **comm_ackchk_time** indicates 1 minute. The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

comm_timer_mode

Parameter description: Specifies whether to enable the timer mode of the TCP proxy communications library, that is, whether to print timer logs in each phase of the communication layer.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The timer logs of the communications library are printed.
- **off**: The timer logs of the communications library are not printed.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If this parameter is set to **on**, a large number of logs will be printed, adding extra overhead and reducing database performance. Therefore, set it to **on** only in debugging scenarios.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

comm_stat_mode

Parameter description: Specifies whether to enable the statistics mode of the TCP proxy communications library, that is, whether to print statistics about the communication layer.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The statistics logs of the communications library are printed.
- **off**: The statistics logs of the communications library are not printed.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: If this parameter is set to **on**, a large number of logs will be printed, adding extra overhead and reducing database performance. Therefore, set it to **on** only in debugging scenarios.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_stateless_pooler_reuse

Parameter description: Specifies whether to enable the reuse of the pooler connection pool. After the parameter is enabled, existing idle TCP connections can be reused. The setting takes effect after the cluster is restarted.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The pooler reuse mode is enabled.
- **off**: The pooler reuse mode is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Synchronize the settings for CNs and DNs. If this parameter is set to **off** for CNs and **on** for DNs, the cluster communication fails. Set this parameter to the same value for CNs and DNs. Restart the cluster for the setting to take effect.

Risks and impacts of improper settings: If this parameter is disabled, the reuse granularity of the pooler connection pool decreases and the memory usage of DNs may increase.

comm_cn_dn_logic_conn

Parameter description: Specifies whether logical connections are used between CNs and DNs. The setting takes effect after the cluster is restarted.

NOTE

Logical connections between CNs and DNs are no longer supported. This parameter is provided for compatibility, but its value is fixed at **off**.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The connections between CNs and DNs are logical, with the libcomm component in use.
- **off**: The connections between CNs and DNs are physical, with the libpq component in use.

Default value: **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is enabled, logical connections are used between CNs and DNs to eliminate the upper limit of sockets. Logical connections cause performance overhead of the communications library.

6.3.4 Resource Consumption

6.3.4.1 Memory

This section describes memory parameters.

NOTE

These parameters, except **local_syscache_threshold**, take effect only after the database restarts.

memorypool_enable

Parameter description: Specifies whether to enable a memory pool.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The memory pool is enabled.
- **off**: The memory pool is disabled.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

memorypool_size

Parameter description: Specifies the memory pool size.

Parameter type: integer.

Unit: KB

Value range: 131072–1073741823

Default value: **524288** (512 MB)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **131072** without a unit,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

memorypool_size indicates 131072 KB. If the value is **128MB**, **memorypool_size** indicates 128 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_memory_limit

Parameter description: Specifies whether to enable the logical memory management module.

NOTE

Fixed cost exists, that is, **shared_buffers** and metadata (about 200 MB). If **max_process_memory** minus the fixed cost is less than 2 GB, GaussDB forcibly sets **enable_memory_limit** to **off**. Metadata is the memory used in GaussDB and is related to some concurrent parameters, such as **max_connections**, **thread_pool_attr**, and **max_prepared_transactions**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The logical memory management module is enabled.
- **off**: The logical memory management module is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **off**, the memory used by the database is not limited. When a large number of concurrent or complex queries are performed, too much memory is used, which may cause OS OOM problems.

enable_threads_memory_reserve

Parameter description: Specifies whether to enable the background thread memory isolation module. After it is enabled, memory resources are reserved for background threads based on the background thread priority. When the logical memory management module is enabled (**enable_memory_limit** is set to **on**), it takes effect only when this parameter is set to **on**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enables the background thread memory isolation module.
- **off**: disables the background thread memory isolation module.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. After this function is enabled, the reserved memory and memory usage upper limit of background threads with different priorities are as follows.

Memory	High Priority	Medium Priority	Low Priority
Reserved memory 1 (maximum available dynamic memory < 10 GB)	Maximum available dynamic memory/10	Maximum available dynamic memory/10	Maximum available dynamic memory/20
Reserved memory 1 (maximum available dynamic memory < 20 GB)	0.75GB	1.5GB	0.75GB
Reserved memory 1 (maximum available dynamic memory < 40 GB)	0.9GB	3.0GB	1.5GB
Reserved memory 1 (maximum available dynamic memory ≥ 40 GB)	1.0GB	4.0GB	2.0GB
Memory usage upper limit	None	Maximum available dynamic memory x 30%	Equal to the reserved memory

Risks and impacts of improper settings: If this parameter is set to **off**, the memory usage of background threads is continuously collected, but the memory used by background threads is not isolated. In the case of high concurrency or complex queries, too much memory is used. As a result, background threads may fail to apply for memory.

max_process_memory

Parameter description: Specifies the maximum physical memory of a database node.

Parameter type: integer.

Unit: KB

Value range: 2097152 to 2147483647

Default value: For details, see "Typical scenario settings."

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **2097152** without a unit,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

max_process_memory indicates 2097152 KB. If the value is **2GB**, **max_process_memory** indicates 2 GB. The unit must be KB, MB, or GB if required.

Setting suggestion: This parameter is used to prevent node OOM caused by memory bloat, ensuring system reliability. On DNs, the value of this parameter depends on the physical memory of the system and the number of primary DNs deployed on a single server. The recommended formula is as follows: (Physical memory – **vm.min_free_kbytes**) x 0.7/(n + Number of primary DNs). **vm.min_free_kbytes** in this formula indicates that the OS memory reserved for the kernel to receive and send data. Its value is at least 5% of the total memory. That is, **max_process_memory** = Physical memory size x 0.665/(n + Number of primary DNs). When the number of nodes in the cluster is less than or equal to 256, n = 1. When the number of nodes in the cluster is greater than 256 and less than 512, n = 2. When the number of nodes in the cluster is greater than 512, n = 3. In independent deployment, n = 0. You can set this parameter on CNs to the same value as that on DNs.

Risks and impacts of improper settings: If this parameter is set to an improper value, for example, greater than the physical memory of the server, OS OOM problems may occur.

Typical scenario settings:

The following typical scenario settings are the optimal output values provided after the recommended specifications are tested and optimized. For details, see **Table 6-6**, **Table 6-7**, **Table 6-8**, and **Table 6-9**.

- Distributed independent deployment (single node, 1 CN or 1 DN)

  Table 6-6 Distributed independent deployment settings

Resource Setting	Recommended Value on CN	Recommended Value on DN
128C/1024G	900GB	900GB
60C/480G	360GB	360GB
32C/256G	192GB	192GB
16C/128G	96GB	96GB
8C/64G	40GB	40GB
4C/32G	20GB	20GB
4C/16G	10GB	10GB

- Finance edition (standard) (single node, 1 CN + 2 DNs)

  Table 6-7 Finance edition (standard) settings

Resource Setting	Recommended Value on CN	Recommended Value on DN
192C/1536G	300GB	550GB

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Resource Setting	Recommended Value on CN	Recommended Value on DN
128C/1024G	200GB	350GB
104C/1024G	200GB	350GB
96C/1024G	200GB	350GB
96C/768G	160GB	260GB
80C/640G	130GB	220GB
72C/576G	120GB	200GB
64C/512G	100GB	180GB
60C/480G	100GB	160GB
32C/256G	50GB	80GB
16C/128G	20GB	40GB
8C/64G	10GB	20GB

- Enterprise edition (single node, 1 CN + 3 DNs)

  Table 6-8 Enterprise edition settings

Resource Setting	Recommended Value on CN	Recommended Value on DN
192C/1536G	200GB	400GB
128C/1024G	150GB	250GB
104C/1024G	150GB	250GB
96C/1024G	150GB	250GB
96C/768G	110GB	190GB
80C/640G	90GB	160GB
80C/512G	75GB	125GB
72C/576G	80GB	140GB
64C/512G	75GB	125GB
60C/480G	70GB	120GB
32C/256G	35GB	60GB
16C/128G	15GB	25GB
8C/64G	9GB	15GB

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Finance edition (data computing) (single node, 1 CN + 4 DNs)

  Table 6-9 Finance edition (data computing) settings

Resource Setting	Recommended Value on CN	Recommended Value on DN
192C/1536G	160GB	300GB
128C/1024G	120GB	200GB
96C/768G	100GB	150GB
72C/576G	60GB	110GB
64C/512G	60GB	100GB
32C/256G	20GB	40GB

local_syscache_threshold

Parameter description: Specifies the maximum memory occupied by the system cache of a single session. When the actual system cache size exceeds the threshold, memory eviction is triggered. If **enable_global_syscache** and **enable_thread_pool** are enabled, this parameter indicates the total cache size of the current thread and sessions bound to the current thread.

Parameter type: integer.

Unit: KB

Value range: 1024 to 524288

Default value:

- Independent deployment: **16MB**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1024** without a unit, **local_syscache_threshold** indicates 1024 KB. If the value is **1MB**, **local_syscache_threshold** indicates 1 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: You are advised to set this parameter to an integer multiple of 1024. The background noise memory is 2 MB. Each time a table or index is added, the memory increases by 11 KB.

Risks and impacts of improper settings: If the value is too small, memory is frequently evicted, affecting performance.

enable_memory_context_control

Parameter description: Enables the function of checking whether the number of memory contexts exceeds the specified limit. This parameter applies only to the DEBUG version.

Parameter type: Boolean.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range:

- **on**: The function of checking the number of memory contexts is enabled.
- **off**: The function of checking the number of memory contexts is disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

uncontrolled_memory_context

Parameter description: Specifies which memory context will not be checked when the function of checking whether the number of memory contexts exceeds the specified limit is enabled. This parameter applies only to the DEBUG version.

NOTE

When the parameter is queried, the title meaning string "MmgrMemoryController white list:" is added to the beginning of the parameter value.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

huge_pages

Parameter description: Specifies whether to enable huge pages of static memory for the shared memory of the database.

NOTE

To enable the static huge page memory function, you need to set sufficient static huge page memory resources in the OS kernel in advance. The related parameters are **vm.nr_hugepages** and **vm.nr_overcommit_hugepages**, which are set in the **/etc/sysctl.conf** file.

Parameter type: string.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: enables the function of applying for static huge page memory. If huge page memory resources cannot be applied for, the database fails to be started.
- **off**: disables the function of applying for static huge page memory and common shared memory is applied for.
- **try**: attempts to apply for static huge page resources. If the application fails, the system applies for common shared memory.

Default value: **try**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

shared_buffers

Parameter description: Specifies the size of shared memory used by GaussDB. Increasing the value of this parameter causes GaussDB to request more System V shared memory than the default configuration allows.

Parameter type: integer.

Unit: page (8 KB)

Value range: 16–1073741823

Default value: For details, see "Typical scenario settings."

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1024** without a unit, **shared_buffers** indicates 8192 KB (1024 x 8 KB). If the value is **8MB**, **shared_buffers** indicates 8 MB. If the value contains a unit, the value must be KB, MB, or GB and must be an integer multiple of 8 KB.

Setting suggestion:

- Set this parameter on DNs to a value greater than that on CNs because most queries in GaussDB are pushed down.
- Set **shared_buffers** to a value less than 40% of **max_process_memory**.
- If **shared_buffers** is set to a larger value, increase the value of **checkpoint_segments** because a longer interval is required to write a large amount of new or changed data.
- If the process fails to be restarted after the value of **shared_buffers** is changed, perform either of the following operations based on the error information:
  - Adjust the **kernel.shmall**, **kernel.shmmax**, and **kernel.shmmin** OS parameters.
  - Run the **free -g** command to check whether the available memory and swap space of the OS are sufficient. If the memory is insufficient, manually stop other user programs that occupy much memory.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - For different specifications, use the default values in the corresponding suggestions. You are advised not to change the value. Otherwise, the value of **shared_buffers** may be too large or too small. Note that the following condition must be met: **data_replicate_buffer_size** + **segment_buffers** + **shared_buffers** + **wal_buffers** + **temp_buffers** + **maintenance_work_mem** + **work_mem** + **query_mem** + (Standby node) **wal_receiver_buffer_size** < **max_process_memory** < Memory size of the physical machine.

Risks and impacts of improper settings: If this parameter value is set higher than the physical memory limit, the database will fail to start as it cannot request sufficient memory.

Typical scenario settings:

The following typical scenario settings are the optimal output values provided after the recommended specifications are tested and optimized. For details, see **Table 6-10**, **Table 6-11**, **Table 6-12**, and **Table 6-13**.

- Distributed independent deployment (single node, 1 CN or 1 DN)

  Table 6-10 Distributed independent deployment settings

Resource Setting	Recommended Value on CN	Recommended Value on DN	CN Value Range	DN Value Range
128C/1024G	8GB	304GB	256MB ~ 409.6GB	2GB ~ 409.6GB
60C/480G	4GB	140GB	256MB ~ 192GB	2GB ~ 192GB
32C/256G	2GB	76GB	256MB ~ 102.4GB	2GB ~ 102.4GB
16C/128G	2GB	40GB	256MB ~ 51.2GB	2GB ~ 51.2GB
8C/64G	1GB	16GB	256MB ~ 25.6GB	2GB ~ 25.6GB
4C/32G	512MB	8GB	256MB ~ 12.8GB	2GB ~ 12.8GB
4C/16G	256MB	4GB	256MB ~ 6.4GB	2GB ~ 6.4GB

- Finance edition (standard) (single node, 1 CN + 2 DNs)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-11 Finance edition (standard) settings

Resource Setting	Recommended Value on CN	Recommended Value on DN
192C/1536G	2GB	220GB
128C/1024G	2GB	140GB
104C/1024G	2GB	140GB
96C/1024G	2GB	140GB
96C/768G	2GB	100GB
80C/640G	2GB	80GB
72C/576G	2GB	80GB
64C/512G	2GB	70GB
60C/480G	2GB	60GB
32C/256G	1GB	30GB
16C/128G	1GB	16GB
8C/64G	512MB	8GB

- Enterprise edition (single node, 1 CN + 3 DNs)

Table 6-12 Enterprise edition settings

Resource Setting	Recommended Value on CN	Recommended Value on DN
192C/1536G	2GB	160GB
128C/1024G	2GB	100GB
104C/1024G	2GB	100GB
96C/1024G	2GB	100GB
96C/768G	2GB	76GB
80C/640G	2GB	64GB
80C/512G	2GB	50GB
72C/576G	2GB	56GB
64C/512G	2GB	50GB
60C/480G	2GB	48GB
32C/256G	1GB	24GB
16C/128G	1GB	10GB

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Resource Setting	Recommended Value on CN	Recommended Value on DN
8C/64G	512MB	6GB

- Finance edition (data computing) (single node, 1 CN + 4 DNs)

  Table 6-13 Finance edition (data computing) settings

Resource Setting	Recommended Value on CN	Recommended Value on DN
192C/1536G	2GB	120GB
128C/1024G	2GB	80GB
96C/768G	2GB	50GB
72C/576G	1GB	40GB
64C/512G	1GB	30GB
32C/256G	512MB	10GB

enable_scale_cache

Parameter description: ScaleCache optimizes buffer management by improving the locality of buffer page pin and unpin, SIMD-accelerated hash retrieval, and lock-free index scan technology of copy on write, thereby improving the overall buffer performance. This parameter can be set only in the Arm 8 or later environment.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The ScaleCache buffer management mode is enabled.
- **off**: The ScaleCache buffer management mode is disabled.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

pin_buffer_group_cores

Parameter description: Specifies the number of CPU cores in each group when the ScaleCache function is used to group CPU cores during pin and unpin

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

optimization. This parameter can be set and modified only when **enable_scale_cache** is set to **on**. This parameter can be set only in the Arm 8 or later environment.

Parameter type: integer.

Unit: none

Value range: 1 to 1073741823

Default value: **4**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter to a power of 2. If the number of CPU cores or the value of this parameter is not a power of 2, this parameter is set to **1**.

Risks and impacts of improper settings: Before modifying this parameter, ensure that the **enable_scale_cache** parameter is enabled. Before modifying this parameter, make sure you understand its meaning and fully verify the change in a test environment.

enable_shared_storage_aio

Parameter description: Specifies whether to enable the function of optimizing asynchronous parallel write logs in OceanStor Dorado shared storage mode.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The asynchronous write function of the OceanStor Dorado shared storage is enabled.
- **off**: The asynchronous write function of the OceanStor Dorado shared memory is disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

page_version_check

Parameter description: Specifies whether to perform verification for underlying storage faults and pages not marked as dirty based on page version information. **page_version_check** is a level-3 switch. The verification for underlying storage faults is to check whether a page read from the underlying storage is of a correct version, which prevents loss of page version information caused by a fault such as a disk power failure. The verification for pages not marked as dirty is to check whether modified pages are not marked as dirty, and is controlled by an independent switch **page_missing_dirty_check**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: enumerated type

Unit: none

Value range:

- **off**: The verification for underlying storage faults and pages not marked as dirty is disabled.
- **memory**: The page version verification function (that is, verification for underlying storage faults and pages not marked as dirty) in pure memory mode is enabled. The page version information is cached only in the memory and will be lost after a restart.
- **persistence**: The persistent page version verification function (that is, verification for underlying storage faults and pages not marked as dirty) is enabled. The page version information is persisted to files and will not be lost after a restart.

Default value: **persistence** (192-core CPU/1536 GB memory, 128-core CPU/1024 GB memory, 104-core CPU/1024 GB memory, 96-core CPU/1024 GB memory, 96-core CPU/768 GB memory, 80-core CPU/640 GB memory, 80-core CPU/512 GB memory, 72-core CPU/576 GB memory, 64-core CPU/512 GB memory, 60-core CPU/480 GB memory, 32-core CPU/256 GB memory, 16-core CPU/128 GB memory); **off** (8-core CPU/64 GB memory, 4-core CPU/16 GB memory, 4-core CPU/32 GB memory)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the server specifications. Recommended settings: **persistence** (192-core CPU/1536 GB memory, 128-core CPU/1024 GB memory, 104-core CPU/1024 GB memory, 96-core CPU/1024 GB memory, 96-core CPU/768 GB memory, 80-core CPU/640 GB memory, 80-core CPU/512 GB memory, 72-core CPU/576 GB memory, 64-core CPU/512 GB memory, 60-core CPU/480 GB memory, 32-core CPU/256 GB memory, 16-core CPU/128 GB memory); **off** (8-core CPU/64 GB memory, 4-core CPU/16 GB memory, 4-core CPU/32 GB memory).

Risks and impacts of improper settings: Setting this parameter to **memory** and **persistence** affects the performance of a device, and the one with smaller specifications suffers more. (For example, the performance of a 16-core CPU/128 GB storage device using TPC-C model is degraded by about 7%.) If the system needs to restart frequently, you are advised to set this parameter to **persistence** to ensure that the version information on the page is not lost. However, the performance will be affected.

page_version_full_validation

Parameter description: Inserts the page version into AntiCache during dirty page flushing for the buffer in AntiCache persistent state. This is more frequent than the memory mode in which the page version is inserted only when the buffer is eliminated. As a result, the AntiCache memory may be fully occupied, causing missing verification. To reduce the probability of missing verification and achieve 100% verification coverage, the insertion mode in persistent state and the Vlog file recycling mechanism need to be optimized. The **page_version_full_validation** parameter specifies whether to enable the full verification mode of AntiCache.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

After this parameter is enabled, if the memory is full when the page is inserted into AntiCache, the system first performs in-place verification on an existing page to release space and then inserts a new page version.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The AntiCache full verification mode is enabled.
- **off**: The AntiCache full verification mode is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

page_missing_dirty_check

Parameter description: Checks whether the modified pages are not marked as dirty. **page_missing_dirty_check** is controlled by **page_version_check**. If **page_version_check** is set to **off**, setting **page_missing_dirty_check** to **on** does not take effect.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The verification for pages not marked as dirty is performed.
- **off**: The verification for pages not marked as dirty is not performed.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised to enable this function in test scenarios to detect as many pages not marked as dirty that lead to code bugs in non-production environments as possible. On the live network, this function is disabled by default to avoid extra cost and performance deterioration.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

page_version_max_num

Parameter description: Specifies the maximum number of page versions that can be cached in the memory. This parameter is valid only when **page_version_check** is not set to **OFF**. Each **page_version** occupies 36 bytes of memory. Pay attention to the memory usage.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

- **0**: The value is automatically calculated based on the value of **shared_buffers** using the following formula: **page_version_max_num** = **shared_buffers** x 2. For example, 32 MB of **shared_buffers** corresponds to 4096 buffers. Therefore, the value of this parameter is set to **8192**.
- Non-zero values: The manually configured value is forcibly used. The parameter cannot be set a value less than 16 times of **page_version_partitions**. Otherwise, it is forcibly set to a value equal to the value of **page_version_partitions** multiplied by 16.

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If the memory is limited, the valid value must be twice to four times the value of **shared_buffers**. **Setting suggestion**: If high performance is required and the memory is sufficient, you are advised to manually set this parameter to a value larger than four times the value of **shared_buffers** and the ratio range of this parameter to **page_version_partitions** is [256,1024].

Risks and impacts of improper settings: If this parameter value is set higher than the physical memory limit, the database will fail to start as it cannot request sufficient memory.

page_version_partitions

Parameter description: Specifies the number of hash table partitions in cached page version information in the memory. This parameter directly affects the hash query efficiency and hash conflict probability. This parameter is valid only when **page_version_check** is not set to **OFF**.

Parameter type: integer.

Unit: none

Value range: 0–2097152

- **0**: The value is automatically calculated based on the value of **page_version_max_num** using the following formula: **page_version_partitions** = **page_version_max_num**/512. If the automatically calculated value is smaller than 4, the parameter is forcibly set to **4**.
- Non-zero values: The manually configured value is forcibly used. The minimum value is **4**. If the value is less than 4, the parameter is forcibly set to **4**.

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: If you have high performance requirements, you are advised to manually set this parameter to a value 1/256 to 1/1024 of the value of **page_version_max_num**.

Risks and impacts of improper settings: If the number of partitions is too large, the hash query efficiency decreases. If the number of partitions is too small, the probability of hash conflicts increases.

page_version_recycler_thread_num

Parameter description: Specifies the number of threads for recycling and verifying page version information. This parameter is valid only when **page_version_check** is not set to **off**.

Parameter type: integer.

Unit: none

Value range: 0–16

- If **page_version_check** is set to **memory**:
  - **0**: The value is automatically calculated based on **page_version_partitions**. The formula is as follows: page_version_recycler_thread_num = page_version_partitions/16384. If the automatically calculated value is greater than 4, the parameter is forcibly set to **4**.
  - Non-zero values: The manually configured value is forcibly used.
  - The value cannot be greater than that of **page_version_partitions**. Otherwise, it is forcibly set to a value equal to that of **page_version_partitions**.
- If **page_version_check** is set to **persistence**:

  If the value is less than 2, set this parameter to **2**. If the value is greater than or equal to 2, the manually configured parameter value is forcibly used.

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_cached_context

Parameter description: Specifies the number of cached memory contexts. If this parameter is set to **0**, the memory context optimization framework does not take effect.

Parameter type: integer.

Unit: none

Value range: 0 to 1024

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If this parameter is set to **0**, the memory context optimization framework does not take effect.
- If this parameter is set to a non-zero value, the memory context optimization framework takes effect, and the value of this parameter is the number of cached memory contexts.

Default value: **128**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

verify_log_buffers

Parameter description: Specifies the size of the verifyLog buffer. This parameter is valid only when **page_version_check** is set to **persistence**. The verifyLog buffer memory is managed by page, and each page is 8 KB.

Parameter type: integer.

Unit: page (8 KB)

Value range: 4-262144

Default value: **1GB** (192-core CPU/1536 GB memory, 128-core CPU/1024 GB memory, 104-core CPU/1024 GB memory, 96-core CPU/1024 GB memory, 96-core CPU/768 GB memory, 80-core CPU/640 GB memory, 80-core CPU/512 GB memory, 72-core CPU/576 GB memory, 64-core CPU/512 GB memory, 60-core CPU/480 GB memory, 32-core CPU/256 GB memory); **512MB** (16-core CPU/128 GB memory); **256MB** (8-core CPU/64 GB memory); **128MB** (4-core CPU/32 GB memory); **16MB** (4-core CPU/16 GB memory)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, the value **131072** of **verify_log_buffers** indicates that the size is 131072 x 8 KB = 1GB, and the value **131072kB** indicates that **verify_log_buffers** is 131072 KB. If the value contains a unit, the value must be KB, MB, or GB and must be an integer multiple of 8 KB.

Setting suggestion: Set this parameter based on the system hardware specifications.

1GB (192-core CPU/1536 GB memory, 128-core CPU/1024 GB memory, 104-core CPU/1024 GB memory, 96-core CPU/1024 GB memory, 96-core CPU/768 GB memory, 80-core CPU/640 GB memory, 80-core CPU/512 GB memory, 72-core CPU/576 GB memory, 64-core CPU/512 GB memory, 60-core CPU/480 GB memory, 32-core CPU/256 GB memory); **512MB** (16-core CPU/128 GB memory); **256MB** (8-core CPU/64 GB memory); **128MB** (4-core CPU/32 GB memory); **16MB** (4-core CPU/16 GB memory)

Risks and impacts of improper settings: If this parameter value is set higher than the physical memory limit, the database will fail to start as it cannot request sufficient memory.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

segment_buffers

Parameter description: Specifies the memory size of a GaussDB segment-page metadata page.

Parameter type: integer.

Unit: page (8 KB)

Value range: 16–1073741823

Default value: **1024** (that is, 8 MB)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1024** without a unit, **segment_buffers** indicates 8192 KB (1024 x 8 KB). If the value is **8MB**, **segment_buffers** indicates 8 MB. If the value contains a unit, the value must be KB, MB, or GB and must be an integer multiple of 8 KB.

Setting suggestions: **segment_buffers** is used to cache the content of segment-page headers, which is key metadata information. To improve performance, it is recommended that the segment headers of ordinary tables be cached in the buffer and not be replaced. You are advised to set this parameter based on the following formula: Number of tables (including indexes and TOAST tables) x Number of partitions x 3 + 128. Multiplying by 3 is because each table (partition) has some extra metadata segments. Generally, a table has three segments. Adding 128 at last is because segment-page tablespace management requires a certain number of buffers. If this parameter is set to a small value, it takes a long time to create a segment-page table for the first time. Therefore, you are advised to retain the default value to avoid setting **segment_buffers** to an excessively large or small value. Note that the following condition must be met: **data_replicate_buffer_size** + **segment_buffers** + **shared_buffers** + **wal_buffers** + **temp_buffers** + **maintenance_work_mem** + **work_mem** + **query_mem** + (Standby node) **wal_receiver_buffer_size** < **max_process_memory** < Memory size of the physical machine.

Risks and impacts of improper settings: If this parameter value is set higher than the physical memory limit, the database will fail to start as it cannot request sufficient memory.

bulk_read_ring_size

Parameter description: Specifies the size of a ring buffer used for parallel data export.

Parameter type: integer.

Unit: KB

Value range: 256 to 2147483647

Default value: **16384** (that is, 16 MB)

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1024** without a unit, **bulk_read_ring_size** indicates 1024 KB. If the value is **1MB**, **bulk_read_ring_size** indicates 1 MB. The unit must be KB, MB, or GB if required.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

bulk_write_ring_size

Parameter description: Specifies the size of a ring buffer used for parallel data import.

Parameter type: integer.

Unit: KB

Value range: 16384 to 2147483647

Default value: **2097152** (that is, 2 GB)

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **16384** without a unit, **bulk_write_ring_size** indicates 16384 KB. If the value is **16MB**, **bulk_write_ring_size** indicates 16 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Increase the value of this parameter on DNs if a huge amount of data will be imported.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

standby_shared_buffers_fraction

Parameter description: Specifies the **shared_buffers** proportion used on the server where a standby instance is deployed.

Parameter type: floating-point number

Unit: none

Value range: 0.1 to 1.0

Default value: **1**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

temp_buffers

Parameter description: Specifies the maximum size of local temporary buffers used by a database session.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- **temp_buffers** can be modified only before the first use of temporary tables within each session. Subsequent attempts to change the value of this parameter will not take effect on that session.
- A session allocates temporary buffers based on the value of **temp_buffers**. If a large value is set in a session that does not require many temporary buffers, only the cost of one buffer descriptor is added. If a buffer is used, additional 8192 bytes will be consumed for it.

Parameter type: integer.

Unit: page (8 KB)

Value range: 100 to 1073741823

Default value: **128** (that is, 1 MB)

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **100** without a unit, **temp_buffers** indicates 800 KB (100 x 8 KB). If the value is **16MB**, **temp_buffers** indicates 16 MB. If the value contains a unit, the value must be KB, MB, or GB and must be an integer multiple of 8 KB.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_prepared_transactions

Parameter description: Sets the maximum number of transactions that can be in the "prepared" state simultaneously. Increasing the value of this parameter causes GaussDB to request more System V shared memory than the default configuration allows.

Parameter type: integer.

Unit: none

Value range: 0 to 262143

Default value:

- Independent deployment:

  2400 (128-core CPU/1024 GB memory); **1200** (60-core CPU/480 GB memory and 32-core CPU/256 GB memory); **800** (16-core CPU/128 GB memory); **400** (8-core CPU/64 GB memory); **300** (4-core CPU/32 GB memory); **200** (4-core CPU/16 GB memory)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestions: The default value is recommended. You need to adjust the value only when a two-phase transaction reports an error indicating insufficient slots. To avoid failures in the preparation step, the value of this parameter must be greater than the number of worker threads in **thread_pool_attr** in thread pool mode. In non-thread pool mode, the value of this parameter must be greater than or equal to the value of **max_connections**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: When GaussDB is deployed as an HA system, set this parameter on standby nodes to a value greater than or equal to that on primary nodes. Otherwise, queries will fail on the standby nodes.

work_mem

Parameter description: Specifies the amount of memory to be used by internal sort operations and hash tables before they write data into temporary disk files. Sort operations are required for **ORDER BY**, **DISTINCT**, and **MERGE JOINS**. Hash tables are used in hash joins, hash-based aggregation, and hash-based processing of IN subqueries.

In a complex query, several sort or hash operations may run concurrently; each operation will be allowed to use as much memory as this parameter specifies. If the memory is insufficient, data will be written into temporary files. In addition, several running sessions could be performing such operations concurrently. Therefore, the total memory used may be many times the value of **work_mem**.

Parameter type: integer.

Unit: KB

Value range: 64 to 2147483647

Default value:

- Independent deployment:

  256MB (128-core CPU/1024 GB memory); **128MB** (60-core CPU/480 GB memory, 32-core CPU/256 GB memory, and 16-core CPU/128 GB memory); **64MB** (8-core CPU/64 GB memory); 32MB (4-core CPU/32 GB memory); **16MB** (4-core CPU/16 GB memory)

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1024** without a unit, **work_mem** indicates 1024 KB. If the value is **1MB**, **work_mem** indicates 1 MB. The unit must be KB, MB, or GB if required.

Setting suggestion:

If the physical memory specified by **work_mem** is insufficient, additional operator calculation data will be written into temporary tables based on query characteristics and the concurrency. This reduces performance by five to ten times, and prolongs the query response time from seconds to minutes.

- For complex serial queries, each query requires five to ten associated operations. Set **work_mem** using the following formula: **work_mem** = 50% of the memory/10.
- For simple serial queries, each query requires two to five associated operations. Set **work_mem** using the following formula: **work_mem** = 50% of the memory/5.
- For concurrent queries, set **work_mem** using the following formula: **work_mem** = **work_mem** for serial queries/Number of concurrent SQL statements.
- BitmapScan hash tables are also restricted by **work_mem**, but will not be forcibly flushed to disks. In the case of complete lossify, each 1 MB of memory used by the hash table corresponds to a 16-GB page of BitmapHeapScan.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

After the upper limit of **work_mem** is reached, the memory increases linearly with the data access traffic based on this ratio.

Risks and impacts of improper settings: The default value is the recommended value. You are advised not to change it. Before changing the values, check the specifications of the parameters and ensure that the hardware resources are sufficient. Otherwise, the cluster may be abnormal.

query_mem

Parameter description: Specifies the memory used by a query.

Parameter type: integer.

Unit: KB

Value range: 0 to 2147483647

- If the value of **query_mem** is greater than 32768 (that is, 32 MB), the optimizer adjusts the memory cost estimate to this value when generating an execution plan.
- If the value is less than 32768 (that is, 32 MB), the default value **0** is forcibly used. In this case, the optimizer does not adjust the estimated query memory.

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **32768** without a unit, **query_mem** indicates 32768 KB. If the value is **32MB**, **query_mem** indicates 32 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

query_max_mem

Parameter description: Specifies the maximum memory that can be used by a query.

Parameter type: integer.

Unit: KB

Value range: 0 to 2147483647

- If the value of **query_max_mem** is greater than 32768 (that is, 32 MB), an error is reported when the query memory usage exceeds the value.
- If the value is less than 32768 (that is, 32 MB), the default value **0** is used. In this case, the optimizer does not limit the query memory.

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **32768** without a unit, **query_max_mem** indicates 32768 KB. If the value is **32MB**, **query_max_mem** indicates 32 MB. The unit must be KB, MB, or GB if required.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

maintenance_work_mem

Parameter description: Specifies the maximum amount of memory to be used by maintenance operations, such as **VACUUM** and **CREATE INDEX**. This parameter may affect the execution efficiency of **VACUUM**, **VACUUM FULL**, **CLUSTER**, and **CREATE INDEX**.

Parameter type: integer.

Unit: KB

Value range: 1024 to 2147483647

Default value:

- Independent deployment:

  CN: **2GB** (128-core CPU/1024 GB memory); **1GB** (60-core CPU/480 GB memory); **512MB** (32-core CPU/256 GB memory); **256MB** (16-core CPU/128 GB memory); **128MB** (8-core CPU/64 GB memory); **64MB** (4-core CPU/32 GB memory); **32MB** (4-core CPU/16 GB memory)

  DN: **4GB** (128-core CPU/1024 GB memory); **2GB** (60-core CPU/480 GB memory); **1GB** (32-core CPU/256 GB memory); **512MB** (16-core CPU/128 GB memory); **256MB** (8-core CPU/64 GB memory); **128MB** (4-core CPU/32 GB memory); **64MB** (4-core CPU/16 GB memory)

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1024** without a unit, **maintenance_work_mem** indicates 1024 KB. If the value is **1MB**, **maintenance_work_mem** indicates 1 MB. The unit must be KB, MB, or GB if required.

Setting suggestion:

- The value of this parameter must be greater than that of **work_mem** so that database dumps can be more quickly cleared or restored. In a database session, only one maintenance operation can be performed at a time. Maintenance is usually performed when there are not many running sessions.
- When the thread described in **Autovacuum** is running, up to ***autovacuum_max_workers*** times this memory may be allocated. In this case, set **maintenance_work_mem** to a value greater than or equal to that of **work_mem**.
- If a large amount of data is to be clustered, increase the value of this parameter in the session.
- Use the default value. You are advised not to change the value. Otherwise, set **maintenance_work_mem** to an excessively large or small value. Note that the following condition must be met: **data_replicate_buffer_size** + **segment_buffers** + **shared_buffers** + **wal_buffers** + **temp_buffers** + **maintenance_work_mem** + **work_mem** + **query_mem** + (Standby node) **wal_receiver_buffer_size** < **max_process_memory** < Memory size of the physical machine.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: The default value is the recommended value. You are advised not to change it. Before changing the values, check the specifications of the parameters and ensure that the hardware resources are sufficient. Otherwise, the cluster may be abnormal.

max_stack_depth

Parameter description: Specifies the maximum safe depth of the GaussDB execution stack. The safety margin is required because the stack depth is not checked in every routine in the server, but only in key potentially-recursive routines, such as expression evaluation.

Parameter type: integer.

Unit: KB

Value range: 100–2147483647

Default value:

- If the value of the system thread stack size minus 640 KB is greater than or equal to 2 MB, the default value of this parameter is **2MB**.
- If the value of the system thread stack size minus 640 KB is less than 2 MB, the default value of this parameter is the system thread stack size minus 640 KB.

NOTICE

- You can run the **ulimit -s** command to view the size of the system thread stack.
- After the size of the system thread stack is modified, you need to restart the cluster for the database process to use the new size of the system thread stack. The database process inherits this limit from its parent process at start time and does not re-read the setting when running.

Setting method: This is a SUSET parameter. Set it based on instructions provided in Table 6-1. For example, if the value is **1024** without a unit, **max_stack_depth** indicates 1024 KB. If the value is **1MB**, **max_stack_depth** indicates 1 MB. The unit must be KB, MB, or GB if required.

Setting suggestion:

- The database needs to reserve 640 KB stack depth. Therefore, the optimal value of this parameter is equal to the system thread stack size minus 640 KB.
- The value of this parameter cannot be greater than the system thread stack size minus 640 KB. If you use the gs_guc tool to set this parameter to a value greater than the system thread stack size minus 640 KB, the setting fails.
- Since not all operations can be detected, you are advised to set a specific value for this parameter.
- The default value is **2 MB**, which is relatively small and does not easily cause system breakdown.

Risks and impacts of improper settings:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If the system thread stack size minus 640 KB is less than the minimum value of this parameter, the database will fail to start.
- Setting this parameter to a value greater than the actual kernel limit means that a running recursive function may crash an individual backend process.

enable_early_free

Parameter description: Specifies whether the operator memory can be released in advance.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The operator memory can be released in advance.
- **off**: The operator memory cannot be released in advance.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

memory_trace_level

Parameter description: Specifies the control level for recording memory allocation information when the dynamic memory usage exceeds the specified percentage threshold (90% by default, which can be changed by setting the **memory_log_config** parameter) of the maximum dynamic memory. This parameter takes effect only when the GUC parameters **use_workload_manager** and **enable_memory_limit** are enabled.

Parameter type: enumerated type

Unit: none

Value range:

- **none**: indicates that memory allocation information is not recorded.
- **level1**: After the dynamic memory usage exceeds the specified percentage threshold of the maximum dynamic memory, the following memory information is recorded and saved in the *$GAUSSLOG*/**mem_log** directory:
  - Global memory overview.
  - Memory usage of the top 20 memory contexts of the instance, session, and thread types.
  - The **totalsize** and **freesize** columns for each memory context.
- **level2**: After the dynamic memory usage exceeds the specified percentage threshold of the maximum dynamic memory, the following memory information is recorded and saved in the *$GAUSSLOG*/**mem_log** directory:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - Global memory overview.
  - Memory usage of the top 20 memory contexts of the instance, session, and thread types.
  - The **totalsize** and **freesize** columns for each memory context.
  - Detailed information about all memory applications in each memory context, including the file where the allocated memory is located, line number, and size.

Default value: **level1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If this parameter is set to **level2**, the memory allocation details (file, line, and size) of each memory context are recorded, which greatly affects the performance. Therefore, exercise caution when setting this parameter.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

- The recorded memory snapshot information can be queried by using the system function gs_get_history_memory_detail(cstring). For details about the function, see **gs_get_history_memory_detail(cstring)**.
- If the **use_workload_manager** parameter is disabled and the **bypass_workload_manager** parameter is enabled, this parameter also takes effect. The **bypass_workload_manager** parameter is of the SIGHUP type; therefore, after the reload mode is set, you need to restart the database for the setting to take effect.
- The recorded memory context is obtained after all memory contexts of the same type with the same name are summarized.

memory_log_config

Parameter description: Specifies the information as follows: when the dynamic memory usage exceeds the threshold, the dynamic memory fluctuation exceeds the threshold, or the statistical period ends, the top *N* memory contexts of each type and the memory information of the top *N* SQL statements are recorded, and memory logs are generated. This parameter takes effect only when the GUC parameters **use_workload_manager** and **enable_memory_limit** are enabled and **memory_trace_level** is not **none**.

Parameter type: string.

Unit: none

Value range:

This parameter consists of the five parts **memory_log_threshold**, **memory_rush_percent**, **memory_log_period**, **topn_memory_contexts**, and **topn_queries**, whose meanings are as follows:

- **memory_log_threshold**: specifies the percentage threshold of the dynamic memory usage. The value is an integer ranging from 0 to 100. The value **0** indicates that the function is disabled. The unit is %.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **memory_rush_percent**: specifies the percentage threshold of dynamic memory fluctuation. The value is an integer ranging from 0 to 100. The value **0** indicates that the function is disabled. The unit is %.
- **memory_log_period**: specifies the statistical period. The value is an integer ranging from 0 to *INT_MAX* (excluding *INT_MAX*). *INT_MAX* is the maximum value **2147483647** of the INT type. The value **0** indicates that the function is disabled. The unit is minute.
- **topn_memory_contexts**: specifies the top-N memory contexts of each type to be recorded. The value is an integer ranging from 0 to *INT_MAX* (excluding *INT_MAX*). The value **0** indicates that the function is disabled.
- **topn_queries**: specifies the top-N SQL statements to be recorded based on memory usage. The value is an integer ranging from 0 to *INT_MAX* (excluding *INT_MAX*). The value **0** indicates that the function is disabled.

Default value: **'90,0,0,20,10'**, indicating that the memory usage threshold is 90%, the function of triggering memory fluctuation is disabled, the function of triggering statistical period is disabled, the top 20 memory contexts of each type are recorded, and the top 10 SQL statements are recorded.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the parameter description.

Risks and impacts of improper settings: Each part of the parameter must be separated by a comma (,). Therefore, the parameter must contain only four commas (,) and be divided into five parts. Each part must meet the requirements of the preceding value range. If any requirement is not met, the setting fails and all parameter parts do not take effect. Before modifying this parameter, make sure you understand its meaning and fully verify the change in a test environment.

resilience_memory_reject_percent

Parameter description: Specifies the dynamic memory usage percentage for escape from memory overload. This parameter takes effect only when the GUC parameters **use_workload_manager** and **enable_memory_limit** are enabled.

Parameter type: string.

Unit: none

Value range: a string of more than 0 characters.

This parameter consists of **recover_memory_percent** and **overload_memory_percent**. The meanings of the two parts are as follows:

- **recover_memory_percent**: Percentage of the dynamic memory usage when the memory recovers from overload to the maximum dynamic memory. When the dynamic memory usage is less than the maximum dynamic memory multiplied by the value of this parameter, the overload escape function is disabled and new connections are allowed. The value ranges from 0 to 100, which indicates a percentage.
- **overload_memory_percent**: Percentage of the dynamic memory usage to the maximum dynamic memory when the memory is overloaded. When the dynamic memory usage is greater than the maximum dynamic memory

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

multiplied by the value of this parameter, the current memory is overloaded. In this case, the overload escape function is triggered to kill sessions and new connections are prohibited. The value ranges from 0 to 100, which indicates a percentage.

Default value: **"0,0"**, indicating that the escape from memory overload function is disabled.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

The following command is an example. When the memory usage exceeds 90% of the upper limit, new connections are forbidden and stacked sessions are killed. When the memory usage is less than 70% of the upper limit, session killing is stopped and new connections are allowed.

The kill policy is as follows: Idle sessions are killed first; executing sessions are killed in descending order of the memory usage.

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c "resilience_memory_reject_percent='70,90'"
```

Setting suggestion: Set the value based on service requirements.

Risks and impacts of improper settings: If the value is too small, it will frequently trigger the escape from memory overload function, causing ongoing sessions to be forcibly terminated and new connections to fail in a short period. Therefore, exercise caution when setting this parameter based on the actual memory usage.

NOTE

- You can query the maximum dynamic memory and used dynamic memory in the pv_total_memory_detail view. **max_dynamic_memory** indicates the maximum dynamic memory, and **dynamic_used_memory** indicates the used dynamic memory.
- If the **use_workload_manager** parameter is disabled and the **bypass_workload_manager** parameter is enabled, this parameter also takes effect. The **bypass_workload_manager** parameter is of the SIGHUP type; therefore, after the reload mode is set, you need to restart the database for the setting to take effect.
- The values of **recover_memory_percent** and **overload_memory_percent** can be **0** at the same time. In addition, the value of **recover_memory_percent** must be smaller than that of **overload_memory_percent**. Otherwise, the setting does not take effect.
- After the parameter is dynamically set to **"0,0"**, if session killing has been started, the killing action will not be terminated. You can wait for more than 10 seconds and then connect to the database to prevent new connections from being killed.

resilience_escape_user_permissions

Parameter description: Specifies the escape permission of users. You can set it for multiple users and separate users by commas (,). The value **sysadmin** indicates that jobs of the SYSADMIN user can be canceled by the escape function. The value **monadmin** indicates that jobs of the MONADMIN user can be canceled by the escape function. By default, this parameter is left blank, indicating that the escape function of the SYSADMIN and MONADMIN users is disabled. The value can only be **sysadmin**, **monadmin**, or an empty string.

Parameter type: string.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: a string of more than 0 characters.

Currently, this parameter supports only three values: **sysadmin**, **monadmin**, and **""**. The meanings of these values are as follows:

- **sysadmin**: Jobs of the SYSADMIN user can be canceled by the escape function.
- **monadmin**: Jobs of the MONADMIN user can be canceled by the escape function.
- **""**: The escape function of the SYSADMIN and MONADMIN users is disabled.

Default value: **""**, indicating that the escape function of the SYSADMIN and MONADMIN users is disabled.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. You can set this parameter to multiple values separated by commas (,), for example, **resilience_escape_user_permissions = 'sysadmin,monadmin'**. You can also set this parameter to only one value, for example, **resilience_escape_user_permissions = 'monadmin'**.

For example, the following command is used to enable the escape function for both the SYSADMIN and MONADMIN users:

```
gs_guc reload -Z coordinator -Z datanode -N all -I all -c
"resilience_escape_user_permissions='sysadmin,monadmin'"
```

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

- If this parameter is set for multiple times, the latest setting takes effect.
- If this parameter is set to any value in the value range, common users support the escape function.
- If a user has both the SYSADMIN and MONADMIN role permissions, the escape function of the user can be triggered only when **resilience_escape_user_permissions** is set to **"sysadmin,monadmin"**.

start_with_max_mem

Parameter description: Specifies the maximum memory occupied by the START WITH operator. If the START WITH query statement contains a large amount of data, the memory usage may be too high due to excessive recursion depth.

Parameter type: integer.

Unit: KB

Value range: –1 to 2147483647

Default value: **-1**, indicating that the memory occupied by the START WITH operator is not limited.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. The value of **start_with_max_mem** can be specified without a unit, for example, **1024** (indicating 1024 KB), or with a unit, for example, **1MB** (indicating 1 MB). The unit must be KB, MB, or GB if required.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Set this parameter based on the amount of data to be queried. The default value is **-1**. If a large amount of data is queried, the memory usage may be too high. You can adjust this parameter as required to prevent a single query from occupying too much memory. For details, see **work_mem**.

Risks and impacts of improper settings: If **start_with_max_mem** is set to a non-default value or a small value, the query statement fails to be executed.

6.3.4.2 Disk Space

This section describes the disk space parameters, which are used to set limits on the disk space for storing temporary files.

sql_use_spacelimit

Parameter description: Specifies the space size for files to be written to disks when a single SQL statement is executed on a single DN. The managed space includes the space occupied by ordinary tables, temporary tables, and intermediate result sets to be written to disks. This parameter does not take effect for the initial user.

Parameter type: integer.

Unit: KB

Value range: –1 to 2147483647. The value **-1** indicates that there is no limit.

Default value: **-1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. If the value contains a unit, the value can be KB, MB, or GB, but cannot be TB.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after you fully understand the parameter meaning and test the parameter.

temp_file_limit

Parameter description: Specifies the maximum disk space that can be used for temporary files in a session. For example, temporary files that need to be sorted or use outer hash tables in a connection session, and temporary files occupied due to the use of cursors.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This parameter does not apply to disk space used for temporary tables during the SQL query process.
- In the SMP scenario (**query_dop** is greater than or equal to 2), the SQL statement in the foreground session is split into several background threads (the number of split SQL statements is equal to the value of **query_dop**) for execution. When the size of temporary files generated by any thread reaches the threshold, an SQL error is reported in the frontend session. In this case, the size of temporary files may be greater than the value of this parameter set in the frontend session. (In this scenario, the peak size of temporary files occupied by a single frontend session does not exceed the value of **temp_file_limit** multiplied by the value of **query_dop**.)
- When the disk usage exceeds the read-only threshold **datastorage_threshold_value_check** and **default_transaction_read_only** is set to **on**, this parameter is taken over by the CM and is set to **0**. In this case, the modification of **temp_file_limit** does not take effect. After the disk usage is restored, the CM restores this parameter to the last value set by the user before the node becomes read-only. In addition, if disk usage goes above the read-only limit **datastorage_threshold_value_check** and **default_transaction_read_only** is **on**, clearing the disk space to reduce the disk usage to below the limit and setting **default_transaction_read_only** to **off** will be considered as bypassing CM monitoring. In this case, you need to manually change the value of **temp_file_limit**.
- If this parameter is set to the default value **-1**, after the disk usage exceeds the read-only threshold **datastorage_threshold_value_check**, the CM takes over the parameter and kills existing services. During this period, services may still be continuously flushed to disks, which may cause the disk to be full. In this case, you can delete the temporary files generated in **pgsql_tmp** in the data directory to reduce the disk usage.

Parameter type: integer.

Unit: KB

Value range: –1 to 2147483647. The value **-1** indicates that there is no limit.

Default value: **-1**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**. If the value contains a unit, the value can be KB, MB, or GB, but cannot be TB.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, frequent I/O operations will occur, causing performance deterioration. If the value is too large, too much disk space may be occupied.

6.3.4.3 Kernel Resource Usage

This section describes kernel resource parameters. Whether these parameters take effect depends on OS settings.

max_files_per_process

Parameter description: Specifies the maximum number of simultaneously open files allowed by each database thread.

Parameter type: integer.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 25 to 2147483647

Default value: **1024**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If the kernel is enforcing a proper limit, setting this parameter is not required. However, on some platforms, such as most Berkeley Software Distribution (BSD) systems, the kernel allows individual processes to open many more files than the system can support. If the message "Too many open files" is displayed, set this parameter to a smaller value. Generally, the system must meet this requirement: Number of file descriptors ≥ Maximum number of concurrent requests x Number of primary DNs on the current server x Value of **max_files_per_process** x 3.

Risks and impacts of improper settings: Set this parameter according to the actual situation. If the value is too small, the number of valid security file handles may be small. As a result, the cluster status is abnormal.

shared_preload_libraries

Parameter description: Specifies the shared libraries that need to be preloaded when the database is started. Multiple library names can be separated by commas (,). For example, **'$libdir/mylib'** will preferentially load **mylib.so** (**mylib.sl** for some platforms). Only the SYSADMIN user can modify this parameter.

NOTE

- If a specified library is not found, the GaussDB service will fail to start.
- Each GaussDB-supported library has a special mark that is checked to guarantee compatibility. Therefore, libraries that do not support GaussDB cannot be loaded in this way.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **"platform_compatibility_b,security_plugin"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You can preload the GaussDB's stored procedure library using the '*$libdir*/pl*XXX*' syntax as described in the preceding text. *XXX* can only be **pgsql**, **perl**, **tcl**, or **python**.

By preloading a shared library and initializing it as required, the library startup time is avoided when the library is first used. However, the time to start each new server process may increase, even if that process never uses the library. Therefore, set this parameter only for libraries that will be used in most sessions.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3.4.4 Cost-based Vacuum Delay

This feature is designed to reduce the I/O impact of maintenance operations such as VACUUM and ANALYZE in high-concurrency databases. The cost-based vacuum delay mechanism enables administrators to balance resource allocation among maintenance tasks and normal service operations. This feature provides the following functions:

- Non-critical delay: Maintenance operations do not need to be completed immediately and can be executed in the backend.
- Resource isolation: Maintenance tasks are prevented from occupying too many I/O resources, ensuring service query performance.

NOTICE

Certain VACUUM operations hold critical locks and should be complete as quickly as possible. In GaussDB, cost-based vacuum delays do not take effect during such operations. To avoid uselessly long delays in such cases, the actual delay is the larger of the two calculated values:

- **vacuum_cost_delay** x **accumulated_balance**/**vacuum_cost_limit**
- **vacuum_cost_delay** x 4

Context

During the execution of the **ANALYZE | ANALYSE** and **VACUUM** statements, the system maintains an internal counter that keeps track of the estimated cost of the various I/O operations that are performed. When the accumulated I/O cost reaches the threshold specified by **vacuum_cost_limit**, the thread that performs the maintenance operation sleeps for the time specified by **vacuum_cost_delay**. After the sleep ends, the counter is reset, and the operation continues. This feature is disabled by default. You need to set **vacuum_cost_delay** to a non-zero value to enable it.

vacuum_cost_delay

Parameter description: VACUUM starts the backend worker thread to vacuum dead tuples in the data table, which conflicts with the worker thread. If **vacuum_cost_delay** is set to the value of **vacuum_cost_limit** to control the impact of VACUUM on services, the VACUUM thread hibernates the time specified by **vacuum_cost_delay** and allocates resources to worker threads for execution to ensure that services are not blocked by backend VACUUM.

Parameter type: integer.

Unit: milliseconds

Value range: 0 to 100. The value **0** indicates that the cost-based vacuum delay feature is disabled, and a positive value indicates that the cost-based vacuum delay feature is enabled. In many systems, the effective resolution of the hibernation time is 10 milliseconds. Therefore, setting **vacuum_cost_delay** to a value that is not an integer multiple of 10 has the same effect as setting it to the next higher multiple of 10.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is usually set to a small value, such as **10ms** or **20ms**. Adjusting vacuum's resource consumption is best done by changing other vacuum cost parameters.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

vacuum_cost_page_hit

Parameter description: Specifies the estimated cost for VACUUM to clean up data pages hit in a buffer.

Parameter type: integer.

Unit: none

Value range: 0 to 10000

Default value: **1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

vacuum_cost_page_miss

Parameter description: Specifies the estimated cost for VACUUM to clean up data pages that do not hit in a buffer.

Parameter type: integer.

Unit: none

Value range: 0 to 10000

Default value: **10**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

vacuum_cost_page_dirty

Parameter description: Specifies the cost of VACUUM cleanup dirty pages. If the data page to be cleaned up is a clean page, VACUUM cleanup generates an extra cost of flushing dirty pages to disks. You can set this parameter to specify the cost of VACUUM cleanup dirty pages.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: none

Value range: 0 to 10000

Default value: 20

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

vacuum_cost_limit

Parameter description: Specifies the cost limit. The vacuum thread will hibernate if this limit is exceeded. When the execution cost of the VACUUM thread reaches **vacuum_cost_limit**, the thread sleeps for the time specified by **vacuum_cost_delay**.

Parameter type: integer.

Unit: none

Value range: 1 to 10000

Default value: **1000**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. A larger value indicates a smaller I/O frequency limit of the VACUUM, a lower probability of entering the hibernation, more efficient VACUUM, and a greater impact on service I/Os.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

vacuum_defer_cleanup_age

Parameter description: Specifies the delay for the VACUUM operation to clear invalid row-store table records. The delay is calculated based on the number of executed transactions. You can set the value of **vacuum_defer_cleanup_age** to control the number of delayed transactions. That is, the VACUUM and VACUUM FULL operations do not clear a deleted tuple immediately. Instead, they clear and delete the tuple only after a certain number of transactions have been processed.

Parameter type: integer.

Unit: none

Value range: 0 to 1000000. **0** indicates no delay.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, VACUUM and VACUUM FULL cannot be performed in a timely manner.

6.3.4.5 Background Writer

This section describes background writer parameters. The background writer thread is used to write dirty page (new or modified data) in shared buffers to disks. It is designed to minimize or eliminate the waiting time caused by write operations (which are asynchronously performed by the background writer thread) when the database process executes user queries.

This mechanism also effectively reduces performance fluctuations caused by checkpoints. The background writer thread continuously refreshes dirty pages to disks. Therefore, when a checkpoint is triggered, only a small number of pages need to be refreshed. However, this mechanism increases the overall I/O load. In the traditional checkpoint interval, pages are refreshed only once though they may be repeatedly modified, while the background writer thread may refresh the pages multiple times in the same interval. Generally, continuous low load is better than periodic peak load. The parameters discussed in this section can be used to adjust the behavior based on actual requirements.

bgwriter_delay

Parameter description: Specifies the interval for periodically scanning dirty data in the buffer block by block. In full checkpoint mode, the amount of data written each time is controlled by the **bgwriter_lru_maxpages** parameter. After the data is written, the system sleeps for *bgwriter_delay* milliseconds and then repeats the operation. In incremental checkpoint mode, the system calculates the number of target idle buffer pages based on the specified **candidate_buf_percent_target**. If the number of candidate buffer pages is insufficient, the system scans 10 GB buffer pages every *bgwriter_delay* milliseconds. If dirty pages are found, the system flushes them to disks and adds them to the candidate linked list. If clean pages are found, the system directly adds them to the candidate linked list. In many systems, the effective resolution of the sleep delay is 10 milliseconds. Therefore, setting this parameter to a value that is not a multiple of 10 has the same effect as setting it to a value multiple of 10.

Parameter type: integer.

Unit: milliseconds

Value range: 10 to 10000

Default value: **2000** (2s)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- Incremental checkpoint mode: If the data volume is multiple times the value of **shared_buffers**, the value of **bgwriter_delay** cannot be greater than 2s. If the data volume is smaller than the value of **shared_buffers**, the value of **bgwriter_delay** can be increased to save I/O resources.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Full checkpoint mode: Retain the default value. If the disk capability is poor, increase the value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

candidate_buf_percent_target

Parameter description: Specifies the expected percentage of the candidate linked list of clean pages to the total number of pages in the buffer when incremental checkpointing is enabled. If the proportion of pages in the linked list of clean pages is less than the value of this parameter, the pagewriter thread starts, scans the buffer pages, adds clean pages to the linked list, flushes dirty pages to disks, and then adds dirty pages to the linked list.

Parameter type: floating-point number

Unit: none

Value range: 0.1 to 0.85

Default value: **0.3**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter directly affects the number of pages flushed by the pagewriter thread. If the data volume is greater than the value of **shared_buffers** and **candidate_buf_percent_target** is set to an excessively small value, the performance is affected. If the data volume is twice or more of the value of **shared_buffers**, the value of this parameter cannot be smaller than the default value. In other scenarios, to reduce I/Os, set this parameter to a smaller value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

bgwriter_lru_maxpages

Parameter description: Specifies the maximum number of dirty pages that can be flushed by the background writer thread each time.

Parameter type: integer.

Unit: none

Value range: 0 to 1000. **0** indicates that the background writer is disabled. This setting does not affect checkpoints.

Default value: **100**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

bgwriter_lru_multiplier

Parameter description: The database kernel sets **shared_buffers** based on the current buffer size. If the value of **shared_buffers** is too small or too large when the kernel runs services, you can adjust the value of **bgwriter_lru_multiplier** to provide the optimal recommended value, the recommended value is the product of the value of this parameter and the value of **shared_buffers**. After you set this parameter to the recommended value, the setting takes effect when the database is restarted.

Parameter type: floating-point number

Unit: none

Value range: 0 to 10

Default value: **2**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- The number of dirty buffers written in each round depends on the number of buffers used by server processes during recent rounds. The estimated number of buffers used in the next round is calculated using the following formula: Average number of recently used buffers x **bgwriter_lru_multiplier**. The background writer writes dirty buffers until sufficient, clean, and reusable buffers are available. The number of buffers the background writer writes in each round is always equal to or less than the value of **bgwriter_lru_maxpages**.
- Therefore, the value **1.0** of **bgwriter_lru_multiplier** indicates a just-in-time policy of writing exactly the number of dirty buffers predicted to be required. Larger values provide some cushion against spikes in demand, whereas smaller values intentionally leave more writes to be done by server processes.
- Smaller values of **bgwriter_lru_maxpages** and **bgwriter_lru_multiplier** reduce the extra I/O load caused by the background writer, but make it more likely that server processes will have to issue writes for themselves, delaying response to queries.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

pagewriter_thread_num

Parameter description: Specifies the number of threads for backend page flushing after incremental checkpointing is enabled. Data is flushed to disks based on the sequence in which dirty pages are set to dirty, clean buffer pages are generated, and recovery points for checkpointing are pushed.

Parameter type: integer.

Unit: none

Value range: 1 to 16

Default value:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- For the 8-core CPU/64 GB memory specification in hybrid deployment mode, the default value is **1**.
- For all specifications in independent deployment mode and other specifications in hybrid deployment mode, the default value is **4**.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

dirty_page_percent_max

Parameter description: Specifies the maximum expected percentage of dirty pages to **shared_buffers** after the incremental checkpoint is enabled. When the value of this parameter is reached, the background page flush thread flushes dirty pages based on the value of **max_io_capacity**.

Parameter type: floating-point number

Unit: none

Value range: 0.1 to 1

Default value: **0.9**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

pagewriter_sleep

Parameter description: Specifies the dirty data flush period of the pagewriter page flush thread in incremental checkpoint mode.

Parameter type: integer.

Unit: milliseconds

Value range: 0 to 3600000

Default value: **2000** (2s)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value for CNs. Do not set this parameter to a value greater than 2s for DNs. If dirty pages are generated quickly, you are advised to set this parameter to a value ranging from 100 ms to 500 ms. If this parameter is set to a large value, redo points are updated slowly, affecting Xlog recycling.

Risks and impacts of improper settings: If the value is too large, the page flushing speed decreases and redo points are pushed too slowly, affecting Xlog

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

reclamation. In addition, the instance needs to wait for the time specified by this parameter when being stopped. As a result, the process exit times out.

max_io_capacity

Parameter description: Specifies the maximum I/Os per second for the background writer to flush dirty pages in batches.

Parameter type: integer.

Unit: KB

Value range: 30720 to 10485760

Default value: **512000** (that is, 500 MB)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. If the value of the parameter does not contain a unit, the default unit is KB. The unit can also be MB or GB. For example, if **max_io_capacity** is set to **512000**, **512000kB**, or **500MB**, it indicates that **max_io_capacity** is 500 MB.

Setting suggestion: Set this parameter based on the service scenario and disk I/O capability of the host. For example, if the disk I/O bandwidth is 500 MB, you are advised to set the bandwidth to a value less than 500 MB. If a shorter RTO is required or a large capacity is required, you can increase the value of **max_io_capacity**, for example, 2 GB.

Risks and impacts of improper settings: If the value is too small, the dirty page flushing speed is slower than the dirty page generation speed. As a result, the dirty pages in the buffer are full and the performance deteriorates. If the value is too large, write amplification occurs, affecting the service life of the storage device.

enable_consider_usecount

Parameter description: Specifies whether the page eviction algorithm of the buffer considers the page popularity.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The page popularity is considered.
- **off**: The page popularity is not considered.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Enable this parameter in scenarios where single-page elimination is frequently performed, such as large-capacity scenarios.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_buffer_evictor

Parameter description: Specifies whether to enable the intelligent eviction algorithm.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The intelligent eviction algorithm is enabled.
- **off**: The intelligent eviction algorithm is disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Enable this parameter in scenarios where single-page elimination is frequently performed, such as large-capacity scenarios.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

eviction_clean_page_percent_min

Parameter description: Specifies the minimum number of clean pages reserved by the intelligent eviction algorithm. If the number of clean pages is less than the value of this parameter, the generation rate of candidate pages to be evicted is limited. If the value is set to **0**, the generation rate is not limited.

Parameter type: floating-point number

Unit: none

Value range: 0 to 0.9

Default value: **0.1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Enable this parameter in scenarios where single-page elimination is frequently performed, such as large-capacity scenarios.

Risks and impacts of improper settings: If the value is too large, the dirty page flushing performance may deteriorate.

eviction_candidate_list_size

Parameter description: Specifies the length of the eviction queue of the intelligent eviction algorithm.

Parameter type: integer.

Unit: none

Value range: 1024 to 65536

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **2048**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Enable this parameter in scenarios where single-page elimination is frequently performed, such as large-capacity scenarios.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

pagewriter_flush_mode

Parameter description: Specifies whether to enable the ultimate dirty page flushing function.

Parameter type: enumerated type

Unit: none

Value range:

- **extre_mode**: The ultimate dirty page flushing function is enabled.
- **default_mode**: The ultimate dirty page flushing function is disabled.

Default value: **default_mode**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Enable this parameter in scenarios where single-page elimination is frequently performed, such as large-capacity scenarios.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

dw_file_num

Parameter description: Specifies the number of batch doublewrite files.

Parameter type: integer.

Unit: none

Value range: 1 to 16

Default value: **1**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: The value of this parameter is related to **pagewriter_thread_num** and cannot be greater than that of **pagewriter_thread_num**. If the value of **dw_file_num** is too large, the value is internally corrected to the value of **pagewriter_thread_num**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

dw_file_size

Parameter description: Specifies the size of each doublewrite file.

Parameter type: integer.

Unit: MB

Value range: 32 to 256

Default value: **256**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

postmaster_parallel_init_thread_num

Parameter description: Specifies the number of parallel threads that can be used when the database is started.

Parameter type: integer.

Unit: none

Value range: 1 to 256

Default value: **16**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter indicates the number of threads for parallel initialization. Therefore, you are advised to set this parameter to a value less than or equal to the number of CPU cores minus 1.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.4.6 Asynchronous I/O

checkpoint_flush_after

Parameter description: Specifies the threshold for the number of dirty pages flushed by the checkpointer thread. If the number of pages exceeds the threshold, the thread instructs the OS to asynchronously flush the data pages cached in the file to disks. In GaussDB, the disk page size is 8 KB.

Parameter type: integer

Unit: page (8 KB)

Value range: 0 to 256. **0** indicates that the asynchronous flush function is disabled.

Default value: **256kB** (32 pages)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. You can set the number of pages or the size in bytes, for example, **32** or **256kB**. After the checkpointer thread continuously writes 32 disk pages, that is, 32 x 8 = 256 KB disk space, the asynchronous flush is performed. **gs_guc** does not allow you to set the number of pages as a parameter unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

bgwriter_flush_after

Parameter description: Specifies the threshold for the number of dirty pages flushed by the background writer thread. If the number of pages exceeds the threshold, the thread instructs the OS to asynchronously flush the data pages cached in the file to disks. In GaussDB, the disk page size is 8 KB.

Parameter type: integer

Unit: page (8 KB)

Value range: 0 to 256. **0** indicates that the asynchronous flush function is disabled.

Default value: **512kB** (64 pages)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. You can set the number of pages or the size in bytes, for example, **64** or **512kB**. After the background writer thread continuously writes 64 disk pages, that is, 64 x 8 = 512 KB disk space, the asynchronous flush is performed. **gs_guc** does not allow you to set the number of pages as a parameter unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

backend_flush_after

Parameter description: Specifies the threshold for the number of dirty pages flushed by the backend thread. If the number of pages exceeds the threshold, the thread instructs the OS to asynchronously flush the data pages cached in the file to disks. In GaussDB, the disk page size is 8 KB.

Parameter type: integer

Unit: page (8 KB)

Value range: 0 to 256. **0** indicates that the asynchronous flush function is disabled.

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. You can set the number of pages or the size in bytes, for example, **64** or **512kB**. After the backend thread continuously writes 64 disk pages, that is, 64 x 8 = 512 KB disk space, the asynchronous flush is performed. **gs_guc** does not allow you to set the number of pages as a parameter unit.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_adio_function

Parameter description: Specifies whether to enable the ADIO mode.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The ADIO mode is enabled.
- **off**: The ADIO mode is disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. You can change the value of this parameter from **off** to **on** by referring to "Method 2" in **Table 6-2**. After the setting is complete, you can use the system function **gs_get_io_type()** to check whether the ADIO mode is successfully switched. If you want to change the value of this parameter from **on** to **off**, restart the instance. For details, see "Method 1" in **Table 6-2**.

Setting suggestion: If the **candidate_slots** column in the DBE_PERF.global_candidate_status system view is less than 15% of the buffer page for multiple times in a short period of time, you are advised to set this parameter to **on**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

If this parameter is enabled, a low configuration of **/proc/sys/fs/aio-max-nr** may stop the AIO Completer thread from starting. You are advised to increase the value of **/proc/sys/fs/aio-max-nr** to a value greater than: Maximum number of instances on the node x Maximum number of AIO threads (5 in the current version) x Maximum depth of the AIO completion queue (10,000 in the current version) + 5,000.

effective_io_concurrency

Parameter description: Specifies the number of requests that are simultaneously processed by a disk subsystem. For a RAID array, the value is the number of disk drive spindles in the array.

Parameter type: integer

Unit: none

Value range: 0 to 1000

Default value: **1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Set this parameter to the number of disks.

Risks and impacts of improper settings: If the value is too large and the related logic is executed, the I/O resources of the entire disk may be occupied, causing I/O peak.

6.3.5 Parallel Data Import and Export

GaussDB provides a parallel data import and export function that enables a large amount of data to be imported and exported in a fast and efficient manner. This section describes parameters for importing and exporting data to GaussDB in parallel.

raise_errors_if_no_files

Parameter description: Specifies whether to distinguish between "the number of imported file records is empty" and "the imported file does not exist." If this parameter is enabled and the imported file does not exist, GaussDB displays an error message indicating that the file does not exist.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The messages of "the number of imported file records is empty" and "the imported file does not exist" are distinguished when files are imported.
- **off**: The messages of "the number of imported file records is empty" and "the imported file does not exist" are the same when files are imported.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

The **raise_errors_if_no_files** parameter is valid only when the distributed GDS tool is used.

gds_debug_mod

Parameter description: Specifies whether to enable the debug function of Gauss Data Service (GDS). This parameter is used to better locate and analyze GDS faults. After the debug function is enabled, types of packets received or sent by GDS, peer end of GDS during command interaction, and other interaction information about GDS are written into the logs of corresponding nodes in the cluster. In this way, the state switching on the GaussDB state machine and the current state are recorded.

Parameter type: Boolean.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **on**: The GDS debug function is enabled.
- **off**: The GDS debug function is disabled.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Enable this function only when locating GDS problems.

Risks and impacts of improper settings: If this parameter is enabled, extra logs will be generated, increasing the log I/O overhead and affecting the performance and log validity.

safe_data_path

Parameter description: Specifies the path required when the COPY, LOAD DATA, and foreign tables of the local node type, and advanced packages of non-initial users are used to read and write files. Whitelist management is implemented on related paths. Only files in the whitelisted paths can be read and written.

Parameter type: string.

Unit: none

Value range: valid directory path containing a maximum of 4096 characters.

Default value: **"copydir_default,copydir_logical_decode"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Ensure that the path **safe_data_path** covers a limited range. Do not set a large path range.

Risks and impacts of improper settings: If **safe_data_path** is left empty and **safe_data_path** is set to a large value, malicious users may access sensitive files on the server.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If a soft link file exists in the **safe_data_path** directory, the system processes the file based on the actual file path to which the soft link points. If the actual path is not in the **safe_data_path** directory, an error is reported.
- If a hard link file exists in the **safe_data_path** directory, it can be used properly. For security purposes, exercise caution when using hard link files. Do not create hard link files that point to other directories in the **safe_data_path** directory. Ensure that the permission on the **safe_data_path** directory is minimized.
- If the **safe_data_path** parameter is left empty, whitelist verification is disabled, which is used for compatibility with earlier versions. Exercise caution when setting this parameter after evaluating the risks and impacts.
- In the value of **safe_data_path**, different paths must be separated by commas (,). The separators cannot contain spaces or tab characters.
- **copydir_default** in the default value of this parameter is a placeholder, indicating the default path that can be used in the whitelist. The actual path is *$GAUSSLOG*/**gs_copy_data**. If the path does not exist, you can manually run the **COPY table_name FROM 'file_name'** or **COPY table_name TO 'file_name'** command to create the path, or run the **mkdir** command in the corresponding directory to create the path. When the whitelist function is enabled, you are advised not to delete this parameter.
- The default value **copydir_logical_decode** of this parameter is the path required for logical decoding. Do not delete it. Otherwise, related functions may be affected.
- When modifying the **safe_data_path** parameter, you can only add paths. Paths in the parameter value are separated using commas (,). Do not delete the default path.

enable_copy_server_files

Parameter description: Specifies whether to enable the permission to copy server files.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The permission to copy server files is enabled. Users with the SYSADMIN permission or users inherited from the built-in role gs_role_copy_files can run the **COPY FROM FILENAME** or **COPY TO FILENAME** command.
- **off**: The permission to copy server files is disabled. Only the initial user is allowed to run the **COPY FROM FILENAME** or **COPY TO FILENAME** command.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestions: Enable this function only when you need to copy server files. After this function is enabled, you are advised to set **safe_data_path** synchronously.

Risks and impacts of improper settings: After this parameter is enabled, malicious users may access sensitive files on the server if the value range of **safe_data_path** is too large.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

support_binary_copy_version

Parameter description: Specifies whether the encoding information of the current database server is included when data is exported in BINARY mode using COPY TO.

Parameter type: string.

Unit: none

Value range: **""** and **"header_encoding"**.

Table 6-14 Configuration items

Configuration Item	Behavior
"header_encoding"	When the BINARY mode of COPY TO is used to export data, the binary file header contains the encoding information of the current database server.
""	Compatibility configuration with earlier versions is performed and data is exported in the original binary format.

Default value: **"header_encoding"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If compatibility with earlier versions is required, leave this parameter empty.

Risks and impacts of improper settings: If this parameter is set to an empty string, the exported result does not contain the server-side encoding information. In scenarios where the encoding information is required, you need to query and record the information.

copy_special_character_version

Parameter description: Determines the processing of invalid characters during data import and export using COPY.

Parameter type: string.

Unit: none

Value range: **""**, **"no_error"**, and **"per_byte"**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-15 Configuration items

Configuration Item	Behavior
"no_error"	When COPY is used to import a data file with the same encoding as that on the server, fault tolerance is performed on the data that does not meet the encoding requirements in the data file. No error is reported and the data with the original codes is directly inserted into the table.
"per_byte"	Determines how to process files encoded in GBK or ZHS16GBK when COPY is used to export text files. After the parameter is set to **"per_byte"**, one byte of data is exported at a time. Otherwise, two bytes of data are exported at a time. (One character occupies two bytes if data is encoded in GBK.)
""	The default value, which does not affect any function. Compatibility with earlier versions is supported. That is, an error is reported when invalid characters are found during COPY.

Default value: ""

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. Use gsql to connect to the database. If you use the set method, the value is case-insensitive. If you use gs_guc, the value can only be lowercase.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

- To ensure that the data to be imported is valid, its encoding must be validated when it is being copied. If this parameter is enabled, verification against invalid encoding will be masked, which causes invalid characters in the field. Therefore, exercise caution before enabling this parameter.
- Currently, data encoding verification is masked only when the server-side encoding is the same as the data encoding. That is, if **copy_special_character_version** is set to **"no_error"**, the database server-side encoding must be the same as the data file encoding. Otherwise, an error is reported. If no data encoding is specified, the client-side encoding is used by default.
- In binary mode, **copy_special_character_version** is set to **"no_error"**, and it takes effect only for fields of the TEXT, CHAR, VARCHAR, NVARCHAR2, or CLOB type.
- This parameter is valid only in the database with character sets encoded in UTF-8, GB18030, GB18030_2022, ZHS16GBK, or LATIN1.
- When the encoding of both the client and server is GBK or ZHS16GBK and the database contains data encoded in an invalid format, if **copy_special_character_version** is not set to **"per_byte"**, the exported data file may contain unexpected data.
- The priority of setting **copy_special_character_version** to **"no_error"** is higher than that of **COMPATIBLE_ILLEGAL_CHARS** in COPY.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_log_copy_illegal_chars

Parameter description: Specifies whether to write records to the database run log if invalid characters are encountered when COPY is used to import or export data.

Parameter type: Boolean

Unit: none

Value range: **on** and **off**

Table 6-16 Configuration items

Configuration Item	Behavior
on	Each time a line of data contains invalid encoding characters, a record is written to the database run log.
off	Records related to invalid encoding characters are not written into database run logs.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value unless otherwise specified. Set this parameter to **off** only when there are requirements on database performance and disk bandwidth.

Risks and impacts of improper settings: If this parameter is set to **off**, records related to invalid encoding characters will not be written into run logs, affecting the fault locating capability.

batch_insert_index_types

Parameter description: Specifies whether to optimize the performance of batch index insertion in specific scenarios.

Parameter type: string.

Unit: none

Value range: **"rcr_ubtree"** and **""**.

Table 6-17 Configuration items

Configuration Item	Behavior
"rcr_ubtree"	The batch index insertion feature is enabled when indexes of the **rcr_ubtree** type are inserted in batches.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
""	The batch index insertion feature is not enabled for any type of index.

Default value: **""**.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If the performance deteriorates greatly, set this parameter to **""**.

NOTE

If any of the following scenarios meet all the specified conditions, enable the GUC parameter to improve the performance of inserting indexes in batches:

Scenario 1: Use JDBC to connect to the database and use the Java batch data insertion API executeBatch to insert indexes in batches.

Scenario 2: Run the **insert into t1 select * from t2** statement. The schemas of the **t1** and **t2** tables are the same, and **t1** contains indexes. This statement supports batch index insertion to **t1**. The stream plan must be enabled (**enable_fast_query_shipping** is set to **off**). Indexes can be inserted in batches only when table data is inserted in batches (you can run the **explain verbose** command to view the table data).

Scenario 3: Run the **copy t1 from 'file.csv' delimiter ',' csv header** statement, where **t1** and **file.csv** have the same schema, and **t1** contains indexes. This statement can be used to insert indexes in batches to the **t1** table.

Condition 1: The inserted index is a UB-tree RCR index.

Condition 2: The inserted data does not increase incrementally. (In this case, the original mechanism uses the cache, and the performance is not improved.)

Condition 3: **enable_stream_operator** is set to **on**.

Condition 4: Batch index insertion is not performed in an SMP environment.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

a_format_enable_copy_empty_lobs

Parameter description: Specifies whether empty strings can be inserted into the BLOB and CLOB types through empty string input when COPY FROM is executed in Oracle-compatible mode.

Parameter type: Boolean.

Unit: none

Value range: **on** and **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-18 Configuration items

Configuration Item	Behavior
on	In Oracle-compatible mode, empty strings can be inserted into CLOB and BLOB types using empty string input. For example, after **1,"",3** is inserted into the table, the result of the second column is empty. However, **where col2 is null** cannot be used for filtering, and the column length is 0.
off	In Oracle-compatible mode, empty strings can be inserted into CLOB and BLOB types using empty string input. An empty string is converted to NULL for storage. For example, after **1,"",3** is inserted into the table, the result of the second column is **NULL**. In this case, **where col2 is null** can be used to filter the result.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If compatibility with earlier versions is required, set this parameter to **off**.

NOTE

- Null string input and null value input:

  In the CSV format of COPY, **1,,3** indicates a null value, and **1,"",3** indicates a null string. When the parameter is enabled, the second column of the former is inserted as **NULL**, and the second column of the latter is inserted as an empty string. When the parameter is disabled, the second columns of the former and latter are inserted as **NULL**.
- For a newly installed database, the default value is **on**. For an upgraded database, the default value is **off** for compatibility with earlier versions.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_copy_case_sensitive

Parameter description: Specifies whether to retain the case sensitivity of column names when expressions are processed during data import using gs_loader or COPY...TRANSFORM() and uppercase columns or special column names are encountered.

Parameter type: Boolean

Unit: none

Value range: **on** and **off**

Default value: **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Set this parameter to **on** if uppercase column names or special character column names exist and expressions need to be used to process the columns during data import.

NOTE

- In the **gs_loader** control file, uppercase and special character column names are converted to lowercase column names by default when expressions are used to process them. If uppercase or special character column names exist, set **enable_copy_case_sensitive** to **on** and use double quotation marks ("") in the control file to escape the column names in the control file. The following is an example:

  ```
  LOAD DATA
  CHARACTERSET UTF8
  truncate into table "IMP_COLNAME_TEST"
  FIELDS TERMINATED BY ","
  (
  id integer external,
  name char(32),
  "Column_Mixed" "TRIM(:""Column_Mixed"")",
  "Col With Space" "TRIM(:""Col With Space"")",
  "COL_UPPER" "TRIM(:""COL_UPPER"")",
  "Col""With""Quote" "TRIM(:""Col""""With""""Quote"")",
  )
  ```

- When data imported using COPY...TRANSFORM () needs to be processed using an expression, use single quotation marks ('') to identify column names in the expression. The following is an example:

  ```
  COPY "IMP_COLNAME_TEST" ( id, name, "Column_Mixed","Col With Space",
  "COL_UPPER", "Col""With""Quote", con, col_lower )
  FROM 'case.csv'
  TRANSFORM("Column_Mixed" AS TRIM(:"Column_Mixed"),"Col With Space" AS
  TRIM(:"Col With Space"),"COL_UPPER" AS TRIM(:"COL_UPPER"),"Col""With""Quote" AS
  TRIM(:"Col""With""Quote")) ;
  ```

Risks and impacts of improper settings: If this parameter is set to **off** by default, the column names in the expression are case-insensitive and are converted to lowercase letters. If the actual column names are uppercase letters, an import error may occur. If this parameter is set to **on**, the column names in the expression are used to import data.

6.3.6 Write Ahead Log

6.3.6.1 Settings

wal_level

Parameter description: Specifies the level of information to be written to the WAL. The value cannot be empty or commented out.

Parameter type: enumerated type

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range:

- **minimal**

  Advantages: Certain bulk operations (including creating tables and indexes, executing cluster operations, and copying tables) are safely skipped in logging, which improves operational efficiency.

  Disadvantages: WALs contain only basic information required for recovery from a database server crash or an emergency shutdown. Data cannot be restored from archived WALs.

- **archive**

  Adds information required for WAL archiving, supporting the database restoration from archives.

- **hot_standby**
  - Further adds information required to run SQL queries on a standby node and takes effect after the database restarts.
  - To enable read-only queries on a standby node, the **wal_level** parameter must be set to **hot_standby** or a higher level on the primary node and **hot_standby** must be enabled on the standby node. The performance difference between the **hot_standby** and **archive** levels is small.

- **logical**

  Only when this parameter is set to **logical**, logical logs can be parsed and the primary key information is recorded in Xlogs.

Default value: **hot_standby**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter to **logical** when the logical replication function is enabled. In other scenarios, set this parameter to **hot_standby**. The parameter levels meet the following priority: **logical** > **hot_standby** > **archive** > **minimal**.

NOTICE

- To enable WAL archiving and data streaming replication between the primary and standby nodes, set this parameter to **archive**, **hot_standby**, or **logical**.
- If this parameter is set to **archive** or **minimal**, **hot_standby** must be set to **off**. In a distributed system, **hot_standby** cannot be set to **off** because this setting can affect other features of the HA system. Therefore, you are advised not to set this parameter to **archive** or **minimal**. Otherwise, the database cannot be started.

Risks and impacts of improper settings: When the read function of the standby node is enabled and the level is lower than **hot_standby**, the node cannot be started.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

fsync

Parameter description: Specifies whether the GaussDB server uses the fsync function to ensure that updated data is written to physical disks in a timely manner. For details about the implementation of the fsync function, see **wal_sync_method**.

NOTICE

- Using the fsync() function ensures that the data can be recovered to a known state when the OS or hardware crashes.
- Setting this parameter to **off** may result in unrecoverable data corruption in a system crash.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The fsync() function is used.
- **off**: The fsync() function is not used.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

wal_sync_method

Parameter description: Specifies the method for forcibly updating WAL data to disks when **fsync** is enabled.

NOTICE

If **fsync** is set to **off**, the setting of this parameter does not take effect because WAL updates will not be forcibly written to disks.

Parameter type: enumerated type

Unit: none

Value range:

- **open_datasync**: WAL files are opened using open() with the **O_DSYNC** option.
- **fdatasync**: fdatasync() is called at each commit (SUSE 10 and SUSE 11 are supported).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **fsync_writethrough**: fsync() is called at each commit to force data in the buffer to be written to disks.
- **fsync**: fsync() is called at each commit. (SUSE 10 and SUSE 11 are supported.)
- **open_sync**: open() with the **O_SYNC** option is used to write WAL files (SUSE 10 and SUSE 11 are supported).

NOTE

- Not all platforms support the preceding parameters.
- **wal_sync_method** can be set to **fsync_writethrough** on a Windows platform, but this setting has the same effect as setting the parameter to **fsync** on the Windows platform.

Default value: fdatasync

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

synchronous_commit

Parameter description: Specifies the synchronization mode of the current transaction.

Generally, logs generated by a transaction are synchronized in the following sequence:

1. The primary node writes the logs to the local memory.
2. The primary node writes the logs in the local memory to the local file system.
3. The primary node flushes the logs in the local file system to disks.
4. The primary node sends the logs to the standby node.
5. The standby node receives the logs and saves them to the local memory.
6. The standby node writes the logs in the local memory to the local file system.
7. The standby node flushes the logs in the local file system to disks.
8. The standby node replays the logs to complete the incremental update of data files.

Parameter type: enumerated type

Unit: none

Value range:

- **on**: The primary node waits for the standby node to flush logs to disks before committing a transaction.
- **off**: The primary node commits transactions without waiting for the primary node to flush logs to disks. This mode is also called asynchronous commit.
- **local**: The primary node waits for the primary node to flush logs to disks before committing a transaction. This mode is also called local commit.
- **remote_write**: The primary node waits for the standby node to write logs to the file system before committing a transaction. (The logs do not need to be flushed to disks.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **remote_receive**: The primary node waits for the standby node to receive logs before committing a transaction. (The logs do not need to be written to the file system.)
- **remote_apply**: The primary node waits for the standby node to complete log replay before committing a transaction.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. The **synchronous_commit** parameter has been taken over by the CM. In normal cases, the automatic copy upgrade function is enabled by default (that is, the CM parameter **enable_az_auto_switchover** is set to **on**). In this case, the modification of the **synchronous_commit** parameter does not take effect. Only multi-AZ scenarios are supported. To modify this parameter, you need to disable the automatic copy upgrade function (that is, set the CM parameter **enable_az_auto_switchover** to **off**). In this case, the modification of **synchronous_commit** takes effect.

Setting suggestion: Retain the default value. If you manually change the value of this parameter to a non-default value, data may be lost and this parameter cannot be put into commercial use.

Risks and impacts of improper settings: This parameter is maintained by the CM. If it is manually modified, data may be lost.

full_page_writes

Parameter description: Specifies whether the GaussDB server writes the entire content of each disk page to WALs during the first modification of that page after a checkpoint. When both **enable_incremental_checkpoint** and **enable_double_write** are enabled, **full_page_writes** is disabled.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: This feature is enabled.
- **off**: This feature is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. After this feature is enabled, a complete page is written to a disk. During a crash and restart, the page can be correctly restored to ensure data integrity at the cost of increasing the amount of data written to WALs.

Risks and impacts of improper settings: After this feature is disabled, the OS crash may cause partial page write to the disk. As a result, the original data cannot be restored. If the server hardware (for example, a disk controller with battery power supply) or file system feature (for example, Reiser4) can effectively reduce the page write risk and the user is aware of the risk, you can disable this parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wal_log_hints

Parameter description: Specifies whether to write an entire page to WALs during the first modification of that page after a checkpoint, even for non-critical modifications of so-called hint bits.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The entire page is written to WALs.
- **off**: The entire page is not written to WALs.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

wal_buffers

Parameter description: Specifies the size of shared memory for storing WALs, in multiples of **XLOG_BLCKSZ** or the actual size. **XLOG_BLCKSZ** indicates the size of an Xlog block, typically 8 KB.

Parameter type: integer.

Unit: 8 KB (**XLOG_BLCKSZ** size)

Value range: –1 to 262144 (number of Xlog blocks)

- If this parameter is set to **-1**, the **wal_buffers** size is dependent on the value of **shared_buffers**. The default value of **wal_buffers** is 1/32 of **shared_buffers**. The minimum value is **XLOG_BLCKSZ** multiplied by 8 and the maximum value is **XLOG_BLCKSZ** multiplied by 2048. If the automatically adjusted value is less than the minimum value, the value is forcibly set to the minimum value. If the value is greater than the maximum value, the value is forcibly set to the maximum value.
- If this parameter is set to a value smaller than 4 (that is, **0**, **1**, **2**, or **3**), the value **4** is forcibly used.

Default value:

- Independent deployment: 1 GB (128-core CPU/1024 GB memory, 60-core CPU/480 GB memory and 32-core CPU/256 GB memory); 512 MB (16-core CPU/128 GB memory); 256 MB (8-core CPU/64 GB memory); 128 MB (4-core CPU/32 GB memory); 64 MB (4-core CPU/16 GB memory)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, the value **2048** indicates that **wal_buffers** is 2048 x 8 KB, and the value **20480kB** indicates that **wal_buffers** is 20480 KB. The unit must be KB, MB, or GB if required. If the value contains a unit and is not an

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

integer multiple of 8 KB, the value is automatically rounded down to an integer multiple of 8 KB.

Setting suggestion: The content of the WAL buffer is written to disks at every transaction commit. Therefore, setting an extremely large value is unlikely to bring a significant increase in system performance. However, setting this parameter to a few megabytes can improve the disk write performance on a server where a large number of transactions are committed at the same time. According to experience, the default value can meet the requirements of most cases. For different specifications, use the default values in the corresponding suggestions. You are advised to retain the default value. Do not set **wal_buffers** to an excessively large or small value, and the following condition must be true: **data_replicate_buffer_size** + **segment_buffers** + **shared_buffers** + **wal_buffers** + **temp_buffers** + **maintenance_work_mem** + **work_mem** + **query_mem** + (Standby node) **wal_receiver_buffer_size** < **max_process_memory** < Memory size of the physical machine. If the value of the memory parameter is too large and exceeds the upper limit of the physical memory, the database fails to be started because no sufficient memory can be allocated.

Risks and impacts of improper settings: The default value is the recommended value. You are advised not to change it. Before changing the values, check the specifications of the parameters and ensure that the hardware resources are sufficient. Otherwise, the cluster may be abnormal.

wal_writer_delay

Parameter description: Specifies the delay between activity rounds for the WAL writer.

Parameter type: integer.

Unit: milliseconds

Value range: 1 to 10000

Default value: **200**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If time is too long, the memory of the WAL buffer may be insufficient. If time is too short, WALs are continuously written, increasing the disk I/O load.

commit_delay

Parameter description: Specifies the duration for committed data to be stored in the WAL buffer.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- When this parameter is set to a non-zero value, the committed transaction is stored in the WAL buffer instead of being written to the WAL immediately. Then the WAL writer flushes the buffer to disks periodically.
- If system load is high, other transactions are probably ready to be committed within the delay. However, if no transaction is ready to be committed, this delay may cause unnecessary waiting.

Parameter type: integer.

Unit: μs

Value range: 0 to 100000. **0** indicates no delay.

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: For details, see the notice.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

commit_siblings

Parameter description: Specifies a threshold on the number of concurrent open transactions. If the number of concurrent open transactions is greater than the value of this parameter, a transaction that initiates a commit request will wait for a period of time specified by **commit_delay**. Otherwise, this transaction is written into WALs immediately.

Parameter type: integer.

Unit: none

Value range: 0 to 1000

Default value: **5**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

wal_block_size

Parameter description: Specifies the size of a log page in a WAL segment file.

Parameter type: integer.

Unit: byte

Value range: **8192**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **8192**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

wal_segment_size

Parameter description: Specifies the size of a WAL segment file. It can be specified by the multiples of **XLOG_BLCKSZ** or the actual size. **XLOG_BLCKSZ** indicates the size of an Xlog block, typically 8 KB.

Parameter type: integer.

Unit: 8 KB

Value range: **2048** (16 MB)

Default value: **16MB** (2048 x 8 KB)

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

force_promote

Parameter description: Specifies whether to enable the forcible switchover function on the standby node.

NOTICE

When a cluster is faulty, the forcible switchover enables the cluster to recover services as soon as possible at the cost of losing some data. This is an escape method used when the cluster is unavailable. You are advised not to trigger this method frequently. You are advised not to use this function if you are not clear about the impact of data loss on services.

Parameter type: integer.

Unit: none

Value range:

- **0**: disabled.
- **1**: enabled.

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. To use the forcible standby node switchover function,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable this parameter for both DNs and CM Servers. The setting takes effect after the cluster is restarted.

Setting suggestion: Do not enable this parameter.

Risks and impacts of improper settings: After this function is enabled, some data will be lost if a forcible switchover is triggered.

wal_file_init_num

Parameter description: Specifies the number of Xlog segment files created at a time when the WAL writer auxiliary thread is started.

Parameter type: integer.

Unit: none

Value range: 0 to 1000000

Default value: **10**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, too much disk space (**wal_file_init_num** x **wal_segment_size**) will be occupied.

wal_file_preinit_bounds

Parameter description: Specifies the maximum number of WAL segment files that can be pre-expanded by the WAL writer auxiliary thread per second during service running. The WAL segment file size is 16 MB. If this parameter is set to **0**, there is no limit.

Parameter type: integer

Unit: none

Value range: 0 to 1024

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Set this parameter to a valid value based on the disk I/O capability. For details, see the value of **max_io_capacity**.

- If **max_io_capacity** is set to **500MB**, set this parameter to **25**.
- If **max_io_capacity** is set to **1GB**, set this parameter to **50**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

walwriter_sleep_threshold

Parameter description: Specifies the number of times that idle Xlogs are refreshed before the Xlog flusher enters the sleep state. If the number of times reaches the threshold, the Xlog flusher enters the sleep state.

Parameter type: integer.

Unit: none

Value range: 1 to 50000

Default value: **500**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

wal_flush_timeout

Parameter description: Specifies the timeout interval for traversing **walInsertStatusTable**. It is the maximum wait time for the adaptive Xlog disk flushing I/O to traverse **walInsertStatusTable**.

Parameter type: integer.

Unit: μs

Value range: 0 to 90000000

Default value: **2**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the Xlog flushing frequency may decrease and the Xlog processing performance may deteriorate.

xlog_file_path

Parameter description: Specifies the path of the Xlog shared disk in dual-cluster shared storage scenarios. This parameter is configured by the OM during database system initialization. You are advised not to modify the configuration.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **""**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: This parameter is set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The database cannot be started.

xlog_file_size

Parameter description: Specifies the size of the Xlog shared disk in dual-cluster shared storage scenarios. This parameter is configured by the OM during database system initialization. You are advised not to modify the configuration.

Parameter type: integer.

Unit: byte

Value range: 134217728 to 576460752303423487

Default value: **549755813888**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: This parameter is set during installation. You are advised not to change the value.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

xlog_lock_file_path

Parameter description: Specifies the path of the lock file preempted by the Xlog shared disk in dual-cluster shared storage scenarios. This parameter is configured by the OM during database system initialization. You are advised not to modify the configuration.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **""**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The database cannot be started.

max_size_for_shared_storage_xlog_write

Parameter description: Specifies the maximum amount of data that can be written to a shared disk at a time.

Parameter type: integer.

Unit: KB

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 8 to 131072

Default value: 1024

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is set during installation. You are advised not to change the value.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

shared_storage_write_mode

Parameter description: Specifies the maximum Xlog write position of the primary cluster in shared storage (for example, OceanStor Dorado devices).

Parameter type: enumerated type

Unit: none

Value range:

- **none**: No limit.
- **standby**: maximum local Xlog LSN of the connected standby node in the cluster plus the value of **xlog_file_size** (size of the Xlog shared disk in dual-cluster shared storage scenarios).
- **disaster_standby**: maximum local LSN of the standby node connected to the standby node in the standby cluster plus the value of **xlog_file_size**.
- **disaster_standby_strict**: maximum local LSN of the standby node in the standby cluster for DR plus the value of **xlog_file_size**. (Different from **disaster_standby**, the write of this parameter is not overwritten even if the data plane network between the primary and standby clusters is isolated.)
- **standby_and_disaster_standby**: smaller value between the maximum local LSNs of the connected standby nodes in the cluster and the connected standby nodes in the standby cluster for DR plus the value of **xlog_file_size**.

Default value: **disaster_standby**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Modifying this parameter may cause the standby cluster to be rebuilt. You are advised not to manually modify this parameter.

enable_shared_storage_optimization

Parameter description: Specifies whether to enable performance optimization on shared storage (for example, OceanStor Dorado devices).

Parameter type: Boolean.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **on**: enables performance optimization in shared storage mode.
- **off**: disables performance optimization in shared storage mode.

Default value:

- For a newly installed database: **on**.
- For a database upgraded from a version earlier than 507.0.0: **off**.
- For a database upgraded from 507.0.0 or later: original parameter value.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the performance of shared storage in low-concurrency scenarios will be affected.

enable_wal_insert_record_single

Parameter description: Specifies whether to enable the **Single** log insertion mode for instances on Arm-based servers. Instances on x86-based servers are not affected.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The **Single** log insertion mode is enabled.
- **off**: The **Single** log insertion mode is disabled.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- In small-scale or 256-core scenarios, you are advised to enable this function to avoid invalid scheduling overhead.
- In high-concurrency multi-core scenarios, you are advised to disable this function to avoid a large number of CAS lock contentions.
- The default value of the 256-core, 2048 GB egress parameter is **on**.

Risks and impacts of improper settings: If this parameter is enabled, the performance in high-concurrency multi-core scenarios will be affected.

6.3.6.2 Checkpoints

checkpoint_segments

Parameter description: Specifies the maximum number of WAL files that can be retained in the period specified by **checkpoint_timeout**. The size of each log file is 16 MB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483646

Default value: 1024

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Increasing the value of this parameter speeds up the import of a large amount of data. Set this parameter based on **checkpoint_timeout** and **shared_buffers**. This parameter affects the number of WAL files that can be reused. Generally, the maximum number of reused files in the **pg_xlog** folder is twice the value of **checkpoint_segments**. The reused files are not deleted and are renamed to the WAL segment files which will be later used.

Risks and impacts of improper settings: If the value is too small, checkpoints are frequently triggered. If the value is too large, the RTO may be increased, the number of WAL files to be retained increases, and more disk space is occupied.

checkpoint_timeout

Parameter description: Specifies the maximum time between automatic checkpoints.

Parameter type: integer.

Unit: seconds

Value range: 30 to 3600

Default value: **900** (that is, 15 minutes)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If the value of **checkpoint_segments** is increased, you need to increase the value of this parameter. The increase of these two parameters further requires the increase of **shared_buffers**. Consider all these parameters during setting.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

checkpoint_completion_target

Parameter description: Specifies the ratio of the estimated completion time of each checkpoint to the value of **checkpoint_timeout**.

Parameter type: floating-point number

Unit: none

Value range: 0.0 to 1.0

Default value: **0.5**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

checkpoint_warning

Parameter description: If the checkpoint interval is less than the value of this parameter because the checkpoint file is written, a warning message is written, indicating that the value of **checkpoint_segments** needs to be increased.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647. **0** indicates that the warning is disabled.

Default value: **300** (5 minutes)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

checkpoint_wait_timeout

Parameter description: Sets the longest time that the checkpoint waits for the checkpointer thread to start.

Parameter type: integer.

Unit: seconds

Value range: 2 to 3600

Default value: **60** (1 minute)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_incremental_checkpoint

Parameter description: Specifies whether to enable incremental checkpoint.

Parameter type: Boolean.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: The incremental checkpoint function is enabled.
- **off**: The incremental checkpoint function is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised to disable this function when the memory is small or the disk performance is poor. In other scenarios, you are advised to enable this function by default.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

When **enable_incremental_checkpoint** is disabled and the checkpoint is manually executed, **full_page_writes** is used to generate the first Xlog. When backup and restoration are triggered, the log containing "xlog contains no data" may be generated.

enable_double_write

Parameter description: Specifies whether to enable dual-write feature. When the incremental checkpoint function **enable_incremental_checkpoint** is enabled and **enable_double_write** is enabled, the **enable_double_write** dual-write feature is used for protection and **full_page_writes** is not used to prevent partial page writes.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The dual-write function is enabled.
- **off**: The dual-write function is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter to **on** in incremental checkpoint mode.

Risks and impacts of improper settings: Partial page writes may occur when this parameter is disabled and the incremental checkpoint function is enabled. As a result, persistent data is unavailable.

incremental_checkpoint_timeout

Parameter description: Specifies the maximum interval between automatic checkpoints when the incremental checkpoint (**enable_incremental_checkpoint**) is enabled.

Parameter type: integer.

Unit: seconds

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 1 to 3600

Default value: **60**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_xlog_prune

Parameter description: Specifies whether the primary node recycles logs if the size of Xlogs exceeds the value of **max_size_for_xlog_prune** when any standby node is disconnected.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The primary node recycles logs when any standby node is disconnected.
- **off**: The primary node does not recycle logs when any standby node is disconnected.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_size_for_xlog_prune

Parameter description: Specifies the amount of Xlogs that are forcibly recycled.

Scenario 1: Maximum number of Xlogs retained on the primary node when the standby node is faulty. This parameter takes effect when **enable_xlog_prune** is enabled. The mechanism is as follows:

1. If all standby nodes specified by the replconninfo series GUC parameters are connected to the primary node, this parameter does not take effect.
2. This parameter takes effect when one or more standby nodes specified by the replconninfo series GUC parameters are disconnected to the primary node. If the number of Xlogs on the host is greater than the value of this parameter, the Xlogs will be forcibly recycled. The following situations are excluded:

   In synchronous commit mode (that is, the **synchronous_commit** parameter is not set to **local** or **off**), if there are standby nodes connecting to the primary node, the primary node retains the logs after the minimum log receiving location in the majority of standby nodes. In this case, the number of retained logs may be greater than the value of **max_size_for_xlog_prune**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

3. If a standby node is being built, this parameter does not take effect. All logs on the primary node are retained to prevent failures caused by log recycling during the build operation.

Scenario 2: When the PITR archive feature is enabled, the maximum number of archive Xlogs is retained. If archive Xlogs are not uploaded for a long time, the Xlogs retained locally are forcibly recycled when the size of the Xlogs exceeds the value of this parameter. In this scenario, you do not need to enable **enable_xlog_prune**.

Parameter type: integer.

Unit: KB

Value range: 0 to 2199023255551

Default value: **256GB**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. If the value of the parameter does not contain a unit, the default unit is KB. The unit can also be MB or GB.

Setting suggestion: Set this parameter based on the available disk space. If PITR archiving is not enabled, it is recommended that the maximum value be less than or equal to 256 GB.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

- If the value is too small, the standby node may be disconnected, or a rebuild process may be triggered because the standby node cannot continuously obtain logs from the primary node when reconnecting to the primary node after build.
- When the PITR archive function is enabled, setting this parameter to a small value may trigger forcible Xlog recycling, causing subsequent XBSA archiving failures.
- If the value is too large, disk space bloat may occur when the standby node is disconnected for a long time.

max_redo_log_size

Parameter description: For a standby DN, this parameter indicates the expected log volume between the latest log replay point and the latest checkpoint position. For a primary DN, this parameter indicates the expected log volume between the latest log insertion point and the latest checkpoint position. This parameter is used to specify the page refreshing speed of checkpoints.

Parameter type: integer.

Unit: KB

Value range: 163840 to 2147483647

Default value: **1048576**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value. It is recommended that this parameter be set to a small value if the RTO is concerned.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

checkpoint_lag_alarm_threshold

Parameter description: Specifies the threshold for triggering an alarm for checkpoint advancement lag at the recovery start point.

- For the primary instance, this threshold indicates the log volume difference between the log write position and the recovery start point.
- For the standby instance, this threshold indicates the log volume difference between the log replay position and the recovery start point.

Parameter type: integer.

Unit: GB

Value range: –1 to 2000

- **-1**: The threshold is automatically configured based on the number of CPU cores. (80 GB for fewer than 16 cores, 100 GB for 16 to 127 cores, and 128 GB for 128 or more cores.)
- **0**: The alarm for checkpoint advancement lag at the recovery start point is disabled.
- *N* (positive integer): The threshold is set to *N* GB.

Default value: **128** (192-core CPU/1536 GB memory, 128-core CPU/1024 GB memory); **100** (104-core CPU/1024 GB memory, 96-core CPU/1024 GB memory, 96-core CPU/768 GB memory, 80-core CPU/640 GB memory, 80-core CPU/512 GB memory, 72-core CPU/576 GB memory, 64-core CPU/512 GB memory, 60-core CPU/480 GB memory, 32-core CPU/256 GB memory, 16-core CPU/128 GB memory); **80** (8-core CPU/64 GB memory, 4-core CPU/32 GB memory, 4-core CPU/16 GB memory)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.6.3 Log Replay

recovery_time_target

Parameter description: Specifies the flow control threshold for the standby node to complete log writing and replay. If the expected replay completion time of the standby node is greater than the value of this parameter, the primary node triggers log flow control and reduces the rate at which the primary node sends logs to the standby node.

Parameter type: integer.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: seconds

Value range: 0 to 3600

- **0**: Log flow control is disabled.
- 1 to 3600: The standby node can write and replay logs within the period specified by the value, so that the standby node can quickly assume the primary role.

Default value: **60**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the performance of the primary node will be affected. For example, if this parameter is set to **10**, there is a delay in replaying logs on the standby node compared with receiving logs on the standby node and the maximum delay is 10s. If the delay exceeds 10s, the primary node cannot send logs to the standby node, which affects the performance of the primary node. In primary and standby scenarios, transactions on the primary node can be committed only after logs on the standby node are flushed to disks. In this case, transaction execution on the primary node is restricted, affecting performance. If the value is too large, the flow control does not take effect. After flow control is disabled, the RTO/RPO may increase.

recovery_max_workers

Parameter description: Specifies the maximum number of parallel replayer threads. If the value is greater than 1, the parallel replay mode is used.

NOTICE

- This parameter takes effect only when both **recovery_parse_workers** and **recovery_redo_workers** are set to **1**, that is, the ultimate RTO is disabled.
- This parameter does not take effect when **recovery_parse_workers** or **recovery_redo_workers** is greater than 1, that is, ultimate RTO is enabled.

Parameter type: integer.

Unit: none

Value range: 0 to 20

Default value:

- For the 8-core CPU/64 GB memory specification in hybrid deployment mode, the default value is **1**.
- For all specifications in independent deployment mode and other specifications in hybrid deployment mode, the default value is **4**.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this parameter is set to a large value, the CPU and memory resources will be insufficient, reducing the replay speed.

recovery_parallelism

Parameter description: Queries the actual number of parallel replayer threads.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483647

Default value: **4**

Setting method: This is a POSTMASTER parameter and cannot be modified. If any of the **recovery_max_workers**, **recovery_parse_workers**, and **recovery_redo_workers** parameters is modified, **recover_parallelism** is recalculated.

Setting suggestion: This parameter is read-only and cannot be modified.

Risks and impacts of improper settings: none

queue_item_size

Parameter description: Specifies the maximum length of the task queue of each redo replayer thread.

Parameter type: integer.

Unit: none

Value range: 1 to 65535

Default value: **560**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, a large number of system resources will be occupied.

enable_parallel_reader

Parameter description: Specifies whether to enable the new ultimate RTO framework.

- This parameter must be used together with **recovery_readers**, **recovery_redo_workers**, **recovery_parse_workers** and **recovery_undo_workers**.
- Before enabling the new ultimate RTO framework, ensure that the value of **replication_type** is **1**.
- For details about how to enable the new ultimate RTO framework, see **Table 6-19**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-19 Enabling the new ultimate RTO framework

Parameter/ Mode	Parallel Replay	Ultimate RTO	New Ultimate RTO Framework
recovery_parse_workers	1	recovery_parse_workers>1 Or, recovery_redo_workers>1	recovery_parse_workers>1 Or, recovery_redo_workers>1
recovery_redo_workers	1		
enable_parallel_reader	off **NOTE** If **enable_parallel_reader** is set to **on**, the system automatically changes the value to **off** in the memory. In this case, this parameter does not take effect.	off	on

Parameter type: Boolean

Unit: none

Value range:

- **on**: The new ultimate RTO framework is enabled.
- **off**: The new ultimate RTO framework is disabled.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised not to enable this function if the number of CPU cores is less than 256.

Risks and impacts of improper settings: If the number of CPU cores is less than 256, enabling the new ultimate RTO framework may cause the cluster to fail to start.

recovery_readers

Parameter description: Specifies the number of readpage threads in the new ultimate RTO framework.

- **recovery_readers** must be used together with **enable_parallel_reader**, **recovery_redo_workers**, **recovery_parse_workers** and **recovery_undo_workers**. If you do not want to enable the new ultimate RTO framework, set **recovery_readers** to **1**.
- If **enable_parallel_reader** is set to **on** and the ultimate RTO is enabled, the value of **recovery_readers** must be greater than 1. If **enable_parallel_reader**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

is set to **on**, **recovery_readers** is set to **1**, and the value of **recovery_parse_workers** or **recovery_redo_workers** is greater than 1, the system changes the value of **enable_parallel_reader** to **false** in the memory.

Parameter type: integer

Unit: none

Value range: 1 to 50

Default value: **1**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: For details about the values of **recovery_readers**, **recovery_undo_workers**, **recovery_parse_workers**, and **recovery_redo_workers** in different CPU, memory, and deployment modes, see **Table 6-21**.

Risks and impacts of improper settings: The readpage thread has its own item for parsing Xlogs. Each time a readpage thread is added, the corresponding memory consumption increases. As services increase, if items are insufficient, item initialization will continuously happen, causing memory usage to increase. The number of items is limited. When the number of items reaches the upper limit, the memory usage does not increase significantly. However, due to operations such as ultra-large transactions may cause memory bloat of an item. Therefore, the memory usage may still increase after the number of items reaches the upper limit. Setting this parameter to a large value may cause high CPU and memory usage, which can reduce the replay speed or cause the replay to be suspended.

The new ultimate RTO framework has memory requirements. During startup, the system checks the configured values of **max_process_memory** and **shared_buffer** and calculates the maximum number of readers supported by this parameter. If the configured value of **recovery_readers** is greater than the calculated number of readers, the value of **recovery_readers** in the memory will be changed to the calculated number of readers.

parallel_max_item_count

Parameter description: Specifies the maximum number of redo items that can be used by a single readpage thread in the new ultimate RTO framework. This parameter takes effect only when the new ultimate RTO framework is used, that is, when **enable_parallel_reader** is set to **on**.

Parameter type: integer.

Unit: none

Value range: 40000 to 400000

Default value: 60000

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter by referring to **Table 6-21**.

Risks and impacts of improper settings: The readpage thread has its own item for parsing Xlogs. Each time a readpage thread is added, the corresponding

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

memory consumption increases. As services increase, if items are insufficient, item initialization will continuously happen, causing memory usage to increase. The number of items is limited. When the number of items reaches the upper limit, the memory usage does not increase significantly. However, due to operations such as ultra-large transactions may cause memory bloat of an item. Therefore, the memory usage may still increase after the number of items reaches the upper limit. If the value is too large, CPU and memory resources will be insufficient, causing the replay performance to deteriorate. If the value is too small, the replay performance will be affected. You are advised not to set this parameter to a value less than 100000. If the replay performance is affected due to a small value, increase the value based on the actual machine configuration.

The new ultimate RTO framework has memory requirements. During startup, the system checks the configured values of **max_process_memory** and **shared_buffer** and calculates the maximum number of readers supported by this parameter. If the configured value of **recovery_readers** is greater than the calculated number of readers, the value of **recovery_readers** in the memory will be changed to the calculated number of readers.

recovery_undo_workers

Parameter description: Specifies the number of undo threads in the new ultimate RTO framework.

recovery_undo_workers must be used together with **enable_parallel_reader**, **recovery_redo_workers**, **recovery_parse_workers** and **recovery_readers**. If you do not want to enable the new ultimate RTO framework, set **recovery_undo_workers** to **1**.

Parameter type: integer

Unit: none

Value range: 1 to 50

Default value: **1**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: For details about the values of **recovery_readers**, **recovery_undo_workers**, **recovery_parse_workers**, and **recovery_redo_workers** in different CPU, memory, and deployment modes, see **Table 6-21**.

Risks and impacts of improper settings: If this parameter is set to a large value, the CPU and memory resources will be insufficient, reducing the replay speed.

recovery_parse_workers

Parameter description: Specifies the number of ParseRedoRecord threads in the ultimate RTO feature.

- It must be used together with **recovery_redo_workers**. If the value of **recovery_parse_workers** or **recovery_redo_workers** is greater than 1, ultimate RTO is enabled. If you do not want to enable ultimate RTO, set **recovery_parse_workers** to **1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Ensure that the value of the parameter **replication_type** is set to **1** when ultimate RTO is enabled.
- If both ultimate RTO and parallel replay are enabled, the ultimate RTO feature takes effect but the parallel replay feature does not take effect.

Parameter type: integer.

Unit: none

Value range: 1 to 16

Default value:

- Independent deployment:

  2 (128-core CPU/1024 GB memory, 60-core CPU/480 GB memory, 32-core CPU/256 GB memory, 16-core CPU/128 GB memory); **1** (8-core CPU/64 GB memory, 4-core CPU/32 GB memory, 4-core CPU/16 GB memory)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set **recovery_parse_workers** and **recovery_redo_workers** based on the CPU, memory, and deployment mode by referring to **Table 6-20**.

Risks and impacts of improper settings: If this parameter is set to a large value, the CPU and memory resources will be insufficient, reducing the replay speed.

NOTE

- After ultimate RTO is enabled, the total number of extra replayer threads started by the standby node = **recovery_parse_workers** × (**recovery_redo_workers** + 2) + 5. More replayer threads occupy more CPU, memory, and I/O resources. Set parameters based on the actual hardware configuration. If the parameter value is too large, the CPU and memory usage may be too high, causing system startup exceptions. In hybrid deployment scenarios, host performance may be affected.
- In the new ultimate RTO framework, the value of **recovery_parse_workers** ranges from 2 to 16. In this mode, if **recovery_parse_workers** is set to **1**, the system automatically sets it to **2** in the memory.

recovery_redo_workers

Parameter description: Specifies the number of PageRedoWorker threads corresponding to each ParseRedoRecord thread in the ultimate RTO feature. **recovery_redo_workers** must be used together with **recovery_parse_workers**. If **recovery_parse_workers** or **recovery_redo_workers** is greater than 1, ultimate RTO is enabled. If you do not want to enable ultimate RTO, retain the default value **1** of **recovery_redo_workers**.

Parameter type: integer.

Unit: none

Value range:

- Ultimate RTO: 1 to 8
- New ultimate RTO framework: 1 to 50

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

In ultimate RTO mode, if the value is greater than 8, the value is automatically changed to **8** in the memory.

Default value:

- Independent deployment:

  8 (128-core CPU/1024 GB memory, 60-core CPU/480 GB memory, 32-core CPU/256 GB memory); **3** (16-core CPU/128 GB memory, 8-core CPU/64 GB memory); **1** (4-core CPU/32 GB memory, 4-core CPU/16 GB memory)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set **recovery_parse_workers** and **recovery_redo_workers** based on the CPU, memory, and deployment mode by referring to **Table 6-20** and **Table 6-21**.

Table 6-20 Parameter settings for different CPUs, memory sizes, and deployment modes

No.	Number of CPUs	Memory (GB)	Hybrid Deployment or Not	recovery_parse_workers	recovery_redo_workers	Number of Replayer Threads	Remarks
1	4	-	-	1	1	Value of **recovery_max_workers**	Ultimate RTO is not recommended.
2	8	-	Yes	1	1	Value of **recovery_max_workers**	Ultimate RTO is not recommended.
3	8	64	No	1	3	10	-
4	16	128	Yes	1	2	9	-
5	16	128	No	2	3	15	-
6	32	256	Yes	2	2	13	-
7	32	256	No	2	8	25	-
8	64	512	Yes	2	4	17	-
9	64	512	No	2	8	25	Systems with larger hardware specifications should also use this recommended value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Number of CPUs	Memory (GB)	Hybrid Deployment or Not	recovery_parse_workers	recovery_redo_workers	Number of Replayer Threads	Remarks
10	96	768	-	2	8	25	Systems with larger hardware specifications should also use this recommended value.

Table 6-21 Parameter settings for different CPUs, memory sizes, and deployment modes in the new ultimate RTO framework

No.	Number of CPUs	Memory (GB)	Hybrid Deployment or Not	recovery_readers	recovery_undo_workers	recovery_parse_workers	recovery_redo_workers	parall_max_item_count	Number of Replayer Threads	Remarks
1	<256	-	-	-	-	-	-	-	-	The new ultimate RTO framework is not recommended.
2	256	2048	No	40	30	3	50	400000	174	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When the new ultimate RTO framework is enabled, you are advised to set the following parameters:
  - redo_bind_cpu_attr='cpuorderbind:0-255'
  - standby_recycle_interval=1
  - base_page_saved_interval=2000
  - pagewriter_sleep=1000
  - pagewriter_thread_num=6
  - max_io_capacity=3GB
- In the new ultimate RTO framework, if the replay speed decreases, run the following command to check the buffer status:
  ```
  select * from dbe_perf.global_candidate_status;
  ```
  If the value in the **candidate_slot** column is small, there are few clean pages in the buffer. If the value in the **get_buf_clock_sweep** column is large, worker threads are evicting pages by themselves. In this case, you can increase the value of **shared_buffer** to increase the number of buffers or improve the disk performance to accelerate the replay speed.

Risks and impacts of improper settings: If this parameter is set to a large value, the CPU and memory resources will be insufficient, reducing the replay speed.

enable_page_lsn_check

Parameter description: Specifies whether to enable the data page LSN check. During replay, the current LSN of the data page is checked to see if it is the expected one.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The data page LSN check function is enabled.
- **off**: The data page LSN check function is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, data page errors may not be quickly detected, resulting in unexpected errors.

promote_lsn_check

Parameter description: Specifies whether to verify the integrity of local log replay when the standby node is promoted to primary after replay.

Parameter type: Boolean.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: The local log replay verification mechanism is enabled.
- **off**: The local log replay verification mechanism is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, promoting to primary in advance due to log faults may not be intercepted, causing data loss risks.

recovery_min_apply_delay

Parameter description: Specifies the replay delay of the standby node.

Parameter type: integer.

Unit: milliseconds

Value range: 0 to 2147483647

Default value: **0** (no delay added)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: See the notice below.

Risks and impacts of improper settings: See the notice below.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- This parameter does not take effect on the primary node. It must be set on the standby node that requires a delay. You are advised to set this parameter on the asynchronous standby node. If the delay is set on the asynchronous standby node, the RTO will take a long time after the node is promoted to primary.
- The delay time is calculated based on the transaction commit timestamp on the primary node and the current time on the standby node. Therefore, ensure that the clocks of the primary and standby nodes are synchronized.
- If the delay time is too long, the disk where the Xlog file is located on the standby node may be full. Therefore, you need to set the delay time based on the disk size.
- Operations without transactions are not delayed.
- After the primary/standby switchover, if the original primary node needs to be delayed, you need to manually set this parameter.
- When **synchronous_commit** is set to **remote_apply**, synchronous replication is affected by the delay. Each commit message is returned only after the replay on the standby node is complete.
- **hot_standby_feedback** will be delayed due to using this feature, which may cause the primary node to expand. Therefore, you need to balance the risk when using both parameters.
- After a DDL operation (such as DROP or TRUNCATE) that holds an AccessExclusive lock is performed on an object on the primary node, if the object is queried on the standby node during the delayed replay of the record on the standby node, it will be returned only after the lock is released.

dcf_truncate_dump_info_level

Parameter description: Specifies whether to print the LSN truncated by the DCF and the subsequent LSNs.

Parameter type: integer.

Unit: none

Value range: 0 to 2

- **0**: disabled.
- **1**: prints all LSNs truncated by the DCF (Xlogs whose LSN is greater than or equal to the truncated LSN).
- **2**: prints all LSNs truncated by the DCF and prints warning-level logs when the LSNs flushed to disks are greater than the truncated LSNs.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

redo_bind_cpu_attr

Parameter description: Specifies the core binding operation of the replayer thread. Only the SYSADMIN user can access this parameter.

Parameter type: string.

Unit: none

Value range: a string of more than 0 characters. The value is case-insensitive.

- **"autobind"**: The log replayer thread is automatically bound to a NUMA group. By default, the log replayer thread is bound to the last NUMA group. This parameter is valid only for Arm servers. Cores are not bound for x86 servers and small-specification servers.
- **"nobind"**: The thread is not bound to a core.
- **"nodebind: 1, 2"**: Use the CPU cores in NUMA groups 1 and 2 to bind threads.
- **"cpubind: 0-30"**: Use the CPU cores 0 to 30 to bind threads.
- **"cpuorderbind: 16-32"**: One thread is bound to one CPU core starting from core 16. If the number of cores in the range is insufficient, the remaining threads are not bound. You are advised to set the interval to a value greater than or equal to the value of **recovery_parallelism** plus 1.

Default value: **"autobind"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: See the note below.

Risks and impacts of improper settings: The database node cannot be started.

NOTE

- This parameter is used for core binding in the Arm environment. You are advised to bind all replayer threads to the same NUMA group for better performance. In hybrid deployment scenarios, you are advised to bind the replayer threads of different nodes on the same host to different NUMA groups. If the parameters related to NUMA core binding are used in a non-Arm environment, the cluster will fail to be started.
- The core binding range specified by this parameter must be different from the core binding range specified by the **thread_pool_attr** parameter.

enable_wal_page_header_crc

Parameter description: Specifies whether to perform CRC calculation, value assignment, and verification on the WAL page header.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: CRC calculation, value assignment, and verification are enabled on the WAL page header.
- **off**: CRC calculation, value assignment, and verification are disabled on the WAL page header.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the system may fail to check whether the WAL page header is correct. As a result, the standby node receives incorrect WALs and fails to replay and synchronize the WALs.

6.3.6.4 Archiving

archive_interval

Parameter description: Specifies the archiving interval. Log files are forcibly archived when the period specified by this parameter has elapsed.

Parameter type: integer.

Unit: seconds

Value range: 1 to 1000

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, data will be archived frequently, occupying storage space. If the value is too large, the RPO of PITR will be affected.

time_to_target_rpo

Parameter description: In dual-cluster remote DR mode, logs of the primary cluster are archived to OBS. Specifies the time allowed from the time when the primary cluster is abnormal to the time when the primary cluster is archived to the recovery point of OBS.

Parameter type: integer.

Unit: seconds

Value range: 0 to 3600

- **0** indicates that log flow control is disabled.
- A value from 1 to 3600 indicates the duration from the time when the primary cluster is abnormal to the time when the primary cluster is archived to the recovery point of OBS. This ensures that the maximum data loss duration is within the allowed range when the primary cluster breaks down due to a disaster.

Default value: **10**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the host performance is affected. If the value is too large, the flow control does not take effect.

6.3.7 HA Replication

6.3.7.1 Sending Server

max_wal_senders

Parameter description: Specifies the maximum number of concurrent connections of the Xlog sender. The value must be smaller than that of **max_connections**.

NOTICE

The prerequisite for setting **max_wal_senders** is as follows: **wal_level** must be set to **archive**, **hot_standby**, or **logical** to allow the connection to the standby node.

Parameter type: integer.

Unit: none

Value range: 0 to 1024. The recommended value range is 8 to 100.

Default value: **20**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- This parameter can be set to **0** only when a single DN is used and there is no primary/standby instance.
- When HA replication, backup and restoration, and logical decoding are used, you are advised to set this parameter to: Number of current standby nodes + Number of backup connections + Number of required logical replication connections.

Risks and impacts of improper settings: If this parameter is set to a value less than the recommended value range, these functions may be unavailable or abnormal.

wal_keep_segments

Parameter description: Specifies the minimum number of Xlog files that can be retained in the **pg_xlog** directory. The standby node obtains the Xlogs from the primary node to perform streaming replication.

Parameter type: integer.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 2 to 2147483647

Default value: **128**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- During WAL archiving or recovery from a checkpoint on the server, the system may retain more log files than the number specified by **wal_keep_segments**.
- If the HA system uses asynchronous transmission, increase the value of **wal_keep_segments** when data greater than 4 GB is continuously imported in COPY mode. Take T6000 board as an example. If the data to be imported reaches 50 GB, you are advised to set this parameter to **1000**. You can dynamically restore the setting of this parameter after data import is complete and the log synchronization is normal.
- If the **synchronous_commit** level is lower than LOCAL_FLUSH, you are advised to set this parameter to **1000** when rebuilding the standby node to prevent rebuilding failures caused by primary node log recycling during the rebuilding.

Risks and impacts of improper settings: If the value is too large, the **pg_xlog** folder may occupy too much disk space. If the value is too small, the transaction log may have been overwritten by a new one before requested by the standby node. As a result, the request fails and the connection between the primary and standby nodes is terminated.

wal_sender_timeout

Parameter description: Specifies the maximum duration that the node waits for the receiver to receive transaction logs.

NOTICE

- If the data volume on the primary node is huge, the value of this parameter must be increased for rebuilding. For example, if the data volume on the primary node reaches 500 GB, you are advised to set this parameter to **600s**.
- This parameter cannot be set to a value larger than the value of **wal_receiver_timeout** or the timeout parameter for database rebuilding.

Parameter type: integer.

Unit: milliseconds

Value range: 0 to 2147483647

Default value: **6000**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the system is more sensitive to network fluctuation and intermittent disconnection. As a result,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

the connection between the primary and standby nodes may be abnormal. If the value is too large, the response to exceptions is slower.

max_replication_slots

Parameter description: Specifies the maximum number of log replication slots on the primary node.

Parameter type: integer.

Unit: none

Value range: 0 to 1024. The recommended value range is 8 to 100.

Default value: **20**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

When HA replication, backup and restoration, and logical decoding are used, you are advised to set this parameter to a value by referring to the following formula: Number of current physical streaming replication slots + Number of backup slots + Number of required logical replication slots. If the actual value is smaller than the recommended value, these functions may be unavailable or abnormal.

- Physical streaming replication slots provide an automatic method to ensure that Xlogs are not removed from a primary DN before they are received by all the standby DNs. Therefore, physical streaming replication slots are used to support the cluster HA. The number of physical streaming replication slots required by the cluster is as follows: ratio of the number of standby DNs to the number of primary DNs in a ring of DNs. If the HA cluster has 1 primary DN and 3 standby DNs, the number of physical streaming replication slots required is 3.
- Backup slot records replication information during backup execution. Full backup and incremental backup correspond to two independent backup slots.
- Plan the number of logical replication slots as follows:
  - A logical replication slot can carry changes of only one database for decoding. If multiple databases are involved, create multiple logical replication slots.
  - If logical replication is needed by multiple target databases, create multiple logical replication slots in the source database. Each logical replication slot corresponds to one logical replication link.
  - A maximum of 20 logical replication slots can be enabled for decoding on the same instance.

Risks and impacts of improper settings: If the value is too small, the standby node cannot be connected, or logical decoding and backup archiving cannot be enabled. If the value is too large, the performance will be affected. When establishing a DR relationship, ensure that **max_replication_slots** is set to the same value on both the primary and standby instances. Otherwise, the DR relationship may fail to be established.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_wal_shipping_compression

Parameter description: Specifies whether to enable cross-cluster log compression in streaming DR mode.

NOTICE

This parameter applies only to a pair of WAL senders and WAL receivers for cross-cluster transmission in streaming DR and is configured in the primary cluster.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Cross-cluster log compression is enabled in streaming DR mode.
- **off**: Cross-cluster log compression is disabled in streaming DR mode.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You do not need to set this parameter separately. This function is enabled by default during DR setup.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

repl_auth_mode

Parameter description: Specifies the validation mode for primary/standby replication and standby node rebuilding.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- If UUID validation is enabled on the primary node and a non-null repl_uuid validation code is configured, UUID validation must also be enabled on the standby node and the same repl_uuid validation code must be configured on the standby node. Otherwise, requests for log replication between the primary and standby nodes and standby node rebuilding will be rejected by the primary node.
- The SIGHUP parameter can dynamically load new values. The modification does not affect the established primary/standby connection and takes effect for subsequent primary/standby replication requests and primary/standby rebuilding requests.
- It supports the standby node rebuild validation under the Quorum and DCF protocols and the primary/standby replication validation under the Quorum protocol. It does not support primary/standby replication validation under the DCF protocol.
- It does not support the authentication between the primary and standby nodes across clusters, including the primary/standby OceanStor Dorado and DR clusters.
- The UUID validation function is used to prevent data crosstalk and pollution caused by incorrect connection between the primary and standby nodes. It is not used for security purposes.
- This parameter cannot be automatically synchronized between the primary and standby nodes.

Parameter type: enumerated type.

Unit: none

Value range:

- **off**: UUID validation is disabled.
- **default**: indicates that UUID validation is disabled.
- **uuid**: indicates that UUID validation is enabled.

Default value: **default**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Determine whether to enable this function based on the restrictions in the notice and requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

repl_uuid

Parameter description: Specifies the UUID used for primary/standby UUID validation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- If UUID validation is enabled on the primary node and a non-null repl_uuid validation code is configured, UUID validation must also be enabled on the standby node and the same repl_uuid validation code must be configured on the standby node. Otherwise, requests for log replication between the primary and standby nodes and standby node rebuilding will be rejected by the primary node.
- The SIGHUP parameter can dynamically load new values. The modification does not affect the established primary/standby connection and takes effect for subsequent primary/standby replication requests and primary/standby rebuilding requests.
- It supports the standby node rebuild validation under the Quorum and DCF protocols and the primary/standby replication validation under the Quorum protocol. It does not support primary/standby replication validation under the DCF protocol.
- It does not support the authentication between the primary and standby nodes across clusters, including the primary/standby OceanStor Dorado and DR clusters.
- The UUID validation function is used to prevent data crosstalk and pollution caused by incorrect connection between the primary and standby nodes. It is not used for security purposes.
- This parameter cannot be automatically synchronized between the primary and standby nodes.

Parameter type: string.

Unit: none

Value range: a string. The value contains 0 to 63 characters, including letters and digits. The value is case-insensitive and is converted to lowercase letters for storage. An empty string indicates that UUID validation is disabled.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Determine whether to enable this function based on the restrictions in the notice and requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

replconninfo1

Parameter description: Specifies the information about the first node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo1 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the first node is configured.

Default value: ""

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

replconninfo2

Parameter description: Specifies the information about the second node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo2 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the second node is configured.

Default value: ""

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

replconninfo3

Parameter description: Specifies the information about the third node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo3 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: a string. An empty string indicates that no information about the third node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

replconninfo4

Parameter description: Specifies the information about the fourth node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo4 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the fourth node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

replconninfo5

Parameter description: Specifies the information about the fifth node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo5 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the fifth node is configured.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: ""

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

replconninfo6

Parameter description: Specifies the information about the sixth node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo6 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the sixth node is configured.

Default value: ""

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

replconninfo7

Parameter description: Specifies the information about the seventh node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo7 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the seventh node is configured.

Default value: ""

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

replconninfo8

Parameter description: Specifies the information about the eighth node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo8 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the eighth node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

replconninfo9

Parameter description: Specifies the information about the ninth node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo9 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the ninth node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

replconninfo10

Parameter description: Specifies the information about the tenth node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo10 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the tenth node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

replconninfo11

Parameter description: Specifies the information about the eleventh node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo11 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the eleventh node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

replconninfo12

Parameter description: Specifies the information about the twelfth node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo12 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the twelfth node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

replconninfo13

Parameter description: Specifies the information about the thirteenth node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo13 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the thirteenth node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

replconninfo14

Parameter description: Specifies the information about the fourteenth node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo14 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the fourteenth node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

replconninfo15

Parameter description: Specifies the information about the fifteenth node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo15 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the fifteenth node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

replconninfo16

Parameter description: Specifies the information about the sixteenth node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo16 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the sixteenth node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

replconninfo17

Parameter description: Specifies the information about the seventeenth node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo17 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the seventeenth node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

replconninfo18

Parameter description: Specifies the information about the eighteenth node to be listened on and authenticated by the current server. This parameter is automatically configured after the cluster is successfully installed.

Example:

```
replconninfo18 = "localhost=XXXX localport=XXXX localheartbeatport=XXXX localservice=XXXX
remotehost=XXXX remoteport=XXXX remoteheartbeatport=XXXX remoteservice=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the eighteenth node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal cluster status.

cross_cluster_replconninfo1

Parameter description: Specifies the information about the first node to be listened on and authenticated by the current server across clusters.

Example:

```
cross_cluster_replconninfo1 = "localhost=XXXX localport=XXXX remotehost=XXXX remoteport=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the first node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal status of a standby cluster for DR.

cross_cluster_replconninfo2

Parameter description: Specifies the information about the second node to be listened on and authenticated by the current server across clusters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Example:

```
cross_cluster_replconninfo2 = "localhost=XXXX localport=XXXX remotehost=XXXX remoteport=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the second node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal status of a standby cluster for DR.

cross_cluster_replconninfo3

Parameter description: Specifies the information about the third node to be listened on and authenticated by the current server across clusters.

Example:

```
cross_cluster_replconninfo3 = "localhost=XXXX localport=XXXX remotehost=XXXX remoteport=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the third node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal status of a standby cluster for DR.

cross_cluster_replconninfo4

Parameter description: Specifies the information about the fourth node to be listened on and authenticated by the current server across clusters.

Example:

```
cross_cluster_replconninfo4 = "localhost=XXXX localport=XXXX remotehost=XXXX remoteport=XXXX"
```

Parameter type: string.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: a string. An empty string indicates that no information about the fourth node is configured.

Default value: ""

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal status of a standby cluster for DR.

cross_cluster_replconninfo5

Parameter description: Specifies the information about the fifth node to be listened on and authenticated by the current server across clusters.

Example:

```
cross_cluster_replconninfo5 = "localhost=XXXX localport=XXXX remotehost=XXXX remoteport=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the fifth node is configured.

Default value: ""

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal status of a standby cluster for DR.

cross_cluster_replconninfo6

Parameter description: Specifies the information about the sixth node to be listened on and authenticated by the current server across clusters.

Example:

```
cross_cluster_replconninfo6 = "localhost=XXXX localport=XXXX remotehost=XXXX remoteport=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the sixth node is configured.

Default value: ""

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal status of a standby cluster for DR.

cross_cluster_replconninfo7

Parameter description: Specifies the information about the seventh node to be listened on and authenticated by the current server across clusters.

Example:

```
cross_cluster_replconninfo7 = "localhost=XXXX localport=XXXX remotehost=XXXX remoteport=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the seventh node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal status of a standby cluster for DR.

cross_cluster_replconninfo8

Parameter description: Specifies the information about the eighth node to be listened on and authenticated by the current server across clusters.

Example:

```
cross_cluster_replconninfo8 = "localhost=XXXX localport=XXXX remotehost=XXXX remoteport=XXXX"
```

Parameter type: string.

Unit: none

Value range: a string. An empty string indicates that no information about the eighth node is configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during installation. You are advised not to change the value.

Risks and impacts of improper settings: The incorrect connection information may cause the abnormal status of a standby cluster for DR.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_time_report

Parameter description: Specifies whether to record the time consumed by each redo log.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The time consumed by each redo log is recorded.
- **off**: The time consumed by each redo log is not recorded.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised not to enable the non-debugging mode because the performance deteriorates after the non-debugging mode is enabled.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

thread_top_level

Parameter description: Increases the priority of the WALWRITERAUXILIARY, WALWRITER, STARTUP, WALRECEIVER, WAL_NORMAL_SENDER, and PGSTAT threads to the highest.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The priority of the preceding threads is increased to the highest.
- **off**: The priority of the preceding threads is not increased.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

page_work_queue_size

Parameter description: Specifies the length of the blocking queue of each redo worker.

Parameter type: integer.

Unit: none

Value range: 1 to 100000

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **4096**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the pipeline capability becomes invalid. If the value is too large, too much memory is occupied.

parallel_build_thread_num

Parameter description: Specifies the degree of parallelism for parallel reconstruction. When the value is **1**, the original single-thread rebuilding code is run. When the value is **n**, **n** threads are established to complete data transmission during rebuilding.

Parameter type: integer.

Unit: none

Value range: 1 to 10

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- Retain the default value. Set this parameter based on the service scenario and disk I/O capability only when a large number of data files need to be rebuilt, the disk I/O performance is good, and the network bandwidth, memory, and CPU resources are sufficient. Higher degree of parallelism indicates more I/O, memory, and CPU resources are occupied and faster rebuilding speed.
- The degree of parallelism is in direct proportion to the CPU and memory usage. (When there are two degrees of parallelism, the maximum CPU usage is three cores. Each time one degree of parallelism is added, the CPU usage increases by one core.)
- Assume that the build speed in a single thread is **a** MB/s. When **enable_parallel_build_compression** is set to **off**, the read I/O on the service side increases by no more than **a** MB/s each time a degree of parallelism is added. When **enable_parallel_build_compression** is set to **on**, the read I/O on the service side increases by no more than 0.7 x **a** MB/s each time a degree of parallelism is added.

Risks and impacts of improper settings:

If this function is enabled in an inappropriate environment, the parallel rebuild speed does not increase significantly and extra resources are occupied.

Each time a degree of parallelism is added, a connection is established with a new WAL Sender on the server, occupying the quota of the GUC parameter **max_wal_senders** on the server. If the degree of parallelism is too large, the parallel rebuilding fails because no connection can be established. Then, the single-thread rebuilding is performed again.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_parallel_build_compression

Parameter description: Specifies whether to perform LZ4 compression during parallel data rebuilding. This parameter is valid only when **parallel_build_thread_num** is greater than 1.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: LZ4 compression is performed during parallel data rebuilding.
- **off**: LZ4 compression is not performed during parallel data rebuilding.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

Retain the default value. When the network bandwidth is limited, set this parameter to **on**.

Risks and impacts of improper settings:

If this function is enabled in an inappropriate environment, the parallel rebuild speed does not increase significantly and extra resources are occupied.

replay_delay

Parameter description: Specifies the replay delay for standby clusters (also referred to as DR instances) when delayed replay is enabled in DR scenarios.

RTO indicates the time required for data restoration when a disaster occurs. The RTO is calculated as follows: (Target replay point – Current replay point)/Replay rate, that is, the ratio of the number of logs to be replayed to the replay rate. The delayed replay feature enables the standby cluster to become a snapshot of the primary cluster before the replay delay. When a failover is performed to promote the standby cluster to primary, logs generated after the target replay point (Current time minus **replay_delay**) will be truncated to ensure that the RTO of the failover is not affected by the **replay_delay** parameter.

Parameter type: integer.

Unit: seconds

Value range: 0 to 4294967295. The value **0** indicates that delayed replay is disabled for standby clusters.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter according to actual requirements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- The replay delay can be set from minutes to years. When the following time synchronization limitations are met, the replay delay can be accurate to seconds: NTP must be used to ensure that the time synchronization precision between the primary and standby clusters is less than or equal to 100 ms, and the time synchronization precision between nodes in the standby cluster is less than or equal to 1 ms.
- After the standby cluster for DR is promoted to primary through a failover, **replay_delay** is reset to the default value **0**.

Risks and impacts of improper settings:

- If the value is too large, the standby cluster may not replay logs for a long time. As a result, logs are stacked and the disk space is used up.
- If the value is not 0, the standby cluster cannot perform switchover, DR simulation, or DR interruption and resumption.

replay_mode

Parameter description: Specifies whether to pause replay on the standby cluster when delayed replay is enabled in DR scenarios.

Parameter type: string.

Unit: none

Value range:

- **""**: Replay pause is disabled.
- **stop**: Replay is paused immediately.
- Timestamp: If this parameter is set to a timestamp, the replay is paused at the barrier point corresponding to the timestamp.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the timestamp of the barrier point.

NOTICE

After the standby cluster for DR is promoted to primary through a failover, **replay_mode** is reset to the default value **""**.

Risks and impacts of improper settings: If this parameter is set to the timestamp of a barrier that has been replayed or does not exist, the replay process is suspended.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

hadr_replconninfo_config_mode

Parameter description: Specifies the management mode of the DR link.

Parameter type: enumerated type.

Unit: none

Value range:

- **local_file**: The DR link management mode is based on GUC configuration.
- **cluster_manager**: The DR link management mode is based on cluster management configuration.

Default value: **local_file**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set during the DR process. You are advised not to change the value.

Risks and impacts of improper settings: The DR link attribute setting becomes invalid.

hadr_source_info

Parameter description: Specifies the link information of the cluster at the log sender for the cluster at the log receiver in the streaming DR scenario.

Parameter type: string.

Unit: none

Value range: a string. **""** indicates that the link information of the cluster at the log sender is not configured.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set in the DR process. You are advised not to change the value. The value of the **hadr_source_info** of the cluster at the log receiver must match **dest_name** configured for the cluster at the log sender.

Risks and impacts of improper settings: If **hadr_source_info** of the main standby node is not correctly configured, the cluster at the log receiver cannot connect to the cluster at the log sender.

6.3.7.2 Primary Node

synchronous_standby_names

Parameter description: Specifies a comma-separated list of names of potential standby nodes that support synchronous replication.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- The current synchronous standby node is on the top of the list. If the current synchronous standby node is disconnected, it will be replaced immediately with the next-highest-priority standby node. Name of the next-highest-priority standby node is added to the list.
- The standby node name can be specified by setting the environment variable **PGAPPNAME**.

Parameter type: string.

Unit: none

Value range: If this parameter is set to *****, the name of any standby node that provides synchronous replication is matched. The value can be configured in the following format:

- ANY *num_sync* (*standby_name* [, ...])
- [FIRST] *num_sync* (*standby_name* [, ...])
- *standby_name* [, ...]

NOTE

- In the preceding command, *num_sync* indicates the number of standby nodes that need to wait for responses from the transaction, *standby_name* indicates the name of the standby node, and FIRST and ANY specify the policies for selecting standby nodes for synchronous replication from the listed servers.
- **ANY N (dn_instanceId1, dn_instanceId2,...)** indicates that any *N* host names in the brackets are selected as the name list of standby nodes for synchronous replication. For example, **ANY 1(dn_instanceId1, dn_instanceId2)** indicates that either of **dn_instanceId1** or **dn_instanceId2** is used as the standby node for synchronous replication.
- **FIRST N (dn_instanceId1, dn_instanceId2, ...)** indicates that the first N primary node names in the brackets are selected as the standby node name list for synchronous replication based on the priority. For example, **FIRST 1 (dn_instanceId1, dn_instanceId2)** indicates that **dn_instanceId1** is selected as the standby node for synchronous replication.
- The meanings of **dn_instanceId1, dn_instanceId2, ...** are the same as those of **FIRST 1 (dn_instanceId1, dn_instanceId2, ...)**.

If you use the gs_guc tool to set this parameter, perform the following operations:

```
gs_guc reload -Z datanode -N @NODE_NAME@ -D @DN_PATH@ -c "synchronous_standby_names='ANY 1(dn_instanceId1, dn_instanceId2)'"
```

or

```
gs_guc reload -Z datanode -N @NODE_NAME@ -D @DN_PATH@ -c "synchronous_standby_names='ANY 1(AZ1, AZ2)'"
```

Default value: *****

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If the synchronization standby list is incorrect, the primary and standby synchronization majority protocol becomes invalid.

NOTICE

This parameter is maintained by the CM. If it is manually modified, data may be lost.

most_available_sync

Parameter description: Specifies whether to block the transaction commit on the primary node when the primary-standby synchronization fails.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The transaction commit on the primary node is not blocked when the synchronization fails.
- **off**: The transaction commit on the primary node is blocked when the synchronization fails.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: After this function is enabled, the primary node does not wait for the standby node to commit for verification. If the host is in the single-primary state, the committing is successful, not the majority write is successful.

Risks and impacts of improper settings: Customers need to exercise caution when enabling this function based on their application scenarios.

enable_stream_replication

Parameter description: Specifies whether data and logs are synchronized between the primary and standby nodes.

NOTICE

- This parameter is used for testing performance with standby DNs and without standby DNs. If this parameter is set to **off**, tests on abnormal scenarios, such as switchover and faults, cannot be performed to prevent inconsistency between the primary and standby nodes.
- This is a restricted parameter, and you are advised not to set it to **off** in normal service scenarios.

Parameter type: Boolean.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range:

- **on**: Synchronization between the primary and standby nodes is enabled.
- **off**: Synchronization between the primary and standby nodes is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the multi-copy capability of the cluster will be unavailable, reducing availability and causing backup failures. Therefore, you are advised not to disable this parameter.

enable_mix_replication

Parameter description: Specifies how WAL files and data are replicated between the primary and standby nodes.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The WAL file and data page mixed replication mode is enabled.
- **off**: The WAL file and data page mixed replication mode is disabled.

Default value: **off**

Setting method: This parameter is a POSTMASTER parameter. Its default value is **off** and cannot be modified.

Setting suggestion: Retain the default value. This parameter cannot be modified in normal service scenarios. That is, the WAL file and data page mixed replication mode is disabled by default.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_wal_replication_compression

Parameter description: Specifies whether to enable the Xlog compression function during physical replication between the primary and standby DNs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The log compression function is enabled.
- **off**: The log compression function is disabled.

Default value: **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter applies only to a pair of WAL Sender and WAL Receiver threads for data transmission between the primary and standby DNs. Configure this parameter on the primary DN. You are advised not to enable this parameter in scenarios with small Xlogs and delay sensitivity.

Risks and impacts of improper settings: After this parameter is enabled, performance and resources are consumed. Therefore, you need to set **wal_replication_compression_thread_settings** and **wal_replication_compression_settings** properly.

- Impact on performance: For details, see the description of the **wal_replication_compression_settings** parameter.
- Configurations in different resource scenarios: For details, see the description of the **wal_replication_compression_thread_settings** parameter.

NOTICE

This parameter has the following limitations:

- The compression algorithm cannot be modified during the upgrade.
- DCF is not supported.
- Currently, standby clusters for DR are not supported.
- You are advised not to enable this function in small specifications (for example, 4 or 8 cores).

wal_replication_compression_thread_settings

Parameter description: Specifies the number of parallel compression threads and single-thread queue depth of the primary and standby clusters. This parameter takes effect only when **enable_wal_replication_compression** is enabled.

Parameter type: string.

Unit: none

Value range: a string of two integers separated by a comma (,). The value ranges of both integer parameters are 1 to 16.

Default value: **'1,6'**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter and **walsender_max_send_size** are used together to specify the memory usage of the compression thread. You need to use this parameter with caution. You are advised to set this parameter to a small value for low-performance machines.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Number of CPUs	Memory (GB)	Hybrid Deployment or Not	walsender_max_send_size (MB)	Maximum Number of Threads for Xlog Compression	Maximum Queue Depth of the Compression Buffer	wal_replication_compression_thread_settings	Remarks
1	4	-	-	8	0	0	-	Generally, Xlog compression is not recommended. If the network traffic is limited, you can set this parameter based on the hardware specifications to enable Xlog compression to relieve the traffic pressure.
2	8	-	Yes	8	0	0	-	
3	8	64	No	8	1	1	1,1	
4	16	128	Yes	8	2	2	2,2	
5	16	128	No	8	2	2	2,2	
6	32	256	Yes	8	4	2 to 4 (Determine the recommended value based on the optimal performance after the performance test.)	4,2~4,4	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Number of CPUs	Memory (GB)	Hybrid Deployment or Not	walsender_max_send_size (MB)	Maximum Number of Threads for Xlog Compression	Maximum Queue Depth of the Compression Buffer	wal_replication_compression_thread_settings	Remarks
7	32	256	No	8	4	2 to 4 (Determine the recommended value based on the optimal performance after the performance test.)	4,2~4,4	
8	64	512	Yes	8	8	2 to 4 (Determine the recommended value based on the optimal performance after the performance test.)	8,2~8,4	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Number of CPUs	Memory (GB)	Hybrid Deployment or Not	walsender_max_send_size (MB)	Maximum Number of Threads for Xlog Compression	Maximum Queue Depth of the Compression Buffer	wal_replication_compression_thread_settings	Remarks
9	64	512	No	8	8	2 to 4 (Determine the recommended value based on the optimal performance after the performance test.)	8,2~8,4	
10	96	768	-	8	12	2 to 4 (Determine the recommended value based on the optimal performance after the performance test.)	12,2~12,4	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this parameter is set to a large value, a large amount of dynamic memory is occupied. As a result, the process cannot be started or the service fails to apply for memory.

wal_replication_compression_settings

Parameter description: Specifies the algorithm rule used for Xlog compression of the primary and standby clusters.

Parameter type: string.

Unit: none

Value range: The parameter value consists of the compression algorithm and compression parameter. The format is '*string*,*int*', for example, **'zstd,5'**. Other formats are incorrect, which will cause the setting failure. The first parameter (compression algorithm) can be set to **lz4** or **zstd**. Other values are invalid.

- If the first parameter is set to **lz4**, the second parameter is the acceleration factor and the value ranges from 1 to 300.
- If the first parameter is set to **zstd**, the second parameter is the compression level and the value ranges from 1 to 15.

Default value: **'lz4,3'**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- A larger acceleration factor of the LZ4 algorithm indicates a lower compression ratio and better performance.
- A higher compression level of the zstd algorithm indicates a higher compression ratio and poorer performance.
- Increasing the compression ratio will reduce the network bandwidth, but the TPS performance will also deteriorate. You are advised to set the compression ratio to the minimum compression ratio that meets the bandwidth requirements.
- The zstd algorithm has a higher compression ratio and more performance degradation, while the lz4 algorithm has a lower compression ratio and less performance degradation.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

data_replicate_buffer_size

Parameter description: Specifies the amount of memory used by queues when the sender sends data pages to the receiver. The value of this parameter affects the buffer size used during the replication from the primary node to the standby node.

Parameter type: integer.

Unit: KB

Value range: 4096 to 1072693248

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **131072**, that is, 128 MB.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Do not set this parameter to an excessively large or small value. The following condition must be met: **data_replicate_buffer_size** + **segment_buffers** + **shared_buffers** + **wal_buffers** + **temp_buffers** + **maintenance_work_mem** + **work_mem** + **query_mem** + **wal_receiver_buffer_size** (standby node) < **max_process_memory** < Memory size of the physical machine.

Risks and impacts of improper settings: If the value of the memory parameter is too large and exceeds the upper limit of the physical memory, the database cannot be started because it cannot apply for sufficient memory.

walsender_max_send_size

Parameter description: Specifies the size of the log or data sending buffer on the primary node.

Parameter type: integer.

Unit: KB

Value range: 8 to 2147483647

Default value: **8192** (8 MB)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, too much memory is occupied. If the value is too small, the synchronization performance between the primary and standby nodes deteriorates.

wal_buffer_reserve_percent

Parameter description: Specifies the ratio of the WAL buffer reservation window on the primary node. The window retains the log data that has been flushed to disks in the buffer. If this parameter is set to a valid non-zero value, this parameter takes effect only when the value of **wal_buffers** is greater than or equal to 1 GB.

Parameter type: floating-point number

Unit: none

Value range: 0.0 to 0.5

Default value: **0.25**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

ha_module_debug

Parameter description: Specifies the replication status log of a specific data block during data replication.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The status of each data block is recorded in logs during data replication.
- **off**: The status of each data block is not recorded in logs during data replication.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Printing logs occupies unnecessary disk, CPU, and I/O bandwidth. Therefore, you are advised not to enable this function.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

sync_config_strategy

Parameter description: Specifies the policy for synchronizing configuration files between the primary node and standby node, and between the standby node and cascaded standby node.

NOTICE

If this parameter is configured after an upgrade from a version earlier than 503.0.0 to 503.0.0, this parameter cannot be identified after a rollback. Do not set this parameter during the upgrade to 503.0.0.

Parameter type: enumerated type.

Unit: none

Value range:

- **all_node**: If this parameter is set to **all_node** for the primary node, the primary node is allowed to proactively synchronize configuration files to all standby nodes. If this parameter is set to **all_node** for a standby node, the standby node is allowed to send synchronization requests to its primary node, and the standby node is allowed to proactively synchronize configuration files to all cascaded standby nodes. If this parameter is set to **all_node** for a cascaded standby node, the current cascaded standby node is allowed to send synchronization requests to its standby node.
- **only_sync_node**: If this parameter is set to **only_sync_node** for the primary node, the primary node is only allowed to proactively synchronize configuration files to all standby nodes. If this parameter is set to

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

only_sync_node for a standby node, the standby node is allowed to send synchronization requests to its primary node, and the standby node is not allowed to proactively synchronize configuration files to all cascaded standby nodes. If this parameter is set to **only_sync_node** for a cascaded standby node, the current cascaded standby node is allowed to send synchronization requests to its standby node.

- **none_node**: If this parameter is set to **none_node** for the primary node, the primary node is not allowed to proactively synchronize configuration files to all standby nodes. If this parameter is set to **none_node** for a standby node, the standby node is not allowed to send synchronization requests to its primary node, and the standby node is not allowed to proactively synchronize configuration files to all cascaded standby nodes. If this parameter is set to **none_node** for a cascaded standby node, the current cascaded standby node is not allowed to send synchronization requests to its standby node.

Default value: all_node

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: After this function is disabled, the primary node stops synchronizing GUC parameters to the standby node. In this case, exercise caution when setting this parameter to a value other than **all_node**.

Risks and impacts of improper settings: If this parameter is set to **none_node**, GUC parameters will not be synchronized between the primary and standby nodes. In this case, exercise caution when setting this parameter.

hadr_recovery_time_target

Parameter description: Specifies whether the standby database instance completes log writing and replay in streaming DR mode.

Parameter type: integer.

Unit: seconds

Value range: 0 to 3600

- **0** indicates that log flow control is disabled.
- 1 to 3600 indicates that the standby node can write and replay logs within the time specified by **hadr_recovery_time_target**. This ensures that logs can be written and replayed within the time specified by **hadr_recovery_time_target** during the switchover between the primary and standby database instances. In this case, the standby database instance can be quickly switched to the primary database instance.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, frequent flow control will affect host performance. If the value is too large or the parameter is disabled, the RTO of the standby node will exceed the expectation if the standby

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

node does not replay data in a timely manner. As a result, the RTO and RPO of HA will be decreased.

hadr_recovery_point_target

Parameter description: Specifies the RPO time allowed for the standby database instance to flush logs to disks in streaming DR mode.

Parameter type: integer.

Unit: seconds

Value range: 0 to 3600

- **0** indicates that log flow control is disabled.
- 1 to 3600 indicates that the standby node can flush logs to disks within the time specified by **hadr_recovery_point_target**. This ensures that logs can be replayed within the time specified by **hadr_recovery_point_target** during the switchover between the primary and standby database instances. In this case, the standby database instance is switched to primary.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, frequent flow control will affect host performance. If the value is too large or the parameter is disabled, the RTO of the standby node will exceed the expectation if the standby node does not replay data in a timely manner. As a result, the RTO and RPO of HA will be decreased.

hadr_super_user_record_path

Parameter description: Specifies the path for storing encrypted files of the **hadr_disaster** user in the standby cluster in streaming DR mode.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: The value is automatically set by the streaming DR password transfer tool and does not need to be manually added. You are advised not to change the value.

Risks and impacts of improper settings: If the settings are incorrect, the standby cluster status will be abnormal, and the primary cluster cannot be connected. As a result, data cannot be synchronized.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

check_sync_standby

Parameter description: Specifies whether to check the standby node. After the **synchronous_standby_names** parameter is correctly configured in the primary and standby scenarios, if the synchronous standby node is faulty, the write service on the primary node reports a write failure.

NOTE

- This parameter cannot be synchronized in job work and autonomous transactions. Otherwise, the check may not take effect.
- If the standby node check is not configured for a specified user or session and the standby node is faulty when the forcible synchronization commit mode is enabled, the write operation on a table causes the query on the same table by another user or in another session to hang. In this case, you need to recover the standby node or manually terminate the hung client.
- The standby node check function cannot be enabled in scenarios (such as VACUUM ANALYZE and gs_clean) where non-write operations trigger log writing. If the standby node does not meet the requirements for synchronizing configurations to the standby node, services will be hung in this scenario. In this case, you need to manually terminate the services.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The standby node check is enabled.
- **off**: The standby node check is disabled.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_wal_sender_crc_check

Parameter description: Specifies whether to enable CRC for the WAL sender thread. CRC is performed before the primary node sends logs. If this parameter is enabled, the performance may deteriorate by less than 5%.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The check function is enabled.
- **off**: The check function is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: This parameter ensures log validity. Determine whether to enable this function based on scenario requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

In DCF mode, the thread for sending logs is dcf_rep_appendlog. If **enable_wal_sender_crc_check** is enabled, CRC can be performed before the dcf_rep_appendlog thread sends logs in DCF mode.

syncrep_wait_proceed_timeout

Parameter description: Specifies the time threshold for timeout detection after the primary thread receives a termination signal when waiting for the majority confirmation during transaction commit. If the duration for which the flushing location of the majority of standby nodes is not updated exceeds the threshold, the primary node process exits.

Parameter type: integer.

Unit: seconds

Value range: 0 to 1000. **0** indicates that timeout detection is not performed after the termination signal is received.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If this timeout detection needs to be enabled, set this parameter to a value greater than or equal to the duration of the automatic copy reduction process. You are advised to set this parameter to a value greater than or equal to 60.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing. If the value is too small, the host process may exit in advance during the automatic copy reduction.

async_build

Parameter description: Specifies whether to use the asynchronous build process to rebuild a faulty DN. If this function is enabled, the asynchronous process is used for rebuilding. If the number of remaining copies is less than the majority, the service write capability can be preferentially restored. If this function is disabled, the synchronous logic is used for rebuilding. If the number of remaining copies is less than the majority, services can be restored only after data and logs are completely synchronized. After this function is enabled, if the standby DN to be rebuilt is required to participate in majority commit, the standby DN to be rebuilt must be in the synchronization list configured by the **synchronous_standby_names** parameter, and at least one DN in the synchronization list must be alive except the DN to be rebuilt.

Parameter type: Boolean.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range:

- **on**: The function of participating in majority commit in asynchronous build is enabled.
- **off**: The function of participating in majority commit in asynchronous build is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter to ensure quick recovery of services in fault scenarios. For example, in four-copy deployment, when two DNs are faulty, blocked services are written to new services because the majority cannot meet the requirements. After the function of asynchronous build to participate in majority commit is enabled, new services can be written during the rebuilding of the faulty DN.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTICE

1. The service recovery time is determined by the Xlog difference between each building instance and the primary node and the network rate. For example, if the Xlog difference is 1 GB and the network rate is 100 MB/s, the service should be recovered about 10s after the building starts to transmit Xlogs.
2. Data reliability of new transactions in the building state may be temporarily lost. That is, when available copies cannot meet the majority commit condition, the building receives logs to meet the majority commit condition, but the building instance does not have complete data. If the remaining copies cannot work and cannot be restored, the new data written during the building process may fail to be restored. After the building instance is restored or the automatic copy reduction capability of the CM is triggered, the risk can be eliminated. For example, in the four-copy scenario, if only one primary copy and one standby copy are normal, and the other two copies are in the building or to-be-repaired state, building receiving logs allows the transactions in the primary node to meet the majority commit condition. However, if the primary and standby copies in the normal state are damaged and cannot be restored, the new logs generated during the fault may fail to be restored.
3. This parameter can be set independently on each node of the cluster and takes effect only when it is set to **on** on both the primary and standby nodes. It is recommended that the parameter settings be the same on all nodes in the cluster.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3.7.3 Standby Node

hot_standby

Parameter description: Specifies whether the standby node is allowed to accept connections and queries after it is restored to the minrecovery point.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- If this parameter is set to **on**, **wal_level** must be set to **hot_standby** or higher. Otherwise, the database startup fails.
- In a distributed system, **hot_standby** cannot be set to **off** because this setting can affect other features of the HA system.
- If the **hot_standby** parameter was disabled and the **wal_level** parameter was set to a value smaller than that of **hot_standby**, perform the following operations to ensure that the logs to be replayed on the standby node can be queried on the standby node before enabling the **hot_standby** parameter again:
  1. Change the value of **wal_level** of the primary and standby nodes to the value of **hot_standby** or a higher value, and restart the instances for the change to take effect.
  2. Perform the checkpoint operation on the primary node and query the **pg_stat_get_wal_senders()** function to ensure that the value of **receiver_replay_location** of each standby node is the same as that of **sender_flush_location** of the primary node. Ensure that the value adjustment of **wal_level** is synchronized to the standby nodes and takes effect, and the standby nodes do not need to replay low-level logs.
  3. Set the **hot_standby** parameter of the primary and standby nodes to **on**, and restart the instances for the setting to take effect.
- If the read on standby function is enabled, conflicts between replay and query on the standby node may cause the query to cancel. The error information is as follows:
  - ERROR: canceling statement due to conflict with recovery.
  - ERROR: terminating connection due to conflict with recovery.
- During the standby node read in serial and parallel replay modes, when the primary node rebuilds indexes online, an error may be reported when the standby node is read. Try again later. Error information:
  - ERROR: could not open relation with OID xxx during recovery delete object, please try again later.
  - ERROR: Catalog is missing xxx attribute(s) for relid xxx.
  - ERROR: cache lookup failed for index xxx, refilenode:xxx, name:"xxx".
  - ERROR: could not find pg_class entry for xxx.
  - ERROR: could not open block during recovery delete object, please try again later.
- During the standby node read in the case of ultimate RTO, the VM with small memory configuration may report an error indicating insufficient memory. You can enable the GUC parameter **exrto_standby_read_opt** (enabled by default) to effectively reduce the memory and I/O overhead.
- The standby node does not support autonomous transactions.
- In the case of parallel replay and read on standby in ultimate RTO mode, Astore tables do not support the generation of the IndexOnlyScan execution plan.
- In the scenario where read on standby is enabled with ultimate RTO, if the **READ COMMITTED** transaction isolation level is required, the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

exrto_transaction_isolation_force_rr parameter must be set to **off**. When the **exrto_transaction_isolation_force_rr** parameter is set to **on**, only the **REPEATABLE READ** transaction isolation level is supported in the scenario where read on standby is enabled with ultimate RTO. When the standby node is read at the **REPEATABLE READ** isolation level, pay attention to the following two conditions: If neither of the two conditions is met, the newly replayed data cannot be read after the session is established.

  - In scenarios where applications connect to the database through drivers, the autocommit function must be enabled.
  - In scenarios where autocommit needs to be disabled, if the latest replayed data needs to be read during the execution of a long transaction, you need to manually commit the current query transaction and then perform a new query.

- Use sysbench to test the read performance of the standby node in typical scenarios: The primary node executes 100 concurrent update services. The primary and standby nodes both execute 200 concurrent read services at the same time. When the I/O and CPU are not limited, the read performance of the standby node in serial replay mode is not lower than 80% of that of the primary node, and the read performance of the standby node in ultimate RTO deteriorates by no more than 10% compared with that of the standby node in serial replay mode.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The standby node is allowed to accept connections and queries after it is restored to the minrecovery point.
- **off**: The standby node is not allowed to accept connections and queries after it is restored to the minrecovery point.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: After the read function of the standby node is enabled, certain resources, including the CPU, memory, disk space, and I/O usage, are consumed even if the standby node does not respond to services. If there is no read service on the standby node, you are advised to disable this parameter.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_standby_archive_delay

Parameter description: Specifies the wait period before queries on the standby node are canceled. A query conflict occurs when you use the standby node to read data in hot standby mode. In the current version, the setting does not take effect and is controlled by the **max_standby_streaming_delay** parameter.

Parameter type: integer.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: milliseconds

Value range: –1 to 2147483647. **-1** indicates that the standby node waits until the conflicting queries are complete.

Default value: **3000** (that is, 3s)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_standby_streaming_delay

Parameter description: Specifies the wait period before queries on the standby node are canceled when the queries conflict with WAL data receiving through streaming replication in hot standby mode. If this parameter is set to a large value or the service load is heavy, an error may be reported, indicating that queries conflict with conflict.

Parameter type: integer.

Unit: milliseconds

Value range: –1 to 2147483647

NOTICE

-1 indicates that the standby node waits until the conflicting queries are complete. In the scenario where serial or parallel replay is enabled, if the system detects that the query thread and replayer thread are in the deadlock state, the system still cancels the query to prevent replay blocking.

0 indicates that the standby node immediately cancels the query when a conflict between replay and query occurs.

Default value: **3000** (that is, 3s)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements. The default value is recommended. A larger value indicates a higher query service priority on the standby node, but affects replay on the standby node. A smaller value indicates that long queries are more likely to be canceled, but the replay priority is higher.

Risks and impacts of improper settings: If the value is too large and long queries conflict with replay, replay may be blocked for a long time, affecting the RTO and RPO of the standby node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wal_receiver_status_interval

Parameter description: Specifies the maximum interval for notifying the primary node of the WAL receiver status.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483

Default value: **5**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **0**, the standby node does not send information, such as the log receiving location, to the primary node. As a result, the transaction commit on the primary node may be blocked, and the switchover may fail. In normal service scenarios, you are advised not to set this parameter to **0**.

hot_standby_feedback

Parameter description: Specifies that the minimum transaction ID of a query performed on the standby node is sent to the primary node to avoid query conflicts.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The ID of the oldest transaction active on standby nodes will be sent to the primary node.
- **off**: The ID of the oldest transaction active on standby nodes will not be sent to the primary node.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **on**, the deletion of source version data on the primary node is restricted by the transactions that are being read by the standby node. That is, the primary node can delete only the changes whose transaction IDs are smaller than those reported by the standby node. Therefore, the performance of the primary node will be affected when this parameter is enabled. If replay conflicts with query on the standby node and a query error is reported, you are advised to increase the value of **max_standby_streaming_delay**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wal_receiver_timeout

Parameter description: Specifies the maximum wait period for a standby node to receive data from the primary node.

Parameter type: integer.

Unit: milliseconds

Value range: 0 to 2147483647

Default value: **6000** (that is, 6s)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the walreceiver thread is frequently disconnected when the network fluctuates.

wal_receiver_connect_timeout

Parameter description: Specifies the maximum timeout interval for the standby server to connect to the primary server.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483

Default value: **2**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

wal_receiver_connect_retries

Parameter description: Specifies the maximum attempts that a standby node connects to the primary node.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483647

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

wal_receiver_buffer_size

Parameter description: Specifies the memory buffer size for the standby nodes to store the received Xlog files.

Parameter type: integer.

Unit: KB

Value range: 4096 to 1047552

Default value: **65536** (that is, 64 MB)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

primary_slotname

Parameter description: Specifies the slot name of the primary node corresponding to a standby node. This parameter is used for verifying the primary-standby relationship.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is incorrectly set, the standby node may fail to connect to the primary node, and the cluster status may be abnormal.

max_standby_base_page_size

Parameter description: Specifies the maximum storage space of base page files on the standby node after the ultimate RTO function is enabled.

Parameter type: integer.

Unit: KB

Value range: 1048576 to 562949953421311

Default value: **268435456** (that is, 256 GB)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter according to the actual size of the physical disk.

Risks and impacts of improper settings: If the value is too small, the probability of read errors on the standby node increases.

max_standby_lsn_info_size

Parameter description: Specifies the maximum size of LSN info files on the standby node after the ultimate RTO function is enabled.

Parameter type: integer.

Unit: KB

Value range: 1048576 to 562949953421311

Default value: **268435456** (that is, 256 GB)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter according to the actual size of the physical disk.

Risks and impacts of improper settings: If the value is too small, the probability of read errors on the standby node increases.

max_keep_csn_info_size

Parameter description: Specifies the maximum size of CSN info linked lists allowed by each DN on the standby node after the ultimate RTO function is enabled.

Parameter type: integer.

Unit: KB

Value range: 16384 to 131072

Default value: **16384** (that is, 16 MB)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

base_page_saved_interval

Parameter description: Specifies the interval for generating base pages on the standby node after the ultimate RTO function is enabled. For the same page, a base page is generated each *n* (specified by this parameter) times.

Parameter type: integer.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range: 5 to 2000

Default value: **400**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the interval is too long, a large number of Xlogs need to be replayed each time the standby node reads data, and the performance deteriorates. If the interval is too short, more disk space is occupied and the forcible recycling point is reached more quickly.

standby_force_recycle_ratio

Parameter description: Specifies the percentage of files read by the standby node to trigger forcible recycling after the ultimate RTO function is enabled. When the total size of base page files exceeds the value of **max_standby_base_page_size** x **standby_force_recycle_ratio** or the total size of LSN info files exceeds the value of **max_standby_lsn_info_size** x **standby_force_recycle_ratio**, forcible recycling is triggered and some queries are canceled. When the value of **standby_force_recycle_ratio** is **0**, forcible recycling is not started, and the setting of **max_standby_base_page_size** and **max_standby_lsn_info_size** does not take effect.

Parameter type: floating-point number

Unit: none

Value range: 0.0 to 1.0

Default value: **0.8**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

standby_recycle_interval

Parameter description: Specifies the interval for the standby node to recycle read files after the ultimate RTO function is enabled. The thread for recycling read resources on the standby node attempts to clear read files on the standby node at the interval specified by this parameter.

Parameter type: integer.

Unit: seconds

Value range: 0 to 86400. The value **0** indicates that historical pages are not recycled and the recycling of Xlogs on the standby node is affected. As a result, the standby node occupies too much disk space.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **10**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the interval is too long, historical pages cannot be recycled in a timely manner, which may cause excessive disk usage.

standby_max_query_time

Parameter description: Specifies the maximum query time supported on the standby node after the ultimate RTO function is enabled. If the query time exceeds the value of this parameter, the query will be canceled.

NOTE

The time when the query is canceled is affected by the interval parameter **standby_recycle_interval** of the recycling thread and the time when the snapshot is obtained. Therefore, the actual execution time of the query on the standby node must be greater than the value of this parameter.

Parameter type: integer.

Unit: seconds

Value range: 0 to 86400. The value **0** indicates that when the recycling thread recycles historical pages, if the queried snapshot is earlier than the current recycling point, the query is canceled immediately.

Default value: **600**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set the value based on service requirements.

Risks and impacts of improper settings: If this parameter is set to a large value, the recycling thread cannot calculate the recycling point. As a result, the recycling of historical versions and Xlogs may be blocked, increasing the disk space usage.

exrto_transaction_isolation_force_rr

Parameter description: Specifies whether to forcibly set the transaction to the **REPEATABLE READ** isolation level in the scenario where read on standby is enabled with ultimate RTO.

Parameter type: Boolean

Unit: none

Value range:

- **on**: In the scenario where read on standby is enabled with ultimate RTO, the transaction is forcibly set to the **REPEATABLE READ** isolation level.
- **off**: In the scenario where read on standby is enabled with ultimate RTO, the transaction can be set to any isolation level.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

In the scenario where read on standby is enabled with ultimate RTO, you need to ensure that the value of **exrto_transaction_isolation_force_rr** is **off**, and then set the transaction isolation level using the **transaction_isolation** parameter.

Default value:

- For a newly installed database, the default value is **off**.
- If the GUC parameter does not exist in the source version, the default value is **on**. If the GUC parameter exists in the source version, the original setting is retained.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

exrto_standby_read_opt

Parameter description: Specifies whether to support read optimization of the standby node with ultimate RTO. This parameter is not synchronized between the primary and standby nodes.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The optimization is enabled.
- **off**: The optimization is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

walrcv_writer_crc_check_level

Parameter description: Specifies whether to enable Xlog verification on the standby node in the primary/standby clusters in the streaming DR scenario. By default, the parameter is enabled only in the standby cluster.

Parameter type: integer.

Unit: none

Value range: 0 to 3

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **0**: The verification is disabled.
- **1**: The verification takes effect in the standby cluster.
- **2**: The verification on the standby node in the primary cluster takes effect.
- **3**: The verification on all standby nodes in the primary cluster and nodes in the standby cluster take effect.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is automatically set in the DR establishment process. If the verification fails, the primary and standby nodes are disconnected, and the status of the standby cluster is abnormal for a short period of time. The status can be restored after the standby cluster is restarted.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

standby_read_use_az_info

Parameter description: Specifies whether the GTM-lite standby node uses AZ information to read data. The standby node in the same AZ as the CN is preferentially selected. This parameter is disabled by default.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements. After this function is enabled, all available standby DNs need to be traversed before being selected, which may cause query performance deterioration.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

standby_read_use_load_balance

Parameter description: Specifies whether the GTM-lite standby node uses the load balancing function to randomly select DNs that meet the consistency point (and AZ information). This function is disabled by default.

Parameter type: Boolean.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: enabled.
- **off**: disabled.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements. After this function is enabled, all available standby DNs need to be traversed before being selected, which may cause query performance deterioration.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_standby_bufferpool

Parameter description: Specifies whether to enable the function of reading data from an independent buffer on the standby node with ultimate RTO.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

When this function is enabled, it takes effect only when **enable_incremental_checkpoint** is enabled and the difference between the value of **standby_shared_buffers_fraction** and that of **standby_bufferpool_scale** is greater than or equal to 0.1.

standby_bufferpool_scale

Parameter description: Specifies the percentage of the **shared_buffers** memory buffer used by the independent buffer after the function of independent buffer for read on the standby node with ultimate RTO is enabled.

Parameter type: floating-point number

Unit: none

Value range: 0.1 to 0.9

Default value: **0.5**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to an improper value, page refreshing will be accelerated.

wal_replication_ssl

Parameter description: Specifies whether to enable SSL communication between primary and standby nodes. This parameter is used together with the GUC parameter **ssl**. SSL encrypted transmission is used only when both parameters are set to **on**. If SSL encrypted transmission is disabled, the communication performance between primary and standby nodes can be improved.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to an improper value, the communication security between the primary and standby nodes deteriorates.

6.3.7.4 Logical Replication

max_changes_in_memory

Parameter description: Specifies the maximum number of DML statements cached in memory for a single transaction during logical decoding.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483647

Default value: **4096**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, a large amount of dynamic memory is occupied when large transactions are decoded. If the value is too small, the decoding efficiency decreases.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

max_cached_tuplebufs

Parameter description: Specifies the upper limit of the total tuple information cached in the memory during logical decoding.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483647

Default value: **8192**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. You are advised to set this parameter to a value greater than or equal to twice of **max_changes_in_memory**.

Risks and impacts of improper settings: If the value is too large, a large amount of dynamic memory is occupied when large transactions are decoded. If the value is too small, the decoding efficiency decreases.

logical_decode_options_default

Parameter description: Specifies the global default parameters when a logical decoding task is started.

Parameter type: string.

Unit: none

Value range: Currently, the following logical decoding options can be specified: **parallel-decode-num**, **parallel-queue-size**, **max-txn-in-memory**, **max-reorderbuffer-in-memory**, **exclude-users**, **skip-generated-columns**, **decode-sequence**, **max-control-memory-size**, **max-toast-memory-size**, **max-queue-cache-num**, **desc-memory-limit**, **data-limit**, and **enable-cn-decode**. For details about the options, see "Feature Usage Guidelines > Logical Replication > Logical Decoding (Non-Multi-Primary) > Logical Decoding Options" in *Developer Guide*.

NOTICE

- Multiple key=value pairs in a string are separated by commas (,), for example, **"parallel-decode-num=4,parallel-queue-size=128,exclude-users=userA,skip-generated-columns=on"**.
- The **exclude-users** option specified by **logical_decode_options_default** is different from the logical decoding startup option **exclude-users**. You are not allowed to specify multiple blacklisted users.
- **decode-sequence** configured in **logical_decode_options_default** can only be set to **false**.

Default value: **""**. An empty string indicates that the default value of the program is used.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

The setting of this SIGHUP parameter does not affect the logical decoding process that has been started. The setting of this parameter will be used as the default configuration for subsequent logical decoding startup. If the same configuration item exists in the decoding task startup command, the default value of the corresponding option of **logical_decode_options_default** does not take effect. Instead, the value specified in the startup command is used.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

logical_decode_options_extend

Parameter description: Specifies the global default parameters when a logical decoding task is started. This parameter can be set at the PDB level.

Parameter type: string.

Unit: none

Value range: a string that contains XIDs, separated by slashes (/), for example, **"skip-xid-list=9737819/9737823"**. Valid XIDs range from 3 to 18446744073709551615.

Default value: **""**, indicating that the preset default value is used.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. Currently, the following logical decoding options can be specified: **skip-xid-list**. For details about the options, see "Feature Usage Guidelines > Logical Replication > Logical Decoding (Non-Multi-Primary) > Logical Decoding Options" in *Developer Guide*.

NOTICE

The setting of this parameter does not affect the logical decoding process that has been started. The setting of this parameter will be used as the default configuration for subsequent logical decoding startup. If the same configuration item exists in the decoding task startup command, the default value of the corresponding option of **logical_decode_options_extend** does not take effect. Instead, the value specified in the startup command is used.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after you fully understand the parameter meaning and test the parameter. If the XIDs are incorrectly set, transactions may be skipped unexpectedly, causing data loss during decoding.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

logical_sender_timeout

Parameter description: Specifies the maximum waiting time for the sender to wait for the receiver to receive logical logs.

Parameter type: integer.

Unit: milliseconds

Value range: 0 to 2147483647

Default value: **30000** (30s)

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_logicalrepl_xlog_prune

Parameter description: Specifies whether to enable the function of forcibly invalidating a logical replication slot. When the GUC parameters **enable_logicalrepl_xlog_prune** is set to **on**, **max_size_for_xlog_retention** is greater than 0, and the number of retained log segments (16 MB for each log segment) exceeds the value of **max_size_for_xlog_retention** due to the current logical replication slot, or **max_size_for_xlog_retention** is less than 0 and the disk usage reaches (–**max_size_for_xlog_retention**)/100, the logical replication slot will be forcibly invalidated, **restart_lsn** is set to **7FFFFFFF/FFFFFFFF**, and the replication slot does not block the recycling of Xlogs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The function of forcibly invalidating a logical replication slot is enabled.
- **off**: The function of forcibly invalidating a logical replication slot is disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Determine whether to enable this function based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_logical_replication_ddl

Parameter description: Specifies whether the logical decoding supports DDL statements, reverse parsing, and log generation.

Parameter type: Boolean.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range:

- **on**: Logical replication supports DDL statements, reversely parses DDL execution results, and generates DDL WALs.
- **off**: DDL statements are not supported, reverse parsing is not performed, and WALs are not generated.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_keep_log_seg

Parameter description: Flow control parameter. In logical replication, physical logs are parsed and converted into logical logs locally on the DN. When the number of physical log files that are not parsed is greater than the value of this parameter, flow control is triggered.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647. **0** indicates that the flow control function is disabled.

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Determine whether to perform flow control based on service requirements.

Risks and impacts of improper settings: If the value is too large, flow control will not be triggered.

logical_replication_dictionary_retention_time

Parameter description: Specifies the retention period of data in the system catalogs related to GS_TXN_LSN_TIME and the data dictionary.

Parameter type: integer

Unit: days (d)

Value range: 1 to 3650

Default value: **365**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Set this parameter based on service requirements.

Risks and impacts of improper settings: If the value is too small, the available interval for decoding the positioning point is too small. If the value is too large, the decoding performance is affected.

logical_replica_identity_force

Parameter description: Specifies the record level of old tuples for global logical replication.

Parameter type: enumerated type

Unit: none

Value range:

- **'full'**: All columns are recorded as replica identifiers.
- **'unique'**: The replica identifier is the primary key and unique key.
- **'default'**: The replica identifier is selected by default (primary key or none).
- **'nothing'**: There is no replica identifier for recording this relationship.
- **'none'**: The global control parameter does not take effect.

Default value: **'none'**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

NOTE

To reset **logical_replica_identity_force** that has been set to a non-default value to **'none'**, you need to restore the GUC parameter to the default value.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.8 Query Planning

This section describes the method configuration, cost constants, planning algorithm, and some configuration parameters for the optimizer.

NOTE

- Two parameters are involved in the optimizer:
  - *INT_MAX* indicates the maximum value of the INT data type. The value is **2147483647**.
  - *DBL_MAX* indicates the maximum value of the FLOAT data type.
- In addition to customer services, global query planning parameters also affect database O&M and monitoring services, such as WDR generation, scale-out, redistribution, and data import and export.

6.3.8.1 Optimizer Method Configuration

These configuration parameters provide a crude method of influencing the query plans chosen by the query optimizer. If the default plan chosen by the optimizer

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

for a particular query is not optimal, a temporary solution is to use one of these configuration parameters to force the optimizer to choose a different plan. Better ways include adjusting the optimizer cost constants, manually running **ANALYZE**, increasing the value of the **default_statistics_target** parameter, and increasing the amount of the statistics collected in specific columns using **ALTER TABLE SET STATISTICS**.

enable_bitmapscan

Parameter description: Specifies the query optimizer's use of bitmap-scan plan types.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this function is disabled, the bitmap scan operator is disabled globally, which may cause performance deterioration in some scenarios. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

force_bitmapand

Parameter description: Specifies whether the query optimizer forcibly uses BitmapAnd plan types.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this parameter is enabled, the optimizer forcibly performs bitmap and index scan when the conditions for using the bitmap and index scan operator are met. In some scenarios, the query performance may be affected. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_hashagg

Parameter description: Specifies the query optimizer's use of Hash aggregation plan types.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the Hash aggregation operator cannot be used in some scenarios, affecting the query rewriting logic and deteriorating the query performance. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

enable_hashjoin

Parameter description: Specifies whether the optimizer uses hash join plan types.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this function is disabled, the Hash Join operator cannot be used globally and other Join plans are preferentially generated. As a result, the performance may deteriorate in some scenarios. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

enable_indexscan

Parameter description: Specifies whether the optimizer uses index-scan plan types.

Parameter type: Boolean.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this function is disabled, the index scan operator is disabled globally, which may cause performance deterioration in some scenarios. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

enable_indexonlyscan

Parameter description: Specifies whether the optimizer uses index-only-scan plan types.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this function is disabled, the index-only-scan operator cannot be used globally. Some plans that require only index scan but do not require table return cannot be generated. As a result, the performance may deteriorate in some scenarios. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

enable_gsiscan

Parameter description: Specifies whether the query optimizer uses global secondary-index-only scan plan types.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: In scenarios where global secondary indexes are available, if this parameter is disabled, only global secondary index scan plans cannot be generated, which may affect the query performance.

enable_gsitablescan

Parameter description: Specifies whether the query optimizer uses the global secondary index table scan plan types.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: In scenarios where global secondary indexes are available, if this parameter is disabled, only global secondary index scan plans cannot be generated, which may affect the query performance.

gsi_fastpath_level

Parameter description: Specifies whether the optimizer executes the determination system of simple query rules of GSI. The simple query rules of GSI are as follows:

- A single table, no subquery, no sublink, or the target table is not a view.
- The query statement cannot generate an execution plan for an ordinary index.
- The query statement cannot generate the IndexOnlyScan execution plan of GSI.
- Query statements do not contain scan hints, such as TableScan and IndexScan.
- The WHERE condition contains an equivalent condition that contains the index key of the candidate GSI (unique and not a partial GSI index).

Parameter type: enumerated type.

Unit: none

Value range:

- **strict**: The equivalent condition contains all GSI index keys.
- **partial**: The equivalent condition contains partial index keys (leftmost match) of GSI. The performance may not be optimal.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **none**: disabled.

Default value: **strict**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Improper settings may affect plan generation in some GSI scenarios, affecting query performance.

NOTE

If you want to use **partial**:

```
gaussdb=# CREATE TABLE t1(a INT, b INT, c INT);
gaussdb=# CREATE GLOBAL UNIQUE INDEX t1_gsi ON t1(b,c);
-- Q1
gaussdb=# SELECT * FROM t1 WHERE b=1;
-- Q2
gaussdb=# SELECT * FROM t1 WHERE b=1 LIMIT 1;
```

In the preceding scenario, the GSI is created on the **b** and **c** columns. When **gsi_fastpath_level** is set to **partial**, the GSI non-stream return plan can be generated for query statements Q1 and Q2. The uniqueness of an index is ensured by the **b** and **c** columns. For query statement Q1, multiple results may be returned. The performance of the GSI non-stream return plan may not be optimal. For query statement Q2, the limit 1 condition is included, the performance of the GSI non-stream return plan is still optimal.

enable_default_gsi

Parameter description: (1) Specifies whether to create a global secondary index (GSI) by default when the primary key constraint or unique constraint is added to a column that does not contain a distribution key during table creation. (2) Specifies whether a GSI can be created for a hash bucket table.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: (1) A GSI is created by default when the primary key constraint or unique constraint is added to a column that does not contain a distribution key during table creation. (2) A GSI can be created for a hash bucket table.
- **off**: (1) A GSI is not created by default when the primary key constraint or unique constraint is added to a column that does not contain a distribution key during table creation. The table fails to be created. (2) A GSI cannot be created for a hash bucket table.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. You need to determine whether to enable this parameter based on the actual application scenario.

Risks and impacts of improper settings:

- Setting this parameter to **off**: (1) The primary key constraint or unique constraint cannot be added to columns that do not contain distribution keys

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

during table creation. You are advised to add distribution keys by reconstructing services. (2) GSIs cannot be created for hash bucket tables.

- If this parameter is set to **on**: (1) GSIs are created by default. PRIMARY KEY constraints or UNIQUE constraints can be added to columns that do not contain distribution keys in the table. Note that if GSIs are created on the table, DML operations may cause potential performance deterioration. (2) Some indexes and GSIs in the form of expression indexes on hash bucket tables are rebuilt offline during redistribution. During the rebuilding, services are blocked. GSIs that are not partial indexes or expression indexes are rebuilt online during redistribution, and services are not blocked. GSIs are valid during scale-out and redistribution.

enable_gsi_iud_returning

Parameter description: Specifies whether to enable the RETURNING functionality when INSERT, UPDATE, and DELETE operations are performed on a base table with GSIs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enable.
- **off**: disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled when INSERT, UPDATE, or DELETE operations are performed on a base table with GSIs, the RETURNING functionality will not be supported.

enable_gsi_distkey_update

Parameter description: Specifies whether to enable the functionality of updating the distribution key of a GSI (UPDATE, UPSERT, and MERGE INTO).

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enable.
- **off**: disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this parameter is disabled, the distribution key of the GSI cannot be updated.

enable_gsi_dml_lp_opt

Parameter description: Specifies whether the optimizer executes the determination system of simple DML rules of the GSI to optimize the execution of simple DML statements. The simple DML optimization rules of the GSI are as follows:

- For INSERT statements, only the following simple INSERT statements are supported:
  - Only a single VALUES clause is supported. Multiple VALUES clauses are not supported.
  - The values in VALUES must be constants or parameters (expressions are not supported).
  - The RETURNING clause, subqueries in any form, and UPSERT statements are not supported.
  - The target table must be a single table that has no more than one GSI. The GSI must be an ordinary GSI (expression indexes or partial indexes are not supported) and cannot contain columns of user-defined types.
- Other DML statements: Not supported currently.
- This parameter must be set on both CNs and DNs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enable.
- **off**: disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, GSI DML Light Proxy optimization is not supported, which may cause performance deterioration in some GSI DML scenarios.

enable_material

Parameter description: Specifies whether the query optimizer uses materialization. It is impossible to suppress materialization entirely, but setting this parameter to **off** prevents the optimizer from inserting materialized nodes.

Parameter type: Boolean.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this function is disabled, the optimizer is prohibited from inserting entity nodes. As a result, the Material operator cannot be generated in some scenarios, and the performance may deteriorate in some scenarios.

enable_mergejoin

Parameter description: Specifies whether the optimizer uses merge-join plan types.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is enabled, the optimizer generates a merge join path for joins that meet the conditions. As a result, the selected join operator in this scenario may not be the optimal one.

enable_nestloop

Parameter description: Specifies whether the query optimizer uses nested-loop join plan types to fully scan internal tables. It is impossible to eliminate nested-loop joins. However, if this parameter is disabled, the optimizer prefers other methods when other methods exist.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is enabled, the optimizer generates a nested-loop join path for joins that meet the conditions. As a result, the selected join operator in this scenario may not be the optimal one.

enable_index_nestloop

Parameter description: Specifies whether the optimizer uses the nested-loop join plan type to scan the parameterized indexes of inner tables.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, parameterized paths cannot be generated in some scenarios. Generally, parameterized paths can improve query performance. Therefore, you are advised to enable this parameter.

enable_remote_param

Parameter description: Specifies whether to use the XC parameterized path feature. If this parameter is enabled, an XC parameterized path plan can be generated.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value:

- **on**: default value of the newly installed database.
- **off**: default value after an upgrade from a version earlier than 505.2.0.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

> **NOTE**
>
> This parameter was introduced in version 505.2.0 and its default value was changed from **off** to **on** in version 507.0.0. After an upgrade from a version earlier than 505.2.0, the value of this parameter is **off**. If this parameter exists and has been set before the upgrade, the original value will be retained after the upgrade.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: In correlated sublink and nested-loop scenarios, if the outer table is small and the inner table is large, and the inner table has an index that meets the leftmost prefix matching rule in the join condition, you are advised to set this parameter to **on**. In this configuration, if the cost model selects a parameterized path, the performance can be improved.

Risks and impacts of improper settings: In correlated sublink and nested-loop scenarios, if the outer table is large and the inner table is small, setting this parameter to **on** and selecting a parameterized path for the cost model may cause performance deterioration.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- The conditions for generating an XC parameterized path are as follows:
  - The condition cannot contain expressions that cannot be pushed down (for example, a function whose PROVOLATILE attribute is volatile).
  - The condition can match an index according to the leftmost matching rule.
  - The WHERE condition of an outer join does not support parameterization.
- XC parameterization supports the following scenarios:
  - A parameterized path for nested loops can be generated, and the join condition can be pushed down to the inner table RemoteQuery (including single-table RemoteQuery and RemoteJoinQuery) so that the query result of the outer table can be passed to the inner table as a parameter.
  - A parameterized path for subplans can be generated, and the Coordinator Qual on the subplan can be pushed down to the RemoteQuery of the subplan so that the result of the parent query of the subplan can be passed to the inner table as a parameter.
- The limitations on using the XC parameterized path feature are as follows:
  - Only XC plans are supported. Stream and FQS plans are not supported.
  - A plan with the parameter **rescan** is generated for the base table only when the index parameterized path is generated for the base table.
  - The nested loop parameterized path supports only the scenario where the inner table is **RemoteQuery**.
  - The nested-loop parameterized path does not support pushing parameters down to subqueries.
  - The predpush hint is not supported.
  - In an M-compatible database, if the **max_datanode_for_plan** parameter is set to a value greater than 0, the DN plan of the parameterized path cannot be displayed.
  - Dynamic DN pruning for remote queries is not supported.

enable_remote_pbe_reuse

Parameter description: Specifies whether to reuse the PBE packets of DNs delivered by the CN as planned by the XC. After this function is enabled, the packets sent by the CN to the DN in the XC plan are changed from unnamed packets to packets with the **_disqry_sys_cached** prefix. The packets are reused during execution, improving query performance.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the XC plan query performance will be affected in some scenarios.

enable_seqscan

Parameter description: Specifies whether the query optimizer uses sequential scan plan types. It is impossible to eliminate sequential scan. However, if this parameter is disabled, the optimizer prefers other methods when other methods exist.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the optimizer tends to select other scan methods and lowers the priority of sequential scan methods. In some scenarios, the query performance may be affected.

enable_sort

Parameter description: Specifies whether the optimizer uses sort methods. It is impossible to eliminate explicit sorting. However, if this parameter is disabled, the optimizer prefers other methods when other methods exist.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the optimizer prefers other methods and lowers the priority of the sorting method. Therefore, you are advised to enable this parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_tidscan

Parameter description: Specifies whether the optimizer uses Tuple ID (TID) scan plan types.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the optimizer prefers other methods and lowers the priority of the TID scan method. Therefore, you are advised to enable this parameter.

enable_kill_query

Parameter description: In CASCADE mode, when a user is deleted, all the objects belonging to the user are deleted. This parameter specifies whether the queries of the objects belonging to the user can be unlocked when the user is deleted.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the unlocking is allowed.
- **off** indicates that the unlocking is not allowed.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this parameter is enabled, when a user is deleted, the query of the user may be exposed, affecting system consistency and security.

enable_stream_concurrent_update

Parameter description: Specifies the optimizer's use of streams in concurrent update scenarios. This parameter is restricted by the **enable_stream_operator** parameter. If concurrent update conflicts occur, an error may be reported. This parameter can be used to control the conflict.

Parameter type: Boolean.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range:

- **on** indicates that the optimizer can generate stream plans for the UPDATE statement.
- **off** indicates that the optimizer can generate only non-stream plans for the UPDATE statement.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the optimizer prefers other methods and generates only non-stream plans for update statements. Therefore, you are advised to enable this parameter.

enable_stream_operator

Parameter description: Specifies whether the optimizer uses stream. When the **enable_stream_operator** parameter is disabled, a large number of logs indicating that the stream plans cannot be pushed down are recorded. If you do not need these logs, you are advised to disable **enable_unshipping_log** when disabling **enable_stream_operator**. During bare metal installation, this parameter is set to **off**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value:

- Independent deployment: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to an improper value, the optimizer cannot generate stream plans, which may deteriorate the query performance.

enable_stream_recursive

Parameter description: Specifies whether to push **WITH RECURSIVE** join queries to DNs for processing.

Parameter type: Boolean.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **on** indicates that WITH RECURSIVE join queries will be pushed down to DNs.
- **off** indicates that WITH RECURSIVE join queries will not be pushed down.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the query performance may deteriorate.

max_recursive_times

Parameter description: Specifies the maximum number of WITH RECURSIVE iterations.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **200**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements.

Risks and impacts of improper settings: This parameter is used to prevent an infinite loop of the WITH RECURSIVE statement by using nocycle in hierarchical queries to exit the loop. If you want to adjust the value, make sure you understand the parameter meaning and adjust the value with caution to avoid risks caused by misoperations.

enforce_a_behavior

Parameter description: Specifies the rule matching modes of regular expressions.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The A matching rule is used.
- **off**: The POSIX matching rule is used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the actual service scenario.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: The regular expression in the query statement cannot be matched as expected.

enable_vector_engine

Parameter description: Specifies whether the optimizer uses vectorized engine.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the optimizer may fail to generate vectorized query plans for specific scenarios, affecting query performance.

enable_broadcast

Parameter description: Specifies whether the query optimizer uses broadcast distribution method when it evaluates the cost of stream.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The accuracy of stream plan generation and the query performance are affected.

enable_change_hjcost

Parameter description: Specifies whether the optimizer excludes inner table running costs when selecting the hash join cost path. If it is set to **on**, tables with a few records and high running costs are more possible to be selected.

Parameter type: Boolean.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The hash join cost may be incorrectly estimated, affecting the accuracy of generating hash join plans.

best_agg_plan

Parameter description: Specifies the type of hash aggregation plan generated by the optimizer. The optimizer can generate three plans for the aggregation operation in a stream:

1. hashagg+gather(redistribute)+hashagg
2. redistribute+hashagg(+gather)
3. hashagg+redistribute+hashagg(+gather)

Parameter type: integer.

Unit: none

Value range: 0 to 3.

- **0**: The optimizer generates the optimal plan based on the estimated costs of the three plans.
- **1**: The first plan is forcibly generated.
- **2**: The second plan is forcibly generated if the **group by** column can be redistributed. Otherwise, the first plan is generated.
- **3**: The third plan is forcibly generated if the **group by** column can be redistributed. Otherwise, the first plan is generated.

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: This parameter affects the hashagg generation mode. You are advised to use the default value. If this parameter is set to another value, the corresponding hashagg generation mode is preferred. If this parameter is set to an improper value, the query performance in some scenarios may be affected.

agg_redistribute_enhancement

Parameter description: When the aggregate operation is performed, which contains multiple **group by** columns and all of the columns are not in the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

distribution key, you need to select one **group by** column for redistribution. This parameter specifies the policy of selecting a redistribution key.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The column that can be redistributed with the most distinct value evaluated is selected for redistribution.
- **off**: The first column that can be redistributed is selected for redistribution.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is enabled, the plan generation time increases, but a more reasonable redistribution key may be selected.

enable_absolute_tablespace

Parameter description: Specifies whether the tablespace can use the absolute path.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: An absolute path can be used.
- **off**: No absolute path can be used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, absolute paths cannot be used when tablespaces are created. You are advised to set this parameter based on service requirements. No other risk exists.

enable_valuepartition_pruning

Parameter description: Specifies whether the DFS partitioned table is dynamically or statically optimized.

Parameter type: Boolean.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: The DFS partitioned table is dynamically or statically optimized.
- **off**: The DFS partitioned table is not dynamically or statically optimized.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this parameter is disabled, the query performance of DFS partitioned tables may deteriorate.

expected_computing_nodegroup

Parameter description: Specifies a compute node group or the way to choose such a group. The node group mechanism is internal. You do not need to set it. During join or aggregation operations, a node group can be selected in four modes. In each mode, the specified candidate compute node groups are listed for the optimizer to select the most appropriate one for the current operator.

Parameter type: string.

Unit: none

Value range:

- **optimal**: The list of candidate compute node groups consists of the node groups where the operator's operation objects are located and the node group that combines all DNs in the node groups on which the current user has the COMPUTE permission.
- **query**: The list of candidate compute node groups consists of the node groups where the operator's operation objects are located and the node group that combines all DNs in the node groups where base tables involved in the query are located.
- *Node group name* (when **enable_nodegroup_debug** is set to **off**): The list of candidate compute node groups consists of the node groups where the operator's operation objects are located and the specified node group.
- *Node group name* (when **enable_nodegroup_debug** is set to **on**): A specific node group is used as the compute node group.

Default value: **query**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The optimizer may fail to select a better compute node group, affecting query performance.

enable_nodegroup_debug

Parameter description: Specifies whether the optimizer assigns computing workloads to a specific node group when multiple node groups exist in an environment. The node group mechanism is internal. You do not need to set it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

This parameter takes effect only when **expected_computing_nodegroup** is set to a specific node group.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Computing workloads are assigned to the node group specified by **expected_computing_nodegroup**.
- **off**: No node group is specified for computing.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The query performance may be affected.

enable_default_index_deduplication

Parameter description: Specifies whether to deduplicate and compress tuples with duplicate key values for a B-tree index by default. The deduplication and compression functions do not take effect for primary key indexes and unique indexes. When there are a large number of indexes with duplicate key values, the deduplication and compression function can effectively reduce the space occupied by indexes. If the WITH (**deduplication** set to **on**/**off**) syntax is used during index creation, the deduplication parameter is preferentially used to determine whether to use deduplication and compression for the index.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The deduplication and compression function is enabled for B-tree indexes.
- **off**: The deduplication and compression function is disabled for B-tree indexes.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. You can enable this parameter for non-unique indexes, high index key value repetition, and large index space usage.

Risks and impacts of improper settings: In scenarios where non-unique indexes are used and index key values are seldom repeated or unique, the deduplication and compression function slightly deteriorates the index insertion performance.

expr_engine

Parameter description: Specifies the execution mode of an expression.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: enumerated type.

Unit: none

Value range:

- **EXPR_TREE**: The recursive engine of the expression is used.

Default value: **EXPR_TREE**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter can only be set to **EXPR_TREE**.

Risks and impacts of improper settings: none

NOTE

The current version does not support expression execution optimization. Therefore, this parameter can only be set to the default value **EXPR_TREE**.

sql_result_check

Parameter description: Specifies the self-verification function of expression results.

Parameter type: enumerated type.

Unit: none

Value range:

- **off**: The expression-based result verification functionality is disabled.
- **single_select**: In the type range that supports expression result verification, if the verification of a column of the numeric precision type fails, an alarm is generated. If the verification of a column of other types fails, an error is thrown.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: When **single_check** is used, the query execution time increases slightly.

stream_multiple

Parameter description: Specifies the weight used by the optimizer to calculate the final cost of stream operators. The base stream cost is multiplied by this weight to obtain the final cost.

NOTE

This parameter is applicable only to Redistribute and Broadcast streams.

Parameter type: floating-point

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range: 0 to *DBL_MAX*

Default value: **1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value and adjust the value based on the deployment environment.

Risks and impacts of improper settings: The optimizer may incorrectly estimate the overhead of the stream operator, affecting the query performance.

qrw_inlist2join_optmode

Parameter description: Specifies whether to enable inlist-to-join query rewriting.

Parameter type: string.

Unit: none

Value range:

- **disable**: The inlist-to-join query rewriting is disabled.
- **cost_base**: cost-based inlist-to-join query rewriting. That is, whether to use inlist-to-join query rewriting is automatically evaluated based on the cost.
- **rule_base**: The rule-based inlist-to-join query rewriting is forcibly enabled.
- A positive integer (1 to 2147483647) indicates the threshold of inlist-to-join query rewriting. If the number of elements in the list is greater than the threshold, the rewriting is performed.

Default value: **cost_base**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Use other values only when manual optimization is required in abnormal scenarios or when the query performance of the plan selected by the optimizer is low.

Risks and impacts of improper settings: Whether the optimizer performs inlist-to-join query rewriting is affected. No better plan is generated due to improper settings.

NOTICE

The following data types can use the inlist-to-join query rewrite function: TINYINT, SMALLINT, INTEGER, BIGINT, NUMERIC, FLOAT4, FLOAT8, BOOLEAN, CHAR, BPCHAR, VARCHAR, NVARCHAR, TEXT, DATE, TIME, TIMEZ, TIMESTAMP, TIMESTAMPTZ, SMALLDATETIME, INTERVAL, TINTERVAL, UNSIGNED TINYINT, UNSIGNED SMALLINT, UNSIGNED INTEGER, and UNSIGNED BIGINT.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

skew_option

Parameter description: Specifies whether an optimization policy is used.

Parameter type: enumerated type.

Unit: none

Value range: **off**, **normal**, and **lazy**

- **off** indicates that the policy is disabled.
- **normal** indicates that a radical policy is used. All possible skews are optimized.
- **lazy** indicates that a conservative policy is used. Uncertain skews are ignored.

Default value: **normal**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Whether the optimizer optimizes skew is affected. The plan generation performance or query execution performance is affected due to improper settings.

default_limit_rows

Parameter description: Specifies the default estimated number of limit rows for generating genericplan.

Parameter type: floating-point

Unit: none

Value range: –100 to *DBL_MAX*. The recommended value range is 0.1 to 10000. If the value is beyond this range, service risks may occur.

- If this parameter is set to a non-negative number, the value is used as the estimated number of limit rows.
- If this parameter is set to a negative value, the value is used as the default estimated number in percentage. A negative value is converted to the default percentage, that is, **–5** indicates 5%.

Default value: **-10**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The genericplan query performance of the limit operator may be affected.

check_implicit_conversions

Parameter description: Specifies whether to check candidate index paths generated for index columns that have implicit type conversion in a query. For

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

details about the application scenarios of this parameter, see "Checking the Implicit Conversion Performance" in *SQL Optimization Guide*.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: A check will be performed for candidate index paths generated for index columns that have implicit type conversion in a query.
- **off**: No check will be performed.

Default value: **off**

NOTE

- When this parameter is set to **on**, you need to set **enable_fast_query_shipping** to **off** so that the mechanism for identifying implicit data type conversion in index columns can take effect.
- This parameter is used to check implicit conversion performance issues. Disable this parameter in the formal production environment.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impact of improper settings: If the check is not performed, the regular expression in the query statement may not be matched as expected.

cost_weight_index

Parameter description: Specifies the cost weight of index_scan.

Parameter type: floating-point

Unit: none

Value range: 1e-6 to 1e+10

Default value: **1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: This parameter specifies whether the optimizer uses index_scan. The optimizer may use a non-optimal plan due to improper settings, affecting query performance.

try_vector_engine_strategy

Parameter description: Specifies the policy for processing row-store tables by using the vectorized engine. By setting this parameter, queries containing row-store tables can be converted to vectorized execution plans for calculation, improving the execution performance of complex queries in AP-like scenarios.

Parameter type: enumerated type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range:

- **off**: This function is disabled. That is, row-store tables will not be converted into vectorized execution plans for execution.
- **force**: The base table to be queried is a row-store table as long as queries do not contain types or expressions that are not supported by the vectorized engine. Queries are forcibly converted to vectorized execution plans for execution. In this case, the performance may deteriorate in different query scenarios.
- **optimal**: On the basis of **force**, the optimizer determines whether to convert a query statement into a vectorized execution plan based on the query complexity to avoid performance deterioration after the conversion.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Whether the optimizer uses a vectorized execution plan is affected. The query performance may be affected in some scenarios due to improper settings.

enable_opfusion_reuse

Parameter description: Specifies whether to reuse the opfusion obj memory.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the opfusion obj memory is reused when conditions are met.
- **off**: This function is disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, plans may be regenerated and the query performance may deteriorate.

enable_iud_fusion

Parameter description: Specifies whether to optimize the IUD statements.

Parameter type: Boolean.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on** indicates that the input is optimized during type conversion.
- **off**: This function is disabled.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. You can enable this function if the IUD statement performance needs to be optimized.

Risks and impacts of improper settings: If this parameter is enabled, the optimizer cost increases without significant risk.

enable_cached_operator

Parameter description: Specifies whether to enable the cache feature for operators in execution state.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Operators can be executed in cache mode.
- **off**: This function is disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the operator execution status cannot be cached, affecting the query performance.

enable_smp_partitionwise

Parameter description: Specifies whether to enable the Partition-wise Join capability in the SMP scenario.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The Partition-wise Join capability is enabled in the SMP scenario.
- **off**: The Partition-wise Join capability is disabled in the SMP scenario.

Default value:

- **on**: default value of the newly installed database.
- **off**: This parameter is added in 505.2.0. If a database is upgraded from a version earlier than 505.2.0, the value of this parameter is **off**. If this parameter exists and is set before the upgrade, the default value remains unchanged after the upgrade.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The data query performance in the SMP scenario may be affected.

force_smp_partitionwise_scan

Parameter description: Specifies whether to enable the Partition-wise Scan parallel scan mode globally after the Partition-wise Join switch **enable_smp_partitionwise** is turned on in the SMP scenario.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The Partition-wise Scan parallel scan mode is enabled in the SMP scenario.
- **off**: The Partition-wise Scan parallel scan mode is disabled in the SMP scenario.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The performance of reading partitioned table data in the SMP scenario may be affected.

enable_nestagg

Parameter description: Specifies whether to enable nested aggregate function. Nested aggregation is a double aggregation syntax similar to sum(min(col)).

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The nested aggregate function is enabled.
- **off**: The nested aggregate function is disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The nested aggregate function is affected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_index_skip_scan

Parameter description: Specifies whether to enable the index skip scan feature. If this feature is disabled, a full table scan is performed when a user uses a composite index for query and the leading column is empty. If this feature is enabled, when a user uses a combination index for query and the leading column is empty, the database determines whether to perform index skip scanning based on the proportion of the number of distinct values (NDV). A lower proportion of NDV indicates greater advantage of index skip scan. When the proportion of the NDV is high, the cost of index skip scan may be higher than that of full table scan. As a result, the database still uses full table scan.

NOTE

- Leading columns cannot be separated. For example, after an index is created for columns **(a,b,c,d)** of table **t1**, you cannot run the **SELECT * FROM t1 WHERE b =** *xxx* **AND d =** *xxx* statement for query.
- The leading columns cannot be a range.
- Expression indexes are not supported.
- Only RCR B-tree and UB-tree indexes are supported.
- Only IndexScan and IndexOnlyScan are supported. BitmapIndexScan is not supported.
- SQL Bypass is no longer evolved. Therefore, this feature does not apply to SQL Bypass.
- SMP is not supported.
- Hash bucket tables are not supported.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The index skip scan feature is enabled.
- **off**: The index skip scan feature is disabled.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the index skip scan feature is disabled, index scan cannot be used when a combination index is used for query and the leading column is empty. As a result, the query efficiency is reduced.

enable_gsi_stream_dml

Parameter description: Specifies whether to enable the stream feature for DML operations on base tables that have global secondary indexes.

Parameter type: Boolean.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: The stream optimization is enabled for DML operations on base tables with global secondary indexes. That is, stream plans can be generated.
- **off**: The stream optimization is disabled for DML operations on base tables with global secondary indexes. That is, stream plans cannot be generated.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the performance may deteriorate.

enable_partition_statistic

Parameter description: Specifies whether to use partition-level statistics in the optimizer phase. A scenario where partition-level statistics can be used is as follows: In a Cplan, the partitioned table involved in the query is statically pruned to a single partition, including pruning a partitioned table to a single partition.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Partition-level statistics can be used.
- **off**: Partition-level statistics cannot be used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The statistics of a partitioned table when the partitioned table is statically pruned to a single partition are affected, and the plan generation is affected.

dist_query_adaptive_mode

Parameter description: Specifies the mode selected by the distributed adaptive plan. The value can be **auto** (default), **plan**, **sql**, or **off**. If the parameter value is not **off**, the cost of the Remote Query operator is calculated.

Parameter type: enumerated type.

Unit: none

Value range:

- **auto**: The distributed plan is adaptively selected based on the cost or rule. If a rule exists (the **dist_query_adaptive_rows** parameter is not set to **0**), the rule has a higher priority than cost-based selection.
- **plan**: The distributed plan of the delivered plan is forcibly executed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **sql**: The distributed plan of the delivered statement is forcibly executed.
- **off**: Adaptive selection of distributed plans is disabled.

Default value:

- **auto**: default value of the newly installed database.
- **off**: This parameter is introduced in 507.0.0. After an upgrade from an earlier version, the value of this parameter is **off**. If this parameter exists and has been set before the upgrade, the original value will be retained after the upgrade.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The optimizer's selection of the generation plan will be affected, and the query performance will be affected in some scenarios. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

NOTE

The process of adaptive selection of distributed plans is as follows: First, a distributed plan (stream plan) for the delivered plan is generated. By analyzing the characteristics of the stream plan, the system determines whether to trigger the adaptive selection of distributed plans. If adaptive optimization is required, a new distributed plan for the delivered statement is generated.

- Cost-based adaptive constraints on distributed plans:
  - Only the SELECT statement supports adaptive selection of distributed plans.
  - If a statement contains a stream hint, adaptive distributed optimization cannot be enabled for the plan.
  - If a stream plan contains only the Stream Gather operator and no other stream operators, adaptive distributed optimization cannot be enabled for the plan.
  - If a stream plan is an SMP plan, adaptive distributed optimization cannot be enabled for the plan.
  - If a stream plan does not contain the join operation or related sublinks, adaptive distributed optimization cannot be enabled for the plan.
  - If the cost of a stream plan is relatively low, adaptive distributed optimization cannot be enabled for the plan.
  - When the memory or CPU usage of a CN is higher than 80%, if the distributed plan generated by the distributed adaptive plan for the delivered statement is a non-parameterized path plan, the system falls back to the stream plan. This constraint is controlled by the **opt_behavior_knob** option **DIST_QUERY_ADAPTIVE_FALLBACK** and is enabled by default.
- Rule-based adaptive constraints on distributed plans:
  - Only the SELECT statement supports adaptive selection of distributed plans.
  - If a statement contains a stream hint, adaptive distributed optimization cannot be enabled for the plan.
  - If a stream plan contains only the Stream Gather operator and no other stream operators, adaptive distributed optimization cannot be enabled for the plan.
  - If a stream plan is an SMP plan, adaptive distributed optimization cannot be enabled for the plan.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

dist_query_adaptive_rows

Parameter description: Selects a distributed plan rule based on the number of rows of the stream operator. If the total number of rows of all stream operators (except the Stream Gather operator) is less than or equal to the value of this parameter, the distributed plan of the delivered statement is selected. If the total number of rows is greater than the value of this parameter, the distributed plan of the delivered plan is selected.

Parameter type: integer.

Unit: none

Value range: [0,2147483647]

Default value: **0** (indicating that the rule is disabled). This parameter is introduced in 507.0.0. After an upgrade from an earlier version, the value of this parameter is **0**. If this parameter exists and has been set before the upgrade, the original value will be retained after the upgrade.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter when there are a large number of DNs. Adjust the value based on the service data scale. It is recommended that the value be less than or equal to 500.

Risks and impacts of improper settings: The optimizer's selection of the generation plan will be affected, and the query performance will be affected in some scenarios. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

NOTE

The Stream operator needs to establish connections with all DNs to transmit data. In scenarios with a large number of DNs and high concurrency, data communication bottlenecks may occur. Therefore, it is recommended that this parameter be used in services where the number of primary DNs in the cluster is greater than 8 and the number of concurrent requests is greater than 100. Before using this parameter, test and adjust its value based on service requirements.

enable_constraint_optimization

Parameter description: Specifies whether to enable the query optimization function based on informative constraints. When this function is enabled, the optimizer can use the constraint information on the table to generate a more efficient query execution plan. The main scenarios are as follows: external data source scan optimization, redundant NOT NULL constraint elimination, and scan and connection optimization that contain the UNIQUE constraint.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The query optimization function based on information constraints is enabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **off**: The query optimization function based on information constraints is disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Improper settings may affect the query performance in some scenarios.

6.3.8.2 Optimizer Cost Constants

This section describes the optimizer cost constants. The cost variables described here are measured on an arbitrary scale. Only their relative values matter, therefore scaling them all up or down by the same factor will result in no change in the optimizer's choices. By default, they use the cost of fetching sequential pages as the basic unit. That is, **seq_page_cost** is set to **1.0** and the other cost variables are set with reference to the parameter. However, you can use a different scale, such as actual execution time in milliseconds.

seq_page_cost

Parameter description: Specifies the optimizer's estimated cost of a disk page fetch that is part of a series of sequential fetches.

Parameter type: floating-point number

Unit: none

Value range: 0 to *DBL_MAX*

Default value: **1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: This parameter affects the performance overhead evaluation of sequentially fetching disk pages in a query. If this parameter is incorrectly set, the optimizer's selection of the generation plan is affected, and the query performance is affected in some scenarios. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

random_page_cost

Parameter description: Specifies the optimizer's estimated cost of an out-of-sequence disk page fetch.

Parameter type: floating-point number

Unit: none

Value range: 0.1 to *DBL_MAX*. The recommended value range is 0.1 to 10. If the value is beyond this range, service risks may occur.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **4**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: See the parameter description.

Risks and impacts of improper settings: This parameter affects the performance overhead evaluation of randomly fetching disk pages in a query. If this parameter is incorrectly set, the optimizer's selection of the generation plan is affected, and the query performance is affected in some scenarios. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

NOTE

- Although the server allows you to set **random_page_cost** to a value less than that of **seq_page_cost**, it is not physically sensitive to do so. Setting them equal makes sense if the database is entirely cached in RAM because in that case there is no penalty for fetching pages out of sequence. Also, in a heavily-cached database you should lower both values relative to the CPU parameters, since the cost of fetching a page already in RAM is much smaller than it would normally be.
- For tables and indexes in a specified tablespace, you can set a tablespace parameter with the same name to overwrite the database-level parameter value. That is, the priority of the tablespace-level parameter is higher than that of the database-level parameter.
- Reducing this value relative to **seq_page_cost** will cause the system to prefer index scans and raising it will make index scans relatively more expensive. You can increase or decrease both values together to change the disk I/O costs relative to CPU costs.
- The value **1.1** is suitable for SSDs. If HDDs are used, you are advised to change the value to **4**.

cpu_tuple_cost

Parameter description: Specifies the optimizer's estimated cost of processing each row during a query.

Parameter type: floating-point number

Unit: none

Value range: 0 to *DBL_MAX*

Default value: **0.01**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: This parameter affects the evaluation of the performance overhead for processing a row of data in a query. If this parameter is incorrectly set, the optimizer's selection of the generation plan is affected, and the query performance is affected in some scenarios. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

cpu_index_tuple_cost

Parameter description: Specifies the optimizer's estimated cost of processing each index entry during an index scan.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: floating-point number

Unit: none

Value range: 0 to *DBL_MAX*

Default value: 0.005

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: This parameter affects the performance cost evaluation of index scan in a query. If this parameter is incorrectly set, the optimizer's selection of the generation plan is affected, and the query performance is affected in some scenarios. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

cpu_operator_cost

Parameter description: Specifies the optimizer's estimated cost of processing each operator or function executed during a query.

Parameter type: floating-point number

Unit: none

Value range: 0 to *DBL_MAX*

Default value: **0.0025**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: This parameter affects the performance overhead evaluation of operators or functions in a query. If this parameter is incorrectly set, the optimizer's selection of the generation plan is affected, and the query performance is affected in some scenarios. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

effective_cache_size

Parameter description: Specifies the optimizer's assumption about the effective size of the disk cache that is available to a single query.

Set this parameter based on the following factors: the GaussDB's shared buffer space, the kernel's disk buffer space, and the estimated number of concurrent queries on different tables that share the available space.

This parameter does not affect the size of the shared memory allocated during actual GaussDB running. It is used only for estimation in the plan generation phase. The value is in the unit of disk page. Usually the size of each page is 8192 bytes.

Parameter type: integer.

Unit: page (8 KB)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 1 to 2147483647

Default value:

Independent deployment:

CN: **4GB** (128-core CPU/1024 GB memory); **2GB** (60-core CPU/480 GB memory); **1GB** (32-core CPU/256 GB memory and 16-core CPU/128 GB memory); **512MB** (8-core CPU/64 GB memory); **256MB** (4-core CPU/32 GB memory); **128MB** (4-core CPU/16 GB memory)

DN: **152GB** (128-core CPU/1024 GB memory); **70GB** (60-core CPU/480 GB memory); **38GB** (32-core CPU/256 GB memory); **20GB** (16-core CPU/128 GB memory); **8GB** (8-core CPU/64 GB memory); **4GB** (4-core CPU/32 GB memory); **2GB** (4-core CPU/16 GB memory)

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, the value **200** indicates 200 pages, that is, 200 x 8 KB. The value **200MB** indicates 200 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: A larger value indicates that the optimizer prefers index scan, and a smaller value indicates that the optimizer prefers full table scanning. Generally, the value is half of the value of **shared_buffers**. More radically, you can set the value to three-fourths of the value of **shared_buffers**. If the memory is sufficient, a larger value indicates better performance.

Risks and impacts of improper settings: The optimizer's selection of the generation plan will be affected, and the query performance will be affected in some scenarios. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

allocate_mem_cost

Parameter description: Specifies the query optimizer's estimated cost of creating a hash table for memory space using hash join. This parameter is used for optimization when the hash join estimation is inaccurate.

Parameter type: floating-point number

Unit: none

Value range: 0 to *DBL_MAX*

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Adjust this parameter only when the Hash Join estimation is inaccurate.

Risks and impacts of improper settings: Hash Join estimation may be inaccurate, affecting query performance in some scenarios. Exercise caution when modifying the parameter to avoid risks caused by misoperations.

6.3.8.3 Genetic Query Optimizer

This section describes parameters related to genetic query optimizer. The genetic query optimizer (GEQO) is an algorithm that plans queries by using heuristic

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

searching. This algorithm reduces planning time for complex queries and the costs of producing plans are sometimes inferior to those found by the normal exhaustive-search algorithm.

geqo

Parameter description: Specifies whether to enable the genetic query optimization.

NOTE

If this parameter is modified by running the **gs_guc reload** command and the connection of a session on the current node is not from the client but from another node in the cluster to which the node belongs, this parameter does not take effect immediately on the session after the **gs_guc reload** command is executed. The setting takes effect only after the connection node is disconnected and then reconnected.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Generally, do not disable the parameter during the execution. **geqo_threshold** provides more subtle control of GEQO.

Risks and impacts of improper settings: If the **geqo** parameter is disabled, dynamic planning is still used to enumerate all combinations when a query statement involves a large number of tables. As a result, the optimizer generates a plan with high cost.

geqo_threshold

Parameter description: Specifies the number of **FROM** items. Genetic query optimization is used to plan queries when the number of statements executed is greater than this value.

NOTE

- For simpler queries, it is best to use the regular, exhaustive-search planner; but for queries with many tables, it is better to use GEQO to manage the queries.
- A **FULL OUTER JOIN** construct counts as only one **FROM** item.

Parameter type: integer.

Unit: none

Value range: 2 to 2147483647

Default value: **12**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If the performance overhead of dynamic planning is acceptable, increase the value of this parameter. If the performance overhead of dynamic planning is unacceptable, decrease the value of this parameter.

Risks and impacts of improper settings: If the value is too large, the cost of enumerating all plans in dynamic planning may be unacceptable. If the value is too small, the plan accuracy may be affected.

geqo_effort

Parameter description: Specifies the trade-off between planning time and query plan quality in GEQO.

Parameter type: integer.

Unit: none

Value range: 1 to 10

Default value: **5**

NOTE

- **geqo_effort** does not do anything directly. This parameter is only used to compute the default values for the other variables that influence GEQO behavior. You can also manually set other parameters.
- Larger values increase the time spent in query planning, but also increase the probability that an efficient query plan is chosen.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the query planning cost may be high. If the value is too small, the quality of the generated query plan may be poor.

geqo_pool_size

Parameter description: Specifies the pool size used by GEQO, that is, the number of individuals in the genetic population.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **0**

NOTE

If this parameter is set to **0**, GaussDB selects a proper value based on **geqo_effort** and the number of tables. In other cases, the value of the parameter is at least **2**, and the useful value typically ranges from 100 to 1000.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the query planning cost may be high. If the value is too small, the quality of the generated query plan may be poor.

geqo_generations

Parameter description: Specifies the number of iterations of the GEQO.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **0**

NOTE

If it is set to **0**, a suitable value is chosen based on **geqo_pool_size**. In other cases, the value of this parameter is at least **1**, and useful values are typically from **100** to **1000**.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the query planning cost may be high. If the value is too small, the quality of the generated query plan may be poor.

geqo_selection_bias

Parameter description: Specifies the selection bias used by GEQO. The selection bias is the selective pressure within the population.

Parameter type: floating-point number

Unit: none

Value range: 1.5 to 2

Default value: **2**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the parameter is set to a large value, the GEQO algorithm may focus on some excellent plans more quickly than exploring diversity in feasible plans. This may reduce the possibility that the algorithm finds the global optimal solution, but can improve the speed of finding a feasible better solution.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

geqo_seed

Parameter description: Specifies the initial value of the random number generator used by GEQO to select random paths through the join order search space.

Parameter type: floating-point number

Unit: none

Value range: 0 to 1

Default value: **0**

NOTE

Varying the value changes the set of join paths explored, and may result in a better or worse best path being found.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Different values bring different randomness, which may affect the quality of connection path generation.

6.3.8.4 DSL Rule Engine

enable_dsl_ruleengine

Parameter description: Specifies whether to enable the DSL rule engine. If this function is enabled, the optimizer applies DSL rules to rewrite SQL statements equivalently.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the built-in rules provided by the DSL rule engine cannot be used for query rewriting optimization. As a result, the generated query plan may not be optimal.

dsl_engine_max_iteration

Parameter description: Specifies the maximum number of iterations applied by the DSL rule engine in an SQL rewriting.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: none

Value range: 1 to 100

Default value: **10**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the DSL rule engine cost may be high. If the value is too small, the generated query plan may not be optimal.

dsl_rulelist

Parameter description: Specifies whether to enable or disable a session-level DSL rule. The DSL rule whose status needs to be changed is specified by rule name, and the rule status is indicated by **on** or **off**.

Parameter type: string.

Unit: none

Value range:

- Rule name: The rule name is obtained from the **rulename** column in the GS_DSL_REWRITERULE system catalog.
- Status:
  - **on**: The rule is enabled.
  - **off**: The rule is disabled.

```
gaussdb=# SHOW dsl_rulelist;
 dsl_rulelist
--------------

(1 row)

gaussdb=# SET dsl_rulelist = 'compute_const:on';
SET
gaussdb=# SHOW dsl_rulelist;
    dsl_rulelist
--------------------
 compute_const:on
(1 row)
```

Default value: **''**, indicating that whether the rule is enabled is controlled by the **enabled** column in the GS_DSL_REWRITERULE system catalog.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If you need to manually set this parameter, determine whether the enabled or disabled rule meets the requirements.

Risks and impacts of improper settings: If a rule is set to **off**, the generated query plan may not be optimal.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3.8.5 Other Optimizer Options

cost_model_version

Parameter description: Specifies the version of the optimizer cost model. It can be regarded as a protection parameter to disable the latest optimizer cost model and keep consistent with the plan of the version corresponding to the specified value. If the function is the same as that of other GUC parameters, see the description of the corresponding GUC parameter for the effective priority.

Parameter type: integer.

Unit: none

Value range: **0**, **1**, **2**, **3**, **4**, **5**, **6**, **7**, and **8**

- **0**: The latest cost estimation model is used. In the current version, it is equivalent to **8**.
- **1**: The original cost estimation model is used.
- **2**: The enhanced COALESCE expression, hash join cost, and semi/anti join cost are used for estimation on the basis of **1**.
- **3**: The boundary correction estimator is used to estimate the NDV on the basis of **2**. The hint of index scan can be applied to index-only scan.
- **4**: Partition-level statistics are used for cost estimation on the basis of **3**.
- **5**: The cost estimation of outer join calculation filter is enhanced on the basis of **4**, making the cost-based query rewriting more accurate. The calculation of filter conditions of the outer table in the outer join during selectivity calculation is optimized, which can be controlled by the **enable_poisson_outer_optimization** parameter.
- **6**: The following functions are added on the basis of **5**:
  - When a subquery is used to update multiple columns, multiple subplans can be optimized to only one. This function can also be enabled or disabled using **enable_shared_set_update**.
  - In the query rewriting phase, redundant DISTINCT and GROUP BY conditions in IN non-correlated sublinks and redundant LIMIT conditions in Exists simple sublinks can be eliminated, eliminate redundant LIMIT conditions in expression-related sublinks with agg and redundant projection columns in subqueries.
  - FQS plans can be generated when sublinks meet the distribution key join conditions of inner and outer tables.
  - FQS plans can be generated using INSERT, UPDATE, and DELETE statements in views, in addition, the involved DNs can be tailored based on the simple filter criteria of the distribution key. (For SELECT view statements, the FQS plan is supported in versions earlier than **6**, but DNs cannot be tailored. In versions later than **6**, DNs can also be tailored.)
  - In the query rewriting phase, redundant left join and right join in sublinks can be eliminated, self-joins on primary keys or NOT NULL UNIQUE keys can be eliminated, and outer joins on primary and foreign keys can be converted into inner joins.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - In the query rewriting phase, simple IN/ANY correlated sublinks that meet the conditions can be switched from semi join to inner join in the scenario where semi join can be performed.
  - Static statistics can be used. The number of rows and pages can be directly read from the PG_CLASS system catalog, and the base table sampling is skipped. This function can also be enabled or disabled using **opt_use_static_stats**.
  - Correlated sublinks of IN boosting in the query rewriting phase is supported. When this parameter takes effect, this function is controlled by the **pullup_in_related** option of the GUC parameter **rewrite_rule**.
  - Cost-based expansion of OR conditions into set operations such as UNION ALL or UNION is supported. When this parameter takes effect, this function is controlled separately by the **enable_or_expand** option of the GUC parameter **costbased_rewrite_rule**.
  - When the bitmap index scan plan is generated for the record array in (whether the multi-column composite type expression exists in the given array set) condition, the generated execution plan may not be globally optimal due to too many predicates. After this parameter is enabled, this scenario can be optimized and indexes that cover more columns are more likely to be retained.
- **7**: The following functions are added on the basis of **6**:
  - For equi-join predicates whose left and right operands are ordinary columns, the selectivity cache function can be used in the path generation phase to accelerate the calculation process and improve the optimizer performance. This function can also be controlled by specifying **enable_sel_cache**.
  - Predicates whose left and right operands are ordinary columns or predicates whose left and right operands are ordinary columns and constant values and are connected by operators such as =, <>, >, >=, <, and <= are supported. These predicates are only placed after WHERE and do not contain subquery lifting and join pushdown. Predicate deduction and reduction can be performed based on deduction rules to simplify predicate logic and improve query performance. This function can be controlled separately by the **PREDICATE_INFER_ENHANCE** option of **opt_behavior_knob**.
  - In M-compatible database, a parallel execution plan can be generated for count(DISTINCT *Multiple columns*) to improve query performance.
  - Redundant ORDER BY elimination is enhanced to improve query performance.
  - In ORA-compatible mode, the FQS can be pushed down to DNs for execution in recursive queries using START WITH CONNECT BY on replication tables.
  - The DSL rule engine can be used in the query rewriting phase to improve query performance. By default, the engine contains query rewriting rules in four scenarios. For details about the rules, see "Table Built-in rules" in "Feature Usage Guidelines > DSL Rule Engine > Overall Process of the DSL Rule Engine" in *Developer Guide*.
- **8**: The following functions are added on the basis of **7**:
  - In some scenarios (xmlagg does not contain the ORDER BY clause, xmlagg contains the ORDER BY clause and the GROUP BY statement

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

exists in the same-layer query, and xmlagg is located inside a correlated subquery), the aggregate function xmlagg can be pushed down to DNs for execution, improving query performance.

- DSL query rewriting rules can be added in version 507.0.0 to improve query performance. For details about the new rules, see the corresponding content of scenarios 5 to 9 under "Table Built-in rules" in "Feature Usage Guidelines > DSL Rule Engine > Overall Process of the DSL Rule Engine" in *Developer Guide*.
- Partition-wise join is supported in more scenarios to optimize query performance. For details about the new capabilities, see "Feature Usage Guidelines > Partitioned Table > Query Optimizations on Partitioned Tables > Partition-wise Join" in *Developer Guide*.

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: When upgrading the database, you are advised to set this parameter the same as that of the source version. When installing a new environment, you are advised to set this parameter to the default value.

Risks and impacts of improper settings: If this parameter is modified, many SQL plans may be changed. Therefore, exercise caution when changing this parameter.

enable_fast_query_shipping

Parameter description: Specifies whether to use the distributed framework for a query planner.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the distributed framework is not used. Execution plans are generated on CNs and DNs separately.
- **off** indicates that the distributed framework is used. Execution plans are generated on CNs and then sent to DNs for execution.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the generated plan may not use the distributed cluster performance more efficiently. You are advised to enable this parameter.

enable_trigger_shipping

Parameter description: Specifies whether the trigger can be pushed to DNs for execution.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the trigger can be pushed to DNs for execution.
- **off** indicates that the trigger cannot be pushed to DNs. It must be executed on CNs.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the performance load is concentrated on the CN, and the cluster performance cannot be maximized. You are advised to enable this parameter.

enable_remotejoin

Parameter description: Specifies whether JOIN operation plans can be pushed to DNs for execution.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: JOIN operation plans can be pushed to DNs for execution.
- **off**: JOIN operation plans cannot be pushed to DNs for execution.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If there is a large amount of data in the JOIN table, the data has been distributed on different DNs based on the JOIN condition, and the DNs have sufficient compute resources to efficiently complete the JOIN operation locally, disabling this parameter may deteriorate the query performance.

enable_remotegroup

Parameter description: Specifies whether the execution plans of GROUP BY and AGGREGATE can be pushed to DNs for execution.

Parameter type: Boolean.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: The execution plans of GROUP BY and AGGREGATE can be pushed to DNs for execution.
- **off**: The execution plans of GROUP BY and AGGREGATE cannot be pushed to DNs for execution.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the query involves grouping of a large amount of data and the data is distributed on multiple data nodes, disabling this parameter may deteriorate the query performance.

enable_remotelimit

Parameter description: Specifies whether the execution plan specified in the LIMIT clause can be pushed to DNs for execution.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The execution plan specified in the LIMIT clause can be pushed down to DNs for execution.
- **off**: The execution plan specified in the LIMIT clause cannot be pushed to DNs for execution.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the query involves quick tailoring of a large amount of data and data is distributed on multiple nodes, disabling this parameter may cause query performance deterioration.

enable_remotesort

Parameter description: Specifies whether the execution plan of the ORDER BY clause can be pushed to DNs for execution.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The execution plan of the ORDER BY clause can be pushed to DNs for execution.
- **off**: The execution plan of the ORDER BY clause cannot be pushed to DNs for execution.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the query involves sorting of a large amount of data and the data is distributed on multiple data nodes, disabling this parameter may cause performance deterioration.

enable_rep_table_strict_shipping

Parameter description: Specifies the behavior when the pushdown necessary and sufficient conditions of a replication table fail to be determined during DML operations on the replication table.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: When DML operations are performed on replication tables, the tables cannot be pushed down if the pushdown conditions fail to be determined.
- **off**: When DML operations are performed on replication tables, the pushdown is not disabled and only prompt information is printed in logs if the pushdown conditions of replication tables fail to be determined.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_csqual_pushdown

Parameter description: Specifies whether to push down filter criteria for a rough check during query.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: A rough check is performed with filter criteria pushed down during query.
- **off**: A rough check is performed without filter criteria pushed down during query.

Default value: **on**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If a query involves filtering of a large amount of data and the data is distributed on different nodes, disabling pushdown of filter criteria may cause performance deterioration.

explain_dna_file

Parameter description: Specifies the object file to be exported when **explain_perf_mode** is set to **run**.

Parameter type: string.

Unit: none

Value range: absolute path plus the file name in .csv format.

Default value: **""**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Set this parameter based on the service scenario if necessary.

Risks and impacts of improper settings: Improper settings may cause file write overhead.

explain_perf_mode

Parameter description: Specifies the display format of the **explain** command.

Parameter type: enumerated type

Unit: none

Value range: **normal**, **pretty**, **summary**, and **run**

- **normal** indicates that the default printing format is used.
- **pretty** indicates a new format improved by using GaussDB. The new format contains a plan node ID, directly and effectively analyzing performance.
- **summary** indicates that the analysis result on this information is printed in addition to the printed information specified by **pretty**.
- **run** indicates that the system exports the printed information specified by **summary** as a CSV file for further analysis.

Default value: **pretty**

NOTE

The pretty mode supports only plans that contain stream operators and does not support plans that deliver statements to DNs. Therefore, the display format is affected by the **enable_stream_operator** parameter. When **enable_stream_operator** is set to **off**, the plan containing the stream operator cannot be generated.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value. You can set another value if necessary.

Risks and impacts of improper settings: Different formats affect the amount of information provided by EXPLAIN.

analysis_options

Parameter description: Specifies whether to enable function options in the corresponding options to use the corresponding location functions, including data verification and performance statistics. For details, see the description of the value range.

Parameter type: string.

Unit: none

Value range:

Each time this parameter is set, the current value is changed through the set operation.

```
gaussdb=# show analysis_options;
                    analysis_options
------------------------------------------------------------
 ALL,on(),off(LLVM_COMPILE,HASH_CONFLICT,STREAM_DATA_CHECK)
(1 row)

gaussdb=# SET analysis_options = 'on(LLVM_COMPILE)';
SET
gaussdb=# show analysis_options;
                    analysis_options
------------------------------------------------------------
 ALL,on(LLVM_COMPILE),off(HASH_CONFLICT,STREAM_DATA_CHECK)
(1 row)

gaussdb=# SET analysis_options = 'on(HASH_CONFLICT)';
SET
gaussdb=# show analysis_options;
                    analysis_options
------------------------------------------------------------
 ALL,on(LLVM_COMPILE,HASH_CONFLICT),off(STREAM_DATA_CHECK)
(1 row)
gaussdb=# SET analysis_options = 'off(ALL)';
SET
gaussdb=# show analysis_options;
                    analysis_options
------------------------------------------------------------
 ALL,on(),off(LLVM_COMPILE,HASH_CONFLICT,STREAM_DATA_CHECK)
(1 row)
```

- **LLVM_COMPILE**: The codegen compilation time of each thread is displayed on the explain performance page.
- **HASH_CONFLICT**: The log file in the **gs_log** directory of the DN process displays the hash table statistics, including the hash table size, hash chain length, and hash conflict information.
- **STREAM_DATA_CHECK**: A CRC check is performed on data before and after network data transmission.

Default value: **off(ALL)**, which indicates that no location function is enabled.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If manual setting is required, check whether the expected result meets the requirements.

cost_param

Parameter description: Specifies use of different estimation methods in specific customer scenarios, allowing the cost model estimation more accurate. You can change the parameter values to enable different methods. This parameter can control multiple methods at the same time. Each method is controlled by a number. If the result of the AND operation between the parameter value and the number corresponding to the method is not 0, the method is enabled.

- When **cost_param & 1** is set to a value other than 0, an improved mechanism is used for connecting the selectivity of non-equi-joins. This method is more accurate for estimating the selectivity of joins between two identical tables. At present, if **cost_param & 1** is set to a value other than 0, the path is not used. That is, a better formula is selected for calculation.
- When **cost_param & 2** is set to a value other than 0, the selectivity is estimated based on multiple filter criteria. The lowest selectivity among all filter criteria, but not the product of the selectivities for two tables under a specific filter criterion, is used as the total selectivity. This method is more accurate when a close join exists between the columns to be filtered.
- When **cost_param & 4** is not 0, the selected debugging model is not recommended when the stream node is evaluated.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter after determining the scenario where the selectivity needs to be adjusted.

Risks and impacts of improper settings: The estimated cost may not meet the expectation. You are advised to confirm the parameter settings after thorough tests.

enable_partitionwise

Parameter description: Specifies whether to select an intelligent algorithm for joining partitioned tables.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that an intelligent algorithm is selected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **off** indicates that an intelligent algorithm is not selected.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Before enabling this function, ensure that SMP is enabled. Otherwise, the performance of the join operation on partitioned tables in non-SMP scenarios may be affected.

partition_page_estimation

Parameter description: Specifies whether to optimize the estimation of partitioned table pages based on the pruning result.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The pruning result is used to optimize the page estimation.
- **off**: The pruning result is not used to optimize the page estimation.

Default value:

- **on**: default value for a newly installed database.
- **off**: For an upgrade from a version earlier than 506.0.0 to the current version, the default value of this parameter is **off**. If this parameter exists before the upgrade, the default value is **off** after the upgrade regardless of its setting.

  In the PDB scenario, if this parameter is not set, the global settings will apply.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_partition_pseudo_predicate

Parameter description: Specifies whether to rewrite pseudo-predicates to calculate the selectivity of query in a specified partition.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that pseudo-predicate rewriting is used.
- **off** indicates that pseudo-predicate rewriting is not used.

Default value: **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

partition_iterator_elimination

Parameter description: Specifies whether to eliminate the partition iteration operator to improve execution efficiency when the partition pruning result of a partitioned table is a partition.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The partition iteration operator is eliminated.
- **off**: The partition iteration operator is not eliminated.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the query performance may deteriorate.

rewrite_rule

Parameter description: Specifies the optional query rewriting rules that are enabled. Some query rewriting rules are optional. Enabling them cannot always improve the query efficiency. In a specific customer scenario, you can set the query rewriting rules through this GUC parameter to achieve optimal query efficiency.

This parameter specifies the combination of query rewriting rules. For example, if there are multiple rewriting rules **rule1**, **rule2**, **rule3**, and **rule4**, you can perform settings as follows:

```
set rewrite_rule=rule1;          -- Enable query rewriting rule rule1
set rewrite_rule=rule2, rule3;     -- Enable query rewriting rules rule2 and rule3
set rewrite_rule=none;         -- Disable all optional query rewriting rules
```

Parameter type: enumerated type

Unit: none

Value range:

- **none**: No optional query rewriting rules are used.
- **lazyagg**: The Lazy Agg query rewriting rules are used to eliminate aggregation operations in subqueries.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **magicset**: The Magic Set query rewriting rules are used to join subqueries which have aggregation operators with the main query in advance to reduce repeated scanning of sublinks.
- **partialpush**: The Partial Push query rewriting rules are used. For statements that cannot be pushed down, push down some subqueries to DNs for execution and the rest to CNs for execution.
- **uniquecheck**: The Unique Check query rewriting rules are used to optimize the subquery statements in target columns without agg and check whether the number of returned rows is 1.
- **disablerep**: The Disable Replicate query rewriting rules are used. The performance may deteriorate after a replication table is optimized. Therefore, after this rule is enabled, subqueries cannot be optimized.
- **intargetlist**: The In Target List query rewriting rules are used to improve subqueries in the target column.
- **predpushnormal**: The predicate push query rewriting rule is used. (The predicate condition is pushed down to the subquery, and the BROADCAST operator may be added to support distributed execution.) After **predpush** is enabled, no vectorized plan is generated.
- **predpushforce**: The predicate push query rewriting rules are used to push down predicate conditions to subqueries and use indexes to accelerate the query. After predpush is enabled, no vectorized plan is generated.
- **predpush**: The optimal plan between **predpushnormal** and **predpushforce** is selected based on the cost. After **predpush** is enabled, no vectorized plan is generated.
- **disable_pullup_expr_sublink**: The optimizer is not allowed to pull up sublinks of the expr_sublink type. For details about sublink classification and pull-up principles, see "Typical SQL Optimization Methods > Optimizing Subqueries" in *SQL Optimization Guide*.
- **enable_sublink_pullup_enhanced**: Enhanced sublink query rewriting rules are used, including unrelated sublink pull-up of the WHERE and HAVING clauses and WinMagic rewriting optimization.
- **disable_pullup_not_in_sublink**: The optimizer is not allowed to pull up sublinks related to NOT IN. For details about sublink classification and pull-up principles, see "Typical SQL Optimization Methods > Optimizing Subqueries" in *SQL Optimization Guide*.
- **disable_rownum_pushdown**: The filter criterion ROWNUM in the parent query cannot be pushed down to the subquery.
- **disable_windowagg_pushdown**: The filter criterion of the window function in the parent query cannot be pushed down to the subquery.
- **cse_rewrite_opt**: The common expression CSE is used to replace the common expression in the Having subquery with windowAgg.
- **groupby_pushdown_subquery**: The Group By clause and aggregate function are pushed down to the subquery.
- **enable_sublink_pullup_rownum**: The optimizer can be used to pull up sublinks when the SQL statement contains the ROWNUM pseudocolumn.
- **disable_in_sublink_reduce_distinct**: Redundant DISTINCT and GROUP BY conditions cannot be eliminated in IN/ANY non-correlated sublink pullup scenarios.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **disable_sublink_reduce_limit**: Redundant LIMIT conditions (for example, LIMIT conditions that do not affect query results in simple Exists sublinks) in sublinks that meet constraints cannot be eliminated.
- **pullup_in_related**: Simple IN/ANY or NOT IN sublinks that meet the conditions can be promoted to semi join or anti join.
- **disable_reduce_unused_targetlist**: Redundant function calculation in the subquery projection column cannot be eliminated.
- **disable_sublink_outer_join_elimination**: Redundant left/right joins in sublinks that meet the conditions cannot be eliminated.
- **disable_fk_pk_outer_join_to_inner_join**: An outer join on a primary foreign key cannot be converted to an inner join.
- **disable_self_join_elimination**: The self-joins on primary keys or NOT NULL UNIQUE keys cannot be eliminated.
- **enable_corr_any_sublink_semijoin_to_innerjoin**: In the query rewriting phase, simple IN/ANY correlated sublinks that meet the conditions can be switched from semi join to inner join in the scenario where semi join can be performed.
- **disable_reduce_orderby**: Redundant ORDER BY statements that meet the conditions are not eliminated.
- **enable_not_in_sublink_reduce_distinct**: Redundant DISTINCT and GROUP BY conditions can be eliminated when non-correlated sublinks of NOT IN type are pulled up.

Default value: **magicset,groupby_pushdown_subquery,pullup_in_related**.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If other rewriting is necessary, you are advised to specify the query rewriting scenario and enable only the corresponding rewriting rule.

Risks and impacts of improper settings: The cost of rewriting statements in some scenarios may not be optimal. You are advised to use the corresponding rules after thorough tests.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When **cost_model_version** is greater than or equal to 6 or is set to **0** and **disable_in_sublink_reduce_distinct** is not set in GUC parameters, the function of eliminating redundant DISTINCT and GROUP BY for IN non-correlated sublinks takes effect.
- When **cost_model_version** is greater than or equal to 6 or is set to **0** and **disable_sublink_reduce_limit** is not set in GUC parameters, the function of eliminating redundant LIMIT for Exists sublinks takes effect.
- When **cost_model_version** is greater than or equal to 6 or is set to **0** and **disable_reduce_unused_targetlist** is not set in GUC parameters, the function of eliminating redundant projection columns in a subquery takes effect.
- When **cost_model_version** is greater than or equal to 6 or is set to **0** and **pullup_in_related** is set in GUC parameters, the IN-correlated sublink improvement function takes effect.
- When **cost_model_version** is greater than or equal to 6 or is set to **0** and **disable_sublink_outer_join_elimination** is not set in GUC parameters, the function of eliminating redundant left/right joins in the sublink takes effect.
- When **cost_model_version** is greater than or equal to 6 or is set to **0** and **disable_fk_pk_outer_join_to_inner_join** is not set in GUC parameters, the function of converting outer joins to inner joins on primary and foreign keys takes effect.
- When **cost_model_version** is greater than or equal to 6 or is set to **0** and **disable_self_join_elimination** is not set in the GUC parameter, the function of eliminating self-join on the primary key or NOT NULL UNIQUE key takes effect.
- When **cost_model_version** is greater than or equal to 6 or is set to **0** and **enable_corr_any_sublink_semijoin_to_innerjoin** is set in GUC parameters, the switching of simple IN/ANY-correlated sublinks from semi join to inner join takes effect.
- When **cost_model_version** is greater than or equal to 6 or is set to **0** and **enable_not_in_sublink_reduce_distinct** is set in GUC parameters, the function of eliminating redundant DISTINCT and GROUP BY for NOT IN non-correlated sublinks takes effect.

costbased_rewrite_rule

Parameter description: Specifies whether to enable the cost-based evaluation policy for a specified query rewriting rule. Some query rewriting rules support cost-based evaluation of whether to rewrite SQL statements. In this way, the kernel can generate better query plans and improve SQL execution efficiency.

This parameter is mutually exclusive with the GUC parameter **rewrite_rule**. If both **costbased_rewrite_rule** and **rewrite_rule** are set, the rule configuration related to **rewrite_rule** does not take effect.

This parameter can be used to set multiple cost-based query rewriting rules. The parameters are not mutually exclusive. For example, if there are multiple rewriting rules **rule1**, **rule2**, **rule3**, and **rule4**, you can perform settings as follows:

```
set costbased_rewrite_rule= rule1;          -- Enable the cost-based evaluation policy of query rewriting rule rule1.
set costbased_rewrite_rule= rule2,rule3;    -- Enable the cost-based evaluation policy for query rewriting rules rule2 and rule3.
set costbased_rewrite_rule=none;            -- Disable the cost-based evaluation policy for all optional query rewriting rules.
```

Parameter type: string.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **none**: No cost-based query rewriting policy is used.
- **pullup_subquery**: The cost-based query rewriting policy is enabled for the rewriting rules of simple subquery expansion.
- **pullup_sublink_any_exists**: The cost-based query rewriting policy is enabled for the rewriting rules of non-correlated sublinks of the ANY type and related sublinks of the [NOT] EXISTS type in a single or AND condition.
- **pullup_not_in_sublink**: The cost-based query rewriting policy is enabled for the rewriting rules of non-correlated sublinks of the NOT IN type in a single or AND condition. This option is mutually exclusive with the **disable_pullup_not_in_sublink** option of the GUC parameter **rewrite_rule**. When this option is enabled, the **disable_pullup_not_in_sublink** option of **rewrite_rule** does not take effect.
- **pullup_expr_sublink**: The rewriting scenarios of expression sublinks, non-correlated sublinks of the ANY type, and related sublinks of [NOT] the EXISTS type in the OR condition take effect. This option is mutually exclusive with the **disable_pullup_expr_sublink**, **enable_sublink_pullup_enhanced**, and **magicset** options of the GUC parameter **rewrite_rule**. When this option is enabled, the corresponding options of **rewrite_rule** do not take effect.
- **intargetlist**: The cost-based query rewriting policy is enabled for the rewriting rules of the sublinks of related expression in the TargetList. This option is mutually exclusive with the **intargetlist** and **magicset** options of the GUC parameter **rewrite_rule**. When this option is enabled, the **intargetlist** and **magicset** options of **rewrite_rule** does not take effect.
- **enable_sublink_pullup_enhanced**: The cost-based query rewriting policy is enabled for the rewriting rules of expression sublinks in the enhanced scenario. This option is affected by the **pullup_expr_sublink** option. This option takes effect only when the **pullup_expr_sublink** option is enabled for the expression sublink rewriting scenario in the AND condition. It is mutually exclusive with the **enable_sublink_pullup_enhanced** option of the GUC parameter **rewrite_rule**. When this option is enabled, the **enable_sublink_pullup_enhanced** option of **rewrite_rule** does not take effect.
- **enable_or_expand**: The OR condition is expanded based on the cost and rewritten into a set operation such as UNION ALL or UNION.

Default value:

- **intargetlist,pullup_expr_sublink,pullup_not_in_sublink,enable_sublink_pullup_enhanced,enable_or_expand** for a newly installed database.
- **none** for a database upgraded from a version earlier than 505.2.0 to the current version, or the same value as that before for upgrade from 505.2.0 to a later version (for example, 505.2.1), as this parameter is available since 505.2.0.

NOTE

- For details about parameter application scenarios, see "Optimization Cases > Case: Adjusting the GUC Parameter costbased_rewrite_rule for Cost-based Query Rewriting" in *SQL Optimization Guide*.
- When **cost_model_version** is greater than or equal to 6 or is set to **0** and **enable_or_expand** is set in GUC parameters, the functions expanded based on costs by OR conditions take effect.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Simple subquery expansion (specified by the **pullup_subquery** parameter), ANY non-correlated sublinks, and [NOT] EXISTS related sublinks (specified by the **pullup_sublink_any_exists** parameter) are rewritten based on the default rules to generate an execution plan. Enabling the cost rewriting policy brings some performance overhead, therefore, you are advised not to enable the cost-based rewriting policy for such rules.

In the query rewriting phase of the optimizer, after this parameter is enabled, more candidate plans are generated compared with the rule-based rewriting model. Therefore, extra time is consumed. In this mode, each time the cost-based rewriting policy is used, the plan generation time is doubled compared with the rule-based rewriting policy, but the execution performance may be better. Therefore, if the SQL statement uses a good execution plan generated based on the rule-based rewriting policy, you are advised to disable this parameter.

Risks and impacts of improper settings: If the rewriting rules related to this parameter are disabled, rewriting is performed based on rules. If the cost evaluation is missing, the generated plans in some scenarios may be poor, affecting the query performance.

costbased_rewrite_rule_max_iterations

Parameter description: In the plan generation phase of an SQL statement, if the number of conditions that meet the cost evaluation in the same rule exceeds the value of this parameter, the cost-based evaluation policy is disabled for the conditions that exceed the threshold in the current request and converted to the rule-based rewriting policy. This parameter takes effect when the cost-based evaluation policy is enabled in the query rewriting phase.

Parameter type: integer.

Unit: none

Value range: 0 to 1000

Default value: **10**

NOTE

- This parameter is affected by the GUC parameter **costbased_rewrite_rule**. This parameter takes effect when the value of **costbased_rewrite_rule** is a value other than **none**.
- If this parameter is set to **0**, the cost-based rewriting policy is disabled for the current SQL statement.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Adjust the value based on service requirements. If the value is too large or too small, extra performance overhead may occur, affecting the final query performance.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

costbased_rewrite_rule_timeout

Parameter description: In the plan generation phase of an executed SQL statement, if the overall cost evaluation time using each rule exceeds the timeout interval specified by this parameter, the cost-based evaluation policy is disabled for subsequent processes of the current request and the rule-based rewriting policy is used. This parameter takes effect when the cost-based evaluation policy is enabled in the query rewriting phase.

Parameter type: integer.

Unit: milliseconds

Value range: –1 to 300000

Default value: **-1**

NOTE

- This parameter is affected by the GUC parameter **costbased_rewrite_rule**. This parameter takes effect when the value of **costbased_rewrite_rule** is a value other than **none**.
- If this parameter is set to **0**, the cost-based rewriting policy is disabled for the current SQL statement.
- If this parameter is set to **-1**, the timeout interval control is disabled for the current SQL statement.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Adjust the value based on service requirements. If the value is too large or too small, extra performance overhead may occur, affecting the final query performance.

optrbo_max_or_expr_numbers

Parameter description: Specifies the maximum number of times that OR can be used for expansion.

Parameter type: integer.

Unit: none

Value range: 2 to 100

Default value: **10**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Set this parameter based on service requirements. If the value is too large, queries with many OR conditions may be rewritten, affecting the query performance.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_pbe_optimization

Parameter description: Specifies whether the optimizer optimizes the query plan for statements executed in parse bind execute (PBE) mode. The optimization principle is to use gplan for FQS.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The optimizer optimizes the query plan for statements executed in PBE mode and the gplan is used for FQS.
- **off** indicates that the optimization is not performed.

Default value: **on**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If you do not use gplan for FQS, set this parameter to **off**.

Risks and impacts of improper settings: If this parameter is disabled, the cplan may be used in some scenarios.

enable_light_proxy

Parameter description: Specifies whether the optimizer optimizes the execution of simple queries on CNs. This parameter does not take effect if the character set of the application side does not match that of the kernel side. You are advised to set the character set to UTF-8 when creating a database.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the optimizer optimizes the execution of simple queries on CNs.
- **off** indicates that the optimization is not performed.

Default value: **on**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the execution pressure of simple queries may be concentrated on the CN.

NOTE

This parameter does not take effect if it is set using SET in a stored procedure.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_global_plancache

Parameter description: Specifies whether to share the cache for the execution plans of statements in PBE queries and stored procedures. If this parameter is set to **on**, the memory usage of the CNs and DNs in high concurrency scenarios can be reduced.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that cache sharing is enabled for the execution plans of statements in PBE queries and stored procedures.
- **off** indicates no sharing.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. The value of this parameter must be the same on CNs and DNs.

Risks and impacts of improper settings: If this parameter is set to different values on CNs and DNs, the packets delivered by CNs to DNs do not match. As a result, errors are reported. After this parameter is enabled, frequent DDL operations may cause plan invalidation, affecting the CPU overhead and performance.

enable_plancache_mprotect

Parameter description: Specifies whether to enable plan write protection in the execution phase.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Plan write protection is enabled in the execution phase.
- **off**: Plan write protection is disabled in the execution phase.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter applies to the debugging version and is mainly used for fault locating, troubleshooting, and debugging. In the production environment, use the default value to avoid affecting normal services.

Risks and impacts of improper settings:

- If this parameter is enabled, the database kernel strictly protects the generated execution plan and forbids any modifications to the plan in the execution phase.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- This parameter uses mprotect for memory protection. If this parameter is enabled, extra time is required when the memory is frequently used or modified in a large scale under heavy load.
- If the mprotect resources are insufficient due to insufficient memory or the execution plan is modified, the process will be restarted for protection.

disable_gpc_store

Parameter description: Specifies whether to prevent subsequent execution plans from being stored in the global plan cache when **enable_global_plancache** is set to **on**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The subsequent execution plan cannot be stored in the global plan cache.
- **off**: The subsequent execution plan can be stored in the global plan cache.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If **enable_global_plancache** is set to **off**, this setting of this GUC parameter does not take effect. After this parameter is set, subsequent execution plans are not stored in the global shared cache, which may increase the memory overhead of the current session. Exercise caution when setting this parameter.

enable_stream_pbe

Parameter description: Specifies whether the optimizer uses cache to optimize the STREAM distributed plan on DNs.

NOTE

- When **enable_global_plancache** is set to **on**, stream PBE optimization is not supported.
- The optimization functionality of **enable_stream_pbe** takes effect only when the **enable_stream_operator** parameter is enabled.
- If a version that does not support stream PBE is upgraded to a version that supports stream PBE, the stream PBE optimization execution mode is not supported during the upgrade that has not been committed.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the optimizer uses the cache to optimize the STREAM distributed plan on DNs.
- **off** indicates that the optimization is not performed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **on**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the cost of generating a plan may increase.

gpc_clean_timeout

Parameter description: Specifies the retention period of a shared plan that is not used. When **enable_global_plancache** is enabled, if a plan in the shared plan list is not used within the period specified by **gpc_clean_timeout**, the plan is cleared at a specified interval.

Parameter type: integer.

Unit: seconds

Value range: 300 to 86400

Default value: **1800**, that is, 30 minutes

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the GPC may occupy too much memory. You are advised to set this parameter to a proper value after thorough tests.

enable_gpc_grayrelease_mode

Parameter description: Specifies whether to enable GPC in a distributed cluster. The cluster needs to be restarted to enable GPC. If you want to enable GPC without restarting the cluster, use **enable_gpc_grayrelease_mode**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

Perform operations in a distributed cluster.

To enable GPC:

1. Enable **enable_gpc_grayrelease_mode** on all DNs.
2. Enable **enable_gpc_grayrelease_mode** on all CNs.
3. Enable the GPC parameter which is a POSTMASTER parameter. You need to reload the parameter and then kill the node in polling mode for GPC on the restarted node to take effect.

To disable GPC:

1. Ensure that **enable_gpc_grayrelease_mode** is set to **on**, reload and then disable the GPC parameter, and kill the node in polling mode for GPC on the restarted node to take effect.
2. Disable **enable_gpc_grayrelease_mode** on all CNs.
3. Disable **enable_gpc_grayrelease_mode** on all DNs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: GPC is enabled in scrolling mode.
- **off**: GPC is disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is enabled, the logic for sending and receiving PBE packets is affected. Therefore, you must set this parameter in strict accordance with the parameter description to avoid improper errors.

enable_opfusion

Parameter description: Specifies whether to optimize simple queries.

NOTE

This parameter is used to optimize the query performance of DNs. You can set **max_datanode_for_plan** to view the execution plan of a query DN. If the execution plan of the DN contains **[Bypass]**, the query can be optimized on the DN.

The limitations on simple queries are as follows:

- Only index scan and index-only scan are supported, and the filter criteria of all WHERE statements are on indexes.
- Only single tables can be added, deleted, modified, and queried. JOIN and USING operations are not supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Only row-store tables are supported. Partitioned tables and tables with triggers are not supported.
- Information statistics features of active SQL statements and queries per second (QPS) are not supported.
- Tables that are being scaled out or in are not supported.
- System columns cannot be queried or modified.
- Only simple SELECT statements are supported. For example:
  ```
  SELECT c3 FROM t1 WHERE c1 = ? and c2 =10;
  ```
  Only columns in the target table can be queried. Columns **c1** and **c2** are index columns, which can be followed by constants or parameters. You can use **for update**.
- Only simple INSERT statements are supported. For example:
  ```
  INSERT INTO t1 VALUES (?,10,?);
  ```
  Only one VALUES is supported. The type in VALUES can be a constant or a parameter. RETURNING is not supported.
- Only simple DELETE statements are supported. For example:
  ```
  DELETE FROM t1 WHERE c1 = ? and c2 = 10;
  ```
  Columns **c1** and **c2** are index columns, which can be followed by constants or parameters.
- Only simple UPDATE statements are supported. For example:
  ```
  UPDATE t1 SET c3 = c3+? WHERE c1 = ? and c2 = 10;
  ```
  The values modified in column **c3** can be constants, parameters, or a simple expression. Columns **c1** and **c2** are index columns, which can be followed by constants or parameters.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this parameter is disabled, the SQL bypass feature will not be used. SQL bypass uses simplified execution paths for simple CRUD statements to reduce the execution overhead caused by condition filtering, projection column calculation, and operator initialization and execution. After this parameter is disabled, the simple CRUD statements specified in the parameter description will use the common execution process, which affects the performance in the corresponding scenarios.

sql_beta_feature

Parameter description: Specifies the SQL engine's optional beta features to be enabled, including optimization of row count estimation and query equivalence

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

estimation. These optional features provide optimization for specific scenarios, but performance deterioration may occur in some scenarios for which testing is not performed. In a specific customer scenario, you can set the query rewriting rules through this GUC parameter to achieve optimal query efficiency.

This parameter determines the combination of the SQL engine's beta features, for example, feature1, feature2, feature3, and feature4, as follows:

```
set sql_beta_feature=feature1;          -- Enable the beta feature 1 of the SQL engine.
set sql_beta_feature=feature2,feature3;    -- Enable the beta features 2 and 3 of the SQL engine.
set sql_beta_feature=none;          -- Disable all optional SQL engine beta features.
```

Parameter type: string.

Unit: none

Value range:

- **none**: uses none of the beta optimizer features.
- **sel_semi_poisson**: uses Poisson distribution to calibrate the equivalent semi-join and anti-join selectivity.
- **sel_expr_instr**: uses the matching row count estimation method to provide more accurate estimation for **instr(col, 'const') > 0, = 0, = 1**.
- **param_path_gen**: generates more possible parameterized paths.
- **rand_cost_opt**: optimizes the random read cost of tables that have a small amount of data.
- **param_path_opt**: uses the bloating ratio of the table to optimize the analysis information of indexes.
- **page_est_opt**: optimizes the **relpages** estimation for the analysis information of table indexes.
- **no_unique_index_first**: disables optimization of the primary key index scan path first.
- **join_sel_with_cast_func**: supports type conversion functions when the number of join rows is estimated.
- **canonical_pathkey**: The regular path key is generated in advance (**pathkey**: a set of ordered key values of data).

  NOTE

  After this option is enabled, the semantics of the output data of statements such as ORDER BY may be different from that of the standard ones in the case of outer join. Contact Huawei technical support to determine whether to enable this parameter.
- **index_cost_with_leaf_pages_only**: specifies whether index leaf nodes are included when the index cost is estimated.
- **partition_opfusion**: supports lightweight cache of some data structures of partitioned tables during database running and lightweight processing. It does not take effect in the current version.
- **a_style_coerce**: enables the Decode type conversion rule to be compatible with the Oracle Database. For details, see "Type Resolution for CASE in ORA-compatible Mode" in **UNION, CASE, and Related Constructs**.
- **plpgsql_stream_fetchall**: enables the function of obtaining all tuple results when the SQL statements which use streams are executed on the for loop or cursor in a stored procedure.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **partition_fdw_on**: supports the creation of SQL statements related to GaussDB foreign tables based on partitioned tables.
- **predpush_same_level**: enables the **predpush** hint to control parameterized paths at the same layer.
- **disable_bitmap_cost_with_lossy_pages**: disables the computation of the cost of lossy pages in the bitmap path cost.
- **enable_upsert_execute_gplan**: allows execution through **gplan** in the PBE scenario, if the UPDATE clause in the ON DUPLICATE KEY UPDATE statement contains parameters.
- **disable_text_expr_flatten**: disables the function of automatically inlining expressions during comparison between text and numeric types (numeric, bigint).

Default value:
"sel_semi_poisson,sel_expr_instr,rand_cost_opt,param_path_opt,page_est_opt"

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. This parameter specifies multiple query optimization and compatibility behaviors. The settings of some options are risky. Therefore, you are advised to use this parameter with caution. If you want to adjust the value, make sure you understand the parameter meaning and adjust the value with caution to avoid risks caused by misoperations.

Risks and impacts of improper settings: Sufficient tests are required. Otherwise, unexpected risks may occur.

enable_global_stats

Parameter description: Specifies the current statistics mode.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The global statistics collection mode is used.
- **off**: The statistics of the first node in the cluster are collected. In this case, the quality of the generated query plan may be affected. However, the information collection performance is optimal. Therefore, exercise caution when disabling this parameter.

Default value: **on**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the query plan generation quality in the cluster environment may be affected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

default_statistics_target

Parameter description: Specifies the default statistics target for table columns without a column-specific target set by running **ALTER TABLE SET STATISTICS**. This parameter affects only the target number of sampled rows in the statistics. The actual number of sampled rows is also affected by the memory parameter **maintenance_work_mem**.

Parameter type: integer.

Unit: none

Value range: –100 to 10000

- If this parameter is set to a positive value, it indicates the expected number of buckets in the statistics histogram. The number of sampled rows is **default_statistics_target** multiplied by 300.
- If this parameter is set to a negative value, it indicates the statistical target in percentage. The negative value is converted to the corresponding percentage. For example, **-5** indicates 5%. The number of sampled rows is the total number of rows multiplied by 5%.

Default value: **100**

NOTE

- A larger positive number than the default value increases the time required to do **ANALYZE**, but might improve the quality of the optimizer's estimates.
- Changing settings of this parameter may result in performance deterioration. If query performance deteriorates, you can:
  1. Restore to the default statistics.
  2. Use hints to force the optimizer to use the optimal query plan. For details, see "Hint-based Optimization" in *SQL Optimization Guide*.
- If this GUC parameter is set to a negative value, the number of samples is greater than or equal to 2% of the total data volume, and the number of records in user tables is less than 1.6 million, the time taken by running **ANALYZE** will be longer than that when this parameter uses its default value.
- If this GUC parameter is set to a negative value, AUTOANALYZE does not take effect.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Before changing settings of this parameter, perform a comprehensive evaluation based on the specific workload and query mode. For tables whose data features are not evenly distributed or whose data volume is large, you may need to manually adjust the value of this parameter.

Risks and impacts of improper settings:

- If the value of **default_statistics_target** is too large, the time and resource consumption of the ANALYZE operation may increase because more sample data needs to be collected to generate statistics. This can result in increased overhead for database maintenance, especially on large tables.
- If the value of **default_statistics_target** is too small, the accuracy of statistics may be reduced. As a result, the query optimizer's capability of generating efficient query plans is affected, and the query performance deteriorates.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

default_gsi_statistics_target

Parameter description: Specifies the default statistics target for global secondary indexes.

Parameter type: integer.

Unit: none

Value range: –100 to 3000000

- If this parameter is set to a positive number, it indicates the number of samples of statistics.
- If this parameter is set to a negative value, it indicates the statistical target in percentage. The negative value is converted to the corresponding percentage. For example, **-5** indicates 5%.

Default value: **0**

NOTE

- The default value **0** indicates that the default sampling rate is used for sampling.
- Changing settings of this parameter may result in performance deterioration. If query performance deteriorates, you can:
  1. Restore to the default statistics.
  2. Use hints to force the optimizer to use the optimal query plan. For details, see "Hint-based Optimization" in *SQL Optimization Guide*.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Before changing settings of this parameter, perform a comprehensive evaluation based on the specific workload and query mode. For tables whose data features are not evenly distributed or whose data volume is large, you may need to manually adjust the value of this parameter.

Risks and impacts of improper settings:

- If the value of **default_gsi_statistics_target** is too large, the time and resource consumption of the ANALYZE operation may increase because more sample data needs to be collected to generate statistics. This may increase the overhead of database maintenance.
- If the value of **default_gsi_statistics_target** is too small, the accuracy of statistics may be reduced. As a result, the query optimizer's capability of generating efficient query plans is affected, and the query performance deteriorates.

opt_mcv_buckets_global

Parameter description: Specifies the maximum number of buckets for storing the most common values (MCVs) for columns that are not specified in **ALTER TABLE <table> ALTER COLUMN <attr> SET (OPT_MCV_BUCKETS = <val>)**. This parameter only specifies the upper limit of the number of stored MCVs. The actual number of MCVs is determined by the actual sampling data distribution. This parameter can be set at the PDB level.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: none

Value range: 0 to 10000. If this parameter is set to **0**, the number of MCVs is automatically calculated based on the number of sampled rows.

Default value:

- For a newly installed database: **100**.
- For a database upgraded from an earlier version that does not support this GUC parameter, the default value is **0** after the upgrade. If the earlier version supports this GUC parameter, the default value is retained after the upgrade.
- In the PDB scenario, if this parameter is not set, the global setting is inherited.

NOTE

- The parameter modification takes effect only for subsequent ANALYZE operations. The old statistics of existing tables are not automatically updated. You need to manually run the **ANALYZE** *Table name*; command to collect statistics again or wait for the statistics collection to be automatically triggered.
- If a single column in a table requires a higher upper limit of buckets, you can use the column-level customized statistics parameter to increase the upper limit. The syntax is as follows:

```
ALTER TABLE <table> ALTER COLUMN <attr> SET (OPT_MCV_BUCKETS = <val>)
```

  After the setting, the column is not restricted by the global parameter **opt_mcv_buckets_global**.
- This parameter is invalid during the upgrade observation period.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Before changing settings of this parameter, perform a comprehensive evaluation based on the specific workload and query mode. For a table with a large number of MCVs in the data distribution, you may need to manually adjust the value of this parameter.

Risks and impacts of improper settings:

- If the value of **opt_mcv_buckets_global** is too large, more MCVs will be collected, which may increase the time and resources consumed by the ANALYZE operation. As a result, the database maintenance overhead increases, especially for large tables.
- If the value of **opt_mcv_buckets_global** is too small, the accuracy of cost estimation may be reduced. As a result, the query optimizer's capability of generating efficient query plans is affected, and the query performance deteriorates.

auto_statistic_ext_columns

Parameter description: Collects statistics about multiple columns based on the first **auto_statistic_ext_columns** columns of the composite index in the data table. For example, if a composite index is (a,b,c,d,e) and the GUC parameter is set to **3**, statistics about multiple columns are generated on columns (a,b) and (a,b,c). Multi-column statistics can enable the optimizer to estimate the cardinality more accurately when combined conditions are used for query.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The system catalog does not take effect.
- The statistics take effect only when the types of all columns support the comparison functions '=' and '<'.
- System pseudocolumns in indexes, such as **tableoid** and **ctid**, are not collected.
- By default, distinct values, MCVs without NULL, and MCVs with NULL are collected. Intelligent statistics are not collected for automatically created multi-column statistics. Only MCVs and MCVs with NULL are collected.
- If the index for creating multi-column statistics is deleted and no other index contains the multi-column combination, the multi-column statistics will be deleted in the next ANALYZE operation.
- If the value of this parameter decreases, the new index generates multi-column statistics based on the value of this parameter. The generated multi-column statistics that exceed the value of this parameter will not be deleted.
- If you want to disable the multi-column statistics on a specific combination only, you can retain the value of this parameter and run the DDL command **ALTER TABLE** *tablename* **disable statistics** *((column list))* to disable the statistics on multiple columns in a specific combination.
- Multi-column statistics with weak correlation will be skipped when multi-column statistics are automatically created. If the product of the number of distinct values (NDV) of a single column is close to 80% of the NDV of multiple columns, the multi-column correlation is weak. If the NDV of manually or automatically created multi-column statistics is **-1**, the correlation does not need to be determined.
- If the percentage sampling mode is used in distributed deployment, multi-column statistics will not be automatically created on CNs.
- After the function of automatically creating multi-column statistics is enabled, the overhead of the statistics calculation phase increases. When there are a large number of composite indexes, the ANALYZE performance slightly decreases.

Parameter type: integer.

Unit: none

Value range: 1 to 4. **1** indicates that statistics about multiple columns are not automatically collected.

Default value:

- For a newly installed database: **4**.
- For a database upgraded from a version earlier than 506.1.0: **1**.
- For a database upgraded from 506.1.0 or later: original parameter value.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. When adjusting the value of this parameter, consider the following factors:

- Table size and data characteristics, such as multi-column correlation.
- Query mode, especially the columns and indexes involved in the query.
- Available system resources, such as CPU, memory, and storage space.

Risks and impacts of improper settings:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If the value is too large, ANALYZE operations may be performed frequently, which increases the database maintenance overhead, especially for large tables.
- If the value is too small, the correlation between columns in the table may not be fully captured. As a result, the query optimizer cannot generate the optimal query plan.

constraint_exclusion

Parameter description: Specifies the query optimizer's use of table constraints to optimize queries.

Parameter type: enumerated type

Unit: none

Value range:

- **on** indicates that constraints for all tables are examined.
- **off** indicates that constraints for any table are not examined.
- **partition** indicates that only constraints for inheritance child tables and **UNION ALL** subqueries are examined.

Default value: **partition**

NOTE

- When **constraint_exclusion** is set to **on**, the optimizer compares query conditions with the table's **CHECK** constraints, and omits scanning tables for which the conditions contradict the constraints.
- Currently, **constraint_exclusion** is enabled by default only for cases that are often used to implement table partitioning. If this parameter is enabled for all tables, extra planning is imposed on simple queries, which has no benefits. If you have no partitioned tables, disable the parameter.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If table constraints are frequently used in service queries and these constraints help optimize the query, it may be beneficial to keep enabling this parameter. Otherwise, if these constraints are not helpful for query optimization, disabling this parameter may improve performance.

Risks and impacts of improper settings: The query performance may be affected.

cursor_tuple_fraction

Parameter description: Specifies the optimizer's estimated fraction of a cursor's rows that are retrieved.

Parameter type: floating-point.

Unit: none

Value range: 0 to 1

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **0.1**

NOTE

Smaller values of this setting bias the optimizer towards using **fast start** plans for cursors, which will retrieve the first few rows quickly while perhaps taking a long time to fetch all rows. Larger values put more emphasis on the total estimated time. At the maximum setting of **1.0**, cursors are planned exactly like regular queries, considering only the total estimated time and how soon the first rows might be delivered.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If most rows in the table are frequently accessed, you can increase the value of this parameter.

Risks and impacts of improper settings: The performance may deteriorate.

from_collapse_limit

Parameter description: Specifies whether the optimizer merges subqueries into upper queries based on the resulting FROM list. The optimizer merges subqueries into upper queries if the resulting FROM list would have no more than this many items.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483647

Default value: **8**

NOTE

Smaller values reduce planning time but may lead to inferior execution plans.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If multiple subqueries need to be processed frequently in a query and these subqueries can be effectively combined to improve query performance, you can increase the value of this parameter. Otherwise, if merging subqueries deteriorates performance or increases the time for the optimizer to generate an execution plan, you can reduce the value.

Risks and impacts of improper settings: The performance may deteriorate.

join_collapse_limit

Parameter description: Specifies whether the optimizer rewrites JOIN constructs (except FULL JOIN) into the FROM list based on the number of the items in the result list.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483647

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **8**

NOTE

- Setting this parameter to **1** prevents join reordering. As a result, the join order specified in the query will be the actual order in which the relations are joined. The query optimizer does not always choose the optimal join order. Therefore, advanced users can temporarily set this parameter to **1**, and then specify the join order they desire explicitly.
- Smaller values reduce planning time but lead to inferior execution plans.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Decreasing the value reduces the planning time but may reduce the plan generation quality. Increasing the value increases the planning time but may generate a better plan.

Risks and impacts of improper settings: The trade-off between the planning time and the plan generation quality. Improper settings may cause the effect of one party to be unacceptable.

plan_mode_seed

Parameter description: This is a debugging parameter. Currently, it supports only **OPTIMIZE_PLAN** and **RANDOM_PLAN**.

Parameter type: integer.

Unit: none

Value range: –1 to 2147483647

- **0**: **OPTIMIZE_PLAN** mode. In this mode, the dynamic planning algorithm is used to estimate the cost and generate the optimal plan.
- **-1**: **RANDOM_PLAN** mode. In this mode, a plan is randomly generated, and the seed value for generating a random number is not specified. The optimizer randomly generates an integer value within the range of 1 to 2147483647 and generates a random execution plan based on the random number.
- Integer within the range of 1 to 2147483647: **RANDOM_PLAN** mode. In this mode, a plan is randomly generated, and the seed value of the seed identifier for generating a random number is specified by the user. The optimizer generates a random execution plan based on the seed value.

Default value: **0**

NOTE

If this parameter is not set to 0, the specified plan hint does not take effect.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: When this parameter is set to the **RANDOM_PLAN** mode, the optimizer randomly generates an execution plan. The execution plan may not be the optimal one, which affects the query performance. Therefore, you are advised to set this parameter to **0** during normal service operations or O&M, such as upgrade, scale-out, and scale-in.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: The performance may deteriorate.

enable_random_datanode

Parameter description: Specifies whether the query of the replication table is conducted on a random DN. A complete replication table is stored on each DN for random retrieval to release the pressure on nodes.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the function of querying the replication table on a random DN is enabled.
- **off** indicates that the function of querying the replication table on a random DN is disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the node pressure may increase sharply.

hashagg_table_size

Parameter description: Specifies the hash table size during the execution of the HASH JOIN operation.

Parameter type: integer.

Unit: none

Value range: 0 to 1073741823. **0** indicates that the database automatically adjusts the size of the hash table as required.

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. In actual applications, if you encounter specific query scenarios, for example, aggregation operations that process a large amount of data, you may need to manually adjust this parameter to optimize performance.

Risks and impacts of improper settings: Increasing the size of the hash table can reduce the disk I/O in the Hash Agg operation because more data can be retained in the memory. However, if the size of the hash table is too large, too much memory may be occupied, resulting in insufficient memory. If this parameter is set to a small value, the memory may not be effectively used, resulting in more disk I/O operations and decreasing the query speed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_codegen

Parameter description: Specifies whether code optimization can be enabled. Currently, the code optimization uses the LLVM optimization.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that code optimization is enabled.
- **off** indicates that code optimization is disabled.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is enabled, resource usage may increase in expression query scenarios.

codegen_compile_thread_num

Parameter description: Specifies the number of Codegen compilation threads.

Parameter type: integer.

Unit: none

Value range: 1 to 8

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Change the value when the number of concurrent services is large and the service performance bottleneck occurs during expression execution.

Risks and impacts of improper settings: If the number of threads is too large, the system performance may deteriorate. However, when there are a large number of concurrent services, you can increase the number of threads to improve the throughput performance.

llvm_max_memory

Parameter description: Specifies the maximum memory occupied by IRs (including cached and in-use IRs) generated during Codegen compilation. The memory used by Codegen is not applied for by preoccupation. It is a part of **max_dynamic_memory** and is restricted by the **llvm_max_memory** parameter.

Parameter type: integer.

Unit: KB

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 0 to 2147483647. If the value exceeds the specified value, the original recursive execution logic, instead of the Codegen execution logic, is used. When the upper limit is reached and a downgrade is triggered, decreasing the value of **llvm_max_memory** cannot immediately release the memory occupied by extra IRs. The memory occupied by IRs is released after the corresponding SQL statements are executed.

Default value: **131072**, that is, 128 MB.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **100** without a unit, it indicates 100 KB. If this parameter is set to **16MB**, it indicates 16 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Retain the default value. Change the value when the value of **llvm_used_memory** in the pv_total_memory_detail view reaches the upper limit of the default value and the service performance bottleneck lies in the expression execution process.

Risks and impacts of improper settings:

- If the parameter is set to an excessively small value, the system does not use the Codegen execution logic, affecting the use of functions.
- If the parameter is set to an excessively large value, LLVM compilation may occupy too many resources of other threads. As a result, the overall system performance deteriorates.

enable_codegen_print

Parameter description: Specifies whether the LLVM IR function can be printed in logs.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the IR function can be printed in logs.
- **off** indicates that the IR function cannot be printed in logs.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Enable this function when locating faults.

Risks and impacts of improper settings: If this parameter is enabled, a large number of logs are generated, occupying disk I/O resources. As a result, the read and write performance of the database system foreground query deteriorates.

codegen_cost_threshold

Parameter description: The LLVM compilation takes some time to generate executable machine code. Therefore, LLVM compilation is beneficial only when the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

actual execution cost is more than the sum of the cost required for generating machine code and the optimized execution cost. Parameter **codegen_cost_threshold** specifies a threshold. If the estimated execution cost exceeds the threshold, LLVM optimization is performed. codegen uses **plan_rows** of the execution operator as the cost to compare with the value of **codegen_cost_threshold**. You can run the **explain** command to view the value of **plan_rows**.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **100000**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. You can change the value only when you need to change the threshold for triggering the codegen mechanism.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_bloom_filter

Parameter description: Specifies whether the Bloom filter optimization can be used.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The Bloom filter optimization can be used.
- **off**: The Bloom filter optimization cannot be used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

bloom_filter_build_max_rows

Parameter description: Specifies the maximum data threshold on the build side of the hash join for creating a Bloom filter when Bloom filter optimization is enabled. If the data volume on the build side is greater than the value of this parameter, no Bloom filter is created.

Parameter type: integer.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 1 to 2147483647

Default value: **2000000**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. This parameter takes effect only when the **enable_bloom_filter** parameter is enabled.

Setting suggestion: If the storage and compute resources are sufficient, you can increase the value of this parameter to achieve better performance.

Risks and impacts of improper settings: When the storage and compute resources are insufficient, **query_dop** is high (greater than 16), and the value is too large (greater than 2000000), the system performance may deteriorate.

bloom_filter_apply_threshold

Parameter description: Specifies the minimum data threshold on the apply side of the hash join for creating a Bloom filter when Bloom filter optimization is enabled. If the data volume on the apply side is less than the value of this parameter, no Bloom filter is created.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483647

Default value: **10000**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. This parameter takes effect only when the **enable_bloom_filter** parameter is enabled.

Setting suggestion: Set this parameter to an integer ranging from 1000 to 20000.

Risks and impacts of improper settings:If this parameter is set to a large value (greater than 20000), scenarios where performance can be optimized after Bloom filter is added will be lost. If this parameter is set to a small value (smaller than 1000), Bloom filter is added for queries with a small amount of data. As a result, the performance deteriorates.

NOTE

- After the **enable_bloom_filter** and **bloom_filter_build_max_rows** parameters are enabled or set, the bloom filter can be generated only when the bloom filter implementation or principle restrictions are met.
- Limitations on the bloom filter implementation and principles:
  - The build side must be inside the hash join.
  - The apply side must be IMCVScan.
  - Left join, full join, and left anti join are not supported.
  - Bloom filter crosses operators that cannot be pushed down during apply/build, such as Func Agg and Limit.
  - Bloom filter calls the semi join API for estimation. Bloom filter can be created only when the outer table selectivity is less than 2/3. The outer table selectivity is calculated based on statistics. For details, see "Feature Usage Guidelines > CBO > Technical Principle > Statistics" in *Developer Guide*.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

You can confirm the build and apply nodes by observing the plan. For the hash join node in the plan, the lower subnode is on the build side, and the upper subnode is on the apply side. The following plan is used as an example, in which **Imcv Scan on public.row_hashjoin_table_01 a** is on the build side and **Imcv Scan on public.row_hashjoin_table_02 b** is on the apply side:

```
                                QUERY PLAN
-------------------------------------------------------------------------------------------------
 Row Adapter  (cost=3033.82..3033.82 rows=59804 width=48)
   Output: a.c_int, a.c_bigint, a.c_smallint, b.c_int, b.c_bigint, b.c_smallint
   ->  Vector Sonic Hash Join  (cost=374.00..3033.82 rows=59804 distinct=[200, 600] width=48)
         Output: a.c_int, a.c_bigint, a.c_smallint, b.c_int, b.c_bigint, b.c_smallint
         Hash Cond: (b.c_int = a.c_int)
         Generate Bloom Filter On Expr: a.c_int
         Generate Bloom Filter On Index: 0
         ->  Imcv Scan on public.row_hashjoin_table_02 b  (cost=0.00..924.00 rows=75852 width=24)
               Output: b.c_int, b.c_bigint, b.c_smallint
               Filter: (b.c_int > 2400)
               Filter By Bloom Filter On Expr: b.c_int
               Filter By Bloom Filter On Index: 0
         ->  Imcv Scan on public.row_hashjoin_table_01 a  (cost=0.00..299.00 rows=6000 width=24)
               Output: a.c_int, a.c_bigint, a.c_smallint
               Filter: (a.c_int > 2400)
```

enable_extrapolation_stats

Parameter description: Specifies whether the extrapolation logic is used for data of date type based on historical statistics. If this logic is used, the estimation accuracy can be improved for tables whose statistics are not collected in a timely manner. However, the estimation may be too large due to inference errors. This parameter must be enabled when date-type data is periodically inserted.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the extrapolation logic is used.
- **off** indicates that the extrapolation logic is not used.

Default value:

- For a newly installed database, the default value is **on**.
- If the GUC parameter does not exist in the source version, the default value is **on**. If the GUC parameter exists in the source version, the original setting is retained.

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Before enabling **enable_extrapolation_stats**, you should evaluate the frequency of data changes and the query mode. Enabling this parameter may be helpful if the data changes quickly and process incomplete statistics are required frequently for the query optimizer. Before official launch, adequate testing should be conducted in the test environment to determine the specific impact of enabling extrapolation statistics on performance. After extrapolation statistics are enabled, closely monitor the query performance and the accuracy of database statistics to ensure that no performance problem is introduced or the accuracy of statistics is greatly reduced.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this parameter is enabled, the query performance may deteriorate due to inference errors.

query_dop

Parameter description: Specifies the user-defined degree of parallelism (DOP).

Parameter type: integer.

Unit: none

Value range: –64 to 64

- A value ranging from 1 to 64 indicates that the fixed SMP is enabled and the system will use the specified DOP.
- **0** indicates that the SMP adaptation is enabled, and the system will dynamically select the optimal DOP based on resource usage and plan characteristics.
- A value ranging from –64 to –1 indicates that the SMP adaptation is enabled, and the system limits the DOP that can be adaptively selected.

NOTE

- After enabling parallel queries, ensure you have sufficient CPU, memory, network, and I/O resources to achieve the optimal performance.
- To prevent performance deterioration caused by an overly large value of **query_dop**, the system calculates the maximum number of available CPU cores for a DN and uses the number as the upper limit for this parameter. If the value of **query_dop** is greater than 4 and also the upper limit, the system resets **query_dop** to the upper limit.

Default value: **1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. When resources such as the CPU, memory, I/O, and network bandwidth are sufficient, a higher degree of parallelism indicates the better performance improvement.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_analyze_check

Parameter description: Specifies whether it is allowed to check whether statistics were collected about tables whose **reltuples** and **relpages** are displayed as **0** in pg_class during plan generation.

Parameter type: Boolean

Unit: none

Value range:

- **on** indicates that the tables will be checked.
- **off** indicates that the tables will not be checked.

Default value: **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Enabling the check may cause plan generation overhead, but ensures that statistics have been collected. If most tables whose **reltuples** and **relpages** are **0** in pg_class do not need to collect additional statistics, you can disable the check.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_sonic_hashagg

Parameter description: Specifies whether to use the Hash Agg operator designed for column-oriented hash tables when certain constraints are met.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the Hash Agg operator designed for column-oriented hash tables is used when certain constraints are met.
- **off** indicates that the Hash Agg operator designed for column-oriented hash tables is not used.

NOTE

- If **enable_sonic_hashagg** is enabled and the Hash Agg operator designed based on the column-oriented hash table is used when the query meets the constraint condition, the memory usage of the Hash Agg operator can be reduced. However, in scenarios where **enable_codegen** is enabled and the performance is significantly improved, the performance of the operator may deteriorate.
- When **enable_sonic_hashagg** is enabled and the hash agg operator designed based on the column-oriented hash table is used when the query meets the limitation, the operator is displayed as **Sonic Hash Aggregation** in the execution plan and execution information of Explain Analyze/Performance; when the query does not meet the limitation, the operator is displayed as **Hash Aggregation**. For details, see "Introduction to the SQL Execution Plan > Execution Plan Details" in *SQL Optimization Guide*.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the query performance may deteriorate in this scenario.

enable_sonic_hashjoin

Parameter description: Specifies whether to use the hash join operator designed for column-oriented hash tables when certain constraints are met.

Parameter type: Boolean.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **on** indicates that the hash join operator designed for column-oriented hash tables is used when certain constraints are met.
- **off** indicates that the hash join operator designed for column-oriented hash tables is not used.

NOTE

- Currently, the parameter can be used only for Inner Join.
- When **enable_sonic_hashjoin** is enabled, the memory usage of query using the Hash Inner operator can be reduced. However, in scenarios where the code generation technology can significantly improve performance, the performance of the operator may deteriorate.
- When **enable_sonic_hashjoin** is enabled and the hash join operator designed based on the column-oriented hash table is used when the query meets the limitation, the operator is displayed as **Sonic Hash Join** in the execution plan and execution information of Explain Analyze/Performance; when the query does not meet the limitation, the operator is displayed as **Hash Join**. For details, see "Introduction to the SQL Execution Plan > Execution Plan Details" in *SQL Optimization Guide*.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the query performance may deteriorate in this scenario.

vector_engine_features

Parameter description: Specifies whether to enable the optional features of the vectorized engine. Enabling these features may not improve the execution performance. In specific service scenarios, you can use this GUC parameter to set vectorized engine features to achieve optimal execution performance.

This parameter is used to set the combination of vectorized engine feature switches. For example, if there are multiple features **rule1**, **rule2**, **rule3**, and **rule4**, you can perform settings as follows:

```
set vector_engine_features='rule1';          -- Enable the vectorized engine feature rule1.
set vector_engine_features='rule2,rule3';    -- Enable the vectorized engine features rule2 and rule3.
set vector_engine_features=none;             -- Disable all optional vectorized engine features.
```

Parameter type: enumerated type

Unit: none

Value range:

- **none**: does not use any optional vectorized engine feature.
- **enable_ca_hashjoin**: uses the Ca Hash Join operator designed for Cache-Aware HashTable.
- **enable_ca_hashagg**: uses the Ca Hash Agg operator designed for Cache-Aware HashTable.

Default value: **enable_ca_hashjoin, enable_ca_hashagg**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-2**.

Setting suggestion: Retain the default value. If necessary, you are advised to enable only the corresponding feature after determining the execution scenario.

Risks and impacts of improper settings: If a feature is disabled, the execution performance may deteriorate in some scenarios. You are advised to enable the feature after thorough tests.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When **enable_ca_hashjoin** is set in the GUC parameter, this switch depends on the basic switch **enable_hashjoin**. When **enable_hashjoin** is disabled, no hash join exists in the plan, and **enable_ca_hashjoin** is invalid.
  - This switch applies only to the following scenarios:
    - Join keys
      - Join key of the same type. For example, **int** = **varchar** is not supported.
      - Non-expression join key. For example, **a** – 1 = **b** is not supported.
      - For details about the data types supported by join keys, see **Table 6-22**.
    - Join types
      - Inner Join: supported.
      - Right Join: partially supported. It is not supported in the case of join conditions and is supported in other cases.
      - Semi Join: partially supported. It is supported in non-projection scenarios and only when outer table data is output. It is not supported in projection scenarios or when inner table data is output.
      - Anti Join: partially supported. It is supported in non-projection scenarios and only when outer table data is output. It is not supported in projection scenarios or when inner table data is output.
      - Right Semi Join: partially supported. It is supported only when inner table data is output.
      - Right Anti Join: partially supported. It is supported only when inner table data is output.
      - Full Join: not supported.
      - Left Join: not supported.
    - Others
      - It is supported only when the number of columns in the output data of an inner table is less than or equal to 64.
  - When **enable_ca_hashjoin** is enabled, the query meets the specified condition, and the Ca Hash Join operator designed for Cache-Aware Hash Table is used, the operator is displayed as **Vector Ca Hash Join** in the execution plan and execution information of Explain Analyze/Performance. If the specified condition is not met, the operator is displayed as **Vector Sonic Hash Join** or **Vector Hash Join**. For details, see "Introduction to the SQL Execution Plan > Execution Plan Details" in *SQL Optimization Guide*.
- When **enable_ca_hashagg** is set in the GUC parameter, this switch depends on the basic switch **enable_hashagg**. When **enable_hashagg** is disabled, no HashAgg exists in the plan, and **enable_ca_hashagg** is invalid.
  - This switch applies only to the following scenarios:
    - Non-numeric type supported by the original vectorized AGG.
    - The number of aggregate functions cannot exceed 21.
    - Columns returned by lower-layer operators include only columns used as HashKey, columns used for aggregation, and columns irrelevant to HashAgg. For example, if **SELECT col1::numeric FROM test_table GROUP BY col1;** is used, the lower-layer operators may return **col1** and **col1::numeric**, and only **col1** is used as the hash key.
    - **unique_check** of the AGG node is **false**. (Currently, it may be **true** only in some subqueries.)
    - For details about the data types supported by join keys, see **Table 6-22**.
  - When **enable_ca_hashagg** is enabled, the query meets the specified condition, and the Ca Hash Agg operator designed for Cache-Aware Hash Table is used, the operator is displayed as **Vector Ca Hash Aggregate** in the execution plan and

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

execution information of Explain Analyze/Performance. If the specified condition is not met, the operator is displayed as **Vector Sonic Hash Aggregate** or **Vector Hash Aggregate**. For details, see "Introduction to the SQL Execution Plan > Execution Plan Details" in *SQL Optimization Guide*.

Table 6-22 Data types supported by join keys

Category	Data Type
Numeric types	TINYINT [UNSIGNED]
	SMALLINT [UNSIGNED]
	MEDIUMINT [UNSIGNED]
	BIGINT [UNSIGNED]
	FLOAT4
	FLOAT8
Date/Time types	DATE
	DATEA
	SMALLDATETIME
	ABSTIME
	INTERVAL
	RELTIME
	TIME [WITHOUT TIME ZONE]
	TIMESTAMP [WITHOUT TIME ZONE]
Character types	BPCHAR
	CHAR
	VARCHAR
	TEXT
Other data types	BOOL
	OID

enable_sonic_optspill

Parameter description: Specifies whether to optimize the number of files to be written to disks for the hash join operator designed for column-oriented hash tables. If this parameter is enabled, the number of files written to disks does not increase significantly when the hash join operator writes a large number of files to disks.

Parameter type: Boolean.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range:

- **on** indicates that the number of files to be written to disks for the hash join operator designed for column-oriented hash tables is optimized.
- **off** indicates that the number of files to be written to disks for the hash join operator designed for column-oriented hash tables is not optimized.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the query performance may deteriorate in this scenario.

plan_cache_mode

Parameter description: Specifies the policy for generating an execution plan in the prepared statement.

Parameter type: enumerated type

Unit: none

Value range: **auto**, **force_generic_plan**, and **force_custom_plan**

- **auto**: The optimizer automatically selects a custom plan or generic plan.
- **force_generic_plan** indicates that the generic plan is forcibly used (soft parse). The generic plan is a plan generated after you run a prepared statement. The plan policy binds parameters to the plan when you run the **EXECUTE** statement to execute the plan. The advantage of this scheme is that repeated optimizer overheads can be avoided in each execution. The disadvantage is that the plan may not be optimal when data skew occurs for the bound parameters and may result in poor plan execution performance. The bound parameters bind the types of parameters set for the first time. If the type of a parameter set to the same placeholder is different from the previous time, an error is reported.
- **force_custom_plan** indicates that the custom plan is forcibly used (hard parse). The custom plan is a plan generated after you run a prepared statement where parameters in the EXECUTE statement are embedded. The custom plan generates a plan based on specific parameters in the EXECUTE statement. This scheme generates a preferred plan based on specific parameters each time and has good execution performance. The disadvantage is that the plan needs to be regenerated before each execution, resulting in a large amount of repeated optimizer overhead.

NOTE

This parameter is valid only for prepared statements. It is used when the parameterized field in a prepared statement has severe data skew.

Default value: **auto**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the actual service scenario.

Risks and impacts of improper settings: The plan generation overhead may increase or the plan generation quality may deteriorate.

enable_router

Parameter description: Specifies whether to enable the manual pushdown function so that supported query statements can be pushed down to specific DNs for execution.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You need to manually set the pushdown node to be enabled for a specific distribution key.

Risks and impacts of improper settings: After this function is enabled, if the distribution key is incorrectly set, data may be stored on an incorrect DN. Therefore, exercise caution when enabling this function.

router

Parameter description: Specifies the detailed attributes of the router function. This parameter is valid only when **enable_router** and **enable_light_proxy** are enabled. This parameter is used to calculate the DN where the given distribution key is located based on the hash distribution key of the table. After the router is set, the supported SQL statements are pushed down to the DN for execution.

Parameter type: string.

Unit: none

Value range:

This parameter consists of two parts: **'schema_name.table_name,"distribute_keys"'** (pay attention to the double quotation marks). The meanings are as follows:

- **schema_name.table_name**: indicates the schema name and table name. If schema_name is not set, the default value current_schema is used.
- **distribute_keys**: Values of all distribution keys in the distributed table are separated by commas (,). The sequence of the values must be the same as that of the distribution keys in the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: ""

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You need to manually set the pushdown node to be enabled for a specific distribution key.

Risks and impacts of improper settings: If **router** is incorrectly set, data may be stored to an incorrect DN, causing unpredictable problems. Therefore, exercise caution when setting this parameter.

enable_force_vector_engine

Parameter description: Specifies whether to forcibly generate vectorized execution plans for a vectorized execution operator if the operator's child node is a non-vectorized operator. When this parameter is set to **on**, vectorized execution plans are forcibly generated.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that vectorized operators are forcibly generated.
- **off** indicates that the vectorized operator optimizer determines whether to perform vectorization.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. You are advised to enable this function in vectorized service scenarios.

Risks and impacts of improper settings: If this parameter is enabled, the query performance may deteriorate.

enable_auto_explain

Parameter description: Specifies whether to enable the function of automatically printing execution plans. This parameter is used to locate slow stored procedures or slow queries and is valid only for the currently connected CN. It does not affect the execution of other CNs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If you want to view the execution plan, enable this parameter. However, this causes the current system performance to deteriorate.

Risks and impacts of improper settings: After this function is enabled, the current system performance may deteriorate.

auto_explain_level

Parameter description: Specifies the log level for automatically printing execution plans.

Parameter type: enumerated type

Unit: none

Value range:

- **log**: Logs of the **log** level or higher will be printed.
- **notice**: Logs of the **notice** level or higher will be printed.

Default value: **log**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

auto_explain_log_min_duration

Parameter description: Specifies the minimum duration of execution plans that are automatically printed. Only execution plans whose duration is greater than the value of **auto_explain_log_min_duration** will be printed. For example, if this parameter is set to **0**, all executed plans are printed. If this parameter is set to **3000**, all executed plans are printed if the execution of a statement takes more than 3000 ms.

Parameter type: integer.

Unit: milliseconds

Value range: 0 to 2147483647

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **100** without a unit, it indicates 100 ms. If this parameter is set to **2min**, it indicates 2 minutes. The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value. You can adjust the value based on service requirements to output slow query statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If the value is too small, too much output content may be generated.

auto_explain_analyze_options

Parameter description: Specifies whether to display the runtime plan information when execution plans are automatically printed. This parameter is used to view the actual execution plans of FQS and PGXC plan pushdown statements in DN logs. It is valid only for the currently connected CN and does not affect the execution of other CNs.

NOTE

- This parameter can be used only when **enable_auto_explain** is enabled. The constraint of **enable_auto_explain** is inherited. EXPLAIN statements are not printed. Only the statement plans that are normally executed can be printed.
- Plan logs on DNs take effect only when **enable_auto_explain** is set to **true** and **auto_explain_level** is set to **log**.
- If this parameter is set to any value except **None**, the ANALYZE function is included, and the basic runtime plan information is displayed.
- Stream, SMP, and SQL Bypass plans are not supported. If no runtime information is displayed for the plan, check whether the Stream, SMP, and SQL Bypass plans are generated.
- Only the plan information of the current node can be displayed on each node. For example, the CN does not display the plan information of DNs in logs or on the client.
- In the current version, AUTO EXPLAIN does not display CPU plan information at runtime. The CPU option is reserved.

Parameter type: enumerated type

Unit: none

Value range:

- **None**: automatically prints execution plan and does not display the runtime plan information.
- **Analyze**: collects basic runtime plan information, including **rows** and **time**.
- **Buffers**: collects runtime information about **rows**, **time**, and **buffer**.
- **Cpu**: collects runtime information about **rows**, **time**, and **cpu**.

Default value: **None**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If you want to view the execution plan information, enable this parameter. However, this causes the current system performance to deteriorate.

Risks and impacts of improper settings: After this function is enabled, the current system performance may deteriorate.

max_datanode_for_plan

Parameter description: Specifies the number of execution plans displayed on DNs when FQS plans are generated. The number of plans that are displayed on DNs is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

determined by the smaller value between the number of DNs on the cluster and the value of this parameter.

Parameter type: integer.

Unit: none

Value range: 0 to 8192

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Change the value when you need a more detailed multi-node execution plan.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_inner_unique_opt

Parameter description: Specifies whether to optimize Inner Unique for nested-loop join, hash join, and sort merge join, that is, whether to reduce the number of matching times when the attribute corresponding to the inner table in the join condition meets the UNIQUE constraint.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the query performance of the corresponding scenarios may deteriorate.

enable_indexscan_optimization

Parameter description: Specifies whether to optimize B-tree index scan (IndexScan and IndexOnlyScan) in the Astore.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: used.
- **off**: not used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the query performance of the corresponding scenarios may deteriorate.

enable_uniq_idx_a_compat

Parameter description: Specifies whether composite unique indexes are compatible with Oracle Database for **NULL** values.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: compatible.
- **off**: incompatible.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is applicable only to ORA-compatible databases. Retain the default value.

Risks and impacts of improper settings: Improper settings may affect compatibility or increase query planning overheads.

create_gsi_opt

Parameter description: Specifies the GSI creation mode. Other modes except **none** take effect only after the **enable_stream_operator** parameter is enabled.

Parameter type: enumerated type

Unit: none

Value range:

- **none**: PGXC is used.
- **build**: The STREAMING and GSI BUILD operators are used.
- **parallel**: The STREAMING and GSI BUILD operators and the parallel sorting mode are used. (The **parallel_workers** parameter of the base table needs to be set.)

Default value: **build**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: The performance of creating global secondary indexes may deteriorate.

session_sequence_cache

Parameter description: Specifies the **sequence** value applied for one-time interaction in the current session. The unused values are automatically discarded after the session ends.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483647

Default value: **10**

NOTE

The default value is **10**. In high-concurrency scenarios, the performance of single and batch insertion is good.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: When using **sequence** to import data in batches, you can increase the value of this parameter to improve the insertion speed and high concurrency performance. When a single data record is inserted concurrently, set this parameter to **1** to reduce the sequence change. If you have high requirements on continuity, you need to specify the required cache when creating a sequence. If the value of this parameter is greater than that of cache, the value automatically becomes invalid.

Risks and impacts of improper settings: If the value is too large, the performance of inserting a single record may deteriorate. If the value is too small, the performance of inserting records in batches may deteriorate.

enable_invisible_indexes

Parameter description: Specifies whether the optimizer can use invisible indexes.

NOTE

After an index is set to invisible, the performance of query statements may be affected. If you do not want to change the index visibility status and want to use invisible indexes, set **enable_invisible_indexes** to **on**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The optimizer can use invisible indexes.
- **off**: The optimizer cannot use invisible indexes.

Default value: **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If invisible indexes are used but this parameter is not enabled, invisible indexes may be ignored. Therefore, a better plan may not be considered.

enable_dynamic_samplesize

Parameter description: Specifies whether to dynamically adjust the number of sampled rows. For a large table with more than one million rows, the number of sampled rows is dynamically adjusted during statistics collection to improve statistics accuracy.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: indicates that the function is enabled.
- **off**: indicates that the function is disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the accuracy of statistics may decrease.

NOTE

The function of dynamically adjusting the number of sampled rows supports only absolute sampling.

default_statistic_granularity

Parameter description: Specifies which partition-level statistics of a partitioned table are collected by default when **PARTITION_MODE** is not specified. This parameter does not take effect for non-partitioned tables.

Parameter type: enumerated type

Unit: none

Value range:

- **all**: Statistics of the entire table and partition are collected.
- **global**: Statistics of the entire table are collected.
- **partition**: Statistics of the partition are collected.
- **global_and_partition**: Statistics about the entire table and partitions are collected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **all_complete**: Statistics about the entire table and partitions are collected. Compared with the scenario where the value is **all**, the sampling rate is higher, which may cause longer time required for collecting statistics.

Default value: **global**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If partition-level statistics need to be collected, you can set this parameter as required. However, the **ANALYZE** performance may be affected.

Risks and impacts of improper settings: You need to balance the maintenance overhead and the accuracy of statistics. Improper settings may cause high costs for one party.

NOTE

In upgrade scenarios, the value of this GUC parameter after the upgrade must be the same as that before the upgrade.

enable_fast_numeric_agg

Parameter description: Specifies whether to enable aggregation optimization for the numeric data type.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The aggregation optimization is enabled for the numeric data type.
- **off**: The aggregation optimization is disabled for the numeric data type.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the query performance of the corresponding scenarios may deteriorate.

enable_poisson_outer_optimization

Parameter description: Specifies whether to use Poisson to correct the selectivity of the filter condition in the outer table in outer join when the selectivity is calculated in the plan generation phase.

Parameter type: Boolean.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: Poisson is not used to modify the selectivity of the filter condition of the outer table in the outer join. That is, the filter condition of the outer table is not calculated as pushdown.
- **off**: The selectivity is calculated based on the filter condition of the outer table in the outer join.

Default value: **off** for an upgraded cluster and **on** for a newly installed cluster.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: A good execution plan may not be selected, and slow SQL statements may occur.

NOTE

The related functions take effect only when the GUC parameter **cost_model_version** is greater than or equal to 5 or is set to **0** and **enable_poisson_outer_optimization** is set to **on**.

enable_predreorder_by_rank

Parameter description: Specifies whether to reorder multiple filter criteria of a single base table based on their selectivity and calculation cost during plan generation to reduce the execution overhead.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Multiple filter criteria are reordered based on their selectivity and calculation cost during plan generation.
- **off**: Multiple filter criteria are not reordered based on their selectivity and calculation cost during plan generation.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this function is disabled, some plans cannot be optimized.

NOTE

The related functions take effect only when **cost_model_version** is greater than or equal to 6 or is set to **0** and **enable_predreorder_by_rank** is set to **on**.

enable_shared_set_update

Parameter description: Specifies whether the optimizer generates a subplan for each column to be updated when a subquery is used to update multiple columns

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

in an UPDATE statement. When this parameter is enabled, only one subplan is generated and shared among multiple columns.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Only one subplan is generated based on the subquery.
- **off**: A subplan is generated for each column to be updated.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this function is disabled, the query execution efficiency of updating multiple columns using subqueries may be reduced, and the query optimization and execution overhead may increase.

NOTE

- The related functions take effect only when **cost_model_version** is greater than or equal to 6 or is set to **0** and **enable_shared_set_update** is set to **on**.
- When **enable_shared_set_update** is set to **on** and a shared set update plan is generated, if the statement contains an IMCV table, no IMCV SCAN plan will be generated.

cursor_sharing

Parameter description: Specifies whether to enable automatic parametrization or reuse of the same statement plan.

Parameter type: enumerated type

Unit: none

Value range:

- **force**: enables automatic parameterization and reuse of the same statement plan. If the automatic parameterization conditions are met, the automatic parameterization cache is used preferentially. If the conditions are not met, the same statement plan reuse cache is used. For example, in MySQL- or M-compatible mode, parameterization is not supported and **cursor_sharing** is automatically set to **exact**.
- **exact**: enables only the same statement plan reuse function.
- **off**: disables both the automatic parameterization and same statement plan reuse functions.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this parameter is set to **force**, some constants in the statement will be replaced by parameters. The SQL plan may be changed, the plan performance may deteriorate, the syntax tree may be affected, the value of **unique_sql_id** may change, and the SQL patch may become invalid.

If this parameter is set to **force** or **exact**, the memory usage may increase. In addition, if the plan template cannot be reused, the performance cannot be improved through plan reuse. Before setting this parameter, check whether the service model is suitable. When the upper limit of the plan memory is controlled, if a large number of different SQL templates are stored, the plan cache memory usage is too high and reaches the upper limit, triggering the planned memory eviction. As a result, the performance deteriorates and even drops to zero in extreme scenarios.

Before modifying this parameter, make sure you understand its meaning and fully verify the change in a test environment.

NOTE

- After automatic parameterization is enabled, its execution mechanism is the same as that of manually binding parameters.
- After automatic parameterization is enabled, the query statement is automatically rewritten in parameterized form. Therefore, the WARNING or ERROR information (such as the prompt text and line number) displayed during the execution may be different from that when automatic parameterization is disabled. This is expected.

opt_use_static_stats

Parameter description: After this function is enabled, statistics such as the number of rows and number of pages can be directly read from the PG_CLASS system catalog, instead of being obtained by sampling pages of the base table. This parameter is used to manually verify the plan generation effect of the optimizer on the premise that the statistics are given.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Statistics are directly read from PG_CLASS.
- **off**: The base table is sampled to obtain statistics such as the number of rows and pages in the default mode.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this function is enabled, the optimizer does not detect the actual table size. The statistics are based only on the PG_CLASS system catalog.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

The related functions take effect only when **cost_model_version** is greater than or equal to 6 or is set to **0** and **opt_use_static_stats** is set to **on**.

enable_sel_cache

Parameter description: When this function is enabled, the selectivity cache is used for the equi-join predicates whose left and right operands are ordinary columns and have been pushed down through deduction by the optimizer in the path generation phase. The cache is used to accelerate the repeated calculation of the predicate selectivity in the path search phase.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: This function is used.
- **off**: This function is not used.

Default value:

- **on**: default value of the newly installed database.
- **off**: For an upgrade from a version earlier than 506.1.0 to the current version, the default value of this parameter is **off**. If this parameter exists before the upgrade, the default value is **off** after the upgrade regardless of its setting.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this function is disabled, the performance may deteriorate during path generation of the optimizer.

NOTE

- Results of multi-column statistics cannot be cached.
- The related functions take effect only when **cost_model_version** is greater than or equal to 7 or is set to **0** and **enable_sel_cache** is set to **on**.
- A cache is built to accelerate selectivity calculation only when an equi-join predicate composed of ordinary columns is encountered.
- If the left and right operands contain type conversion and the converted types are forcibly converted in binary mode, this function can also be used to accelerate calculation. For details, see the description of **castmethod** in **PG_CAST**.
- If both **enable_hashjoin** and **enable_mergejoin** are disabled, this function does not take effect.

stats_history_record_limit

Parameter description: Specifies the maximum number of historical statistics that can be retained for each object (including tables, columns, partitions, and indexes). When collecting statistics about an object, the system saves the statistics to the historical statistics table. When the number of statistics about the object in

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

the historical statistics table reaches the threshold and new statistics are collected, the earlier statistics are deleted.

Parameter type: integer.

Unit: none

Value range: 0 to 100

Default value: **10**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestions: The default value is recommended. If you need to record statistics of more historical versions, increase the value of this parameter. However, the performance of ANALYZE may be affected.

Risks and impacts of improper settings: If the value is too large, the ANALYZE performance may be affected.

stats_history_retention_time

Parameter description: Specifies the retention period of historical statistics about each object (including tables, columns, partitions, and indexes). When collecting statistics about an object, the system saves the statistics to the historical statistics table. If the retention period of the statistics about the object in the historical statistics table exceeds the threshold, the system deletes the statistics that exceed the retention period when collecting new statistics.

Parameter type: floating-point.

Unit: days (d)

Value range: **-1** or a value ranging from 0 to 365000. **-1** indicates that the history statistics are not cleared over time.

Default value: **31**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestions: The default value is recommended. If you need to record statistics of earlier versions, increase the value of this parameter. However, the performance of ANALYZE may be affected.

Risks and impacts of improper settings: If the value is too large, the ANALYZE performance may be affected.

opt_behavior_knob

Parameter description: Specifies the behavior change of the optimizer in specific scenarios for manual query optimization.

Parameter type: string.

Unit: none

Value range: All behavior changes in specific scenarios are enabled. You can use SHOW opt_behavior_knob to check the changes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

ALL,on(ROWID_OPT,DIST_QUERY_ADAPTIVE_FALLBACK,XC_REMOTE_JOIN_PARAM,MERGE_INTO_SHIP,SUBLINK_QUERY_SHIP,SUBPLAN_SORTED_PATH_FIX,PARAM_PATH_SKIP_CACHING_VARRATIO,PREFER_UNIQUE_INDEX,PREFER_LONGER_INDEX,PPI_COST_MODEL1,PREDICATE_INFER_ENHANCE,HIERARCHY_QRY_CHECK_SUBLINK_PULLUP,ACTUAL_RANGE_BY_INDEX,FIX_PART_INDEX_SCAN_COST,OPT_KNOB_20260703_1,OPT_KNOB_20260807_1,OPT_KNOB_20260728_1,OPT_KNOB_20260812_1),off()

- **PARAM_PATH_SKIP_CACHING_VARRATIO**: specifies whether to skip the caching of the selectivity of related predicates during cardinality estimation when parameterized paths are generated, ensuring the accuracy of upper-layer join cardinality estimation.
- **SUBPLAN_SORTED_PATH_FIX**: specifies the path selection related to the sorted path of sublinks, preventing paths with low costs from being directly eliminated and ensuring the accuracy of path selection.
- **DIST_QUERY_ADAPTIVE_FALLBACK**: specifies whether to roll back a plan to a stream plan if the following conditions are met: CN resources are insufficient (memory or CPU usage exceeds 80%); the plan generated by the distributed adaptive plan is a distributed plan for delivering statements; and the plan is a non-parameterized path plan.
- **XC_REMOTE_JOIN_PARAM**: specifies whether an outer table can pass parameters to an inner table for execution when the inner table is a remote join operator.
- **MERGE_INTO_SHIP**: specifies the pushdown of the MERGE INTO statement. In the MERGE INTO USING SELECT FROM dual table scenario, the MERGE INTO statement can be pushed down to DNs to execute the CN lightweight/FQS plan.
- **SUBLINK_QUERY_SHIP**: specifies whether to push down the sublink FQS. When the pushdown conditions are met, the sublink can be pushed down to DNs for execution.
- **ROWID_OPT**: specifies whether to enable the ROWID system column predicate condition rewriting, ORDER BY statement rewriting, max/min optimization, distribution pruning, and partition pruning. For details, see "Feature Usage Guidelines > ROWID Table > Operation Guide > DML" in *Developer Guide*.
- **PREFER_UNIQUE_INDEX**: specifies whether the rule that the unique index scan path is preferred takes effect. This parameter is compatible with **no_unique_index_first** in **sql_beta_feature**. The relationship between the two parameters is OR. The rule that the unique index scan path is preferred takes effect when **no_unique_index_first** is not set in **sql_beta_feature** or **PREFER_UNIQUE_INDEX** is enabled. Only the equivalent scenario is supported. The priority of this rule is lower than that of **PREFER_LONGER_INDEX**.
- **PREFER_LONGER_INDEX**: specifies whether the rule that the longest and leftmost index scan path is preferred takes effect. Only the equivalent scenario is supported. The priority of this rule is higher than that of **PREFER_UNIQUE_INDEX**.
- **PPI_COST_MODEL1**: specifies the cost correction in the scenario where the LOCAL index parameterized path is used. This prevents the plan from being suboptimal due to small cost estimation when the LOCAL index is selected and the query condition does not contain the partition key. If the database is upgraded from an earlier version that does not support this parameter, this function is enabled by default and the parameter does not take effect during the observation period. After the observation period ends, the function is automatically disabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **PREDICATE_INFER_ENHANCE**: controls predicate inference and reduction, simplifies predicate logic, and improves query performance. For details, see the description of predicate inference and reduction in **cost_model_version**.
- **HIERARCHY_QRY_CHECK_SUBLINK_PULLUP**: controls the behavior of pulling up sublinks at the same level in hierarchical queries. When this parameter is enabled, a stricter check is performed on sublink pulling up. After an upgrade from an earlier version that does not support this parameter to the current version, this function is disabled by default and does not take effect during the upgrade observation period.
- **ACTUAL_RANGE_BY_INDEX**: controls whether the optimizer directly queries the actual maximum or minimum value through an index when estimating the selectivity of a range condition and finds that the condition falls on the leftmost or rightmost end of the statistical histogram. This prevents estimation deviation caused by outdated statistics.
- **FIX_PART_INDEX_SCAN_COST**: specifies whether to correct the estimated cost of local index scans on partitioned tables to prevent the plan from being suboptimal due to estimation deviation. If the database is upgraded from an earlier version that does not support this parameter, this function is enabled by default and the parameter does not take effect during the observation period. After the observation period ends, the function is automatically disabled.
- **OPT_KNOB_20260703_1**: specifies whether the optimizer counts the null ratio when using multi-column statistics to calculate the NDV. If this function is enabled, the null ratio is not counted, and a more accurate cardinality estimation can be obtained. If the database is upgraded from an earlier version that does not support this parameter, this function is enabled by default and the parameter does not take effect during the observation period. After the observation period ends, the function is automatically disabled.
- **OPT_KNOB_20260807_1**: specifies whether to correct the optimizer's estimation on the number of index pages in a partitioned table to prevent the plan from being suboptimal due to estimation deviation. If the database is upgraded from an earlier version that does not support this parameter, this function is enabled by default and the parameter does not take effect during the observation period. After the observation period ends, the function is automatically disabled.
- **OPT_KNOB_20260728_1**: controls the optimizer agg cost model. When this function is enabled, the hypergeometric distribution probability algorithm is used. When this function is disabled, the proportional scaling algorithm is used. This affects the estimation of the number of agg rows in hashagg and avoids poor planning due to estimation deviation. If the database is upgraded from an earlier version that does not support this parameter, this function is enabled by default and the parameter does not take effect during the observation period. After the observation period ends, the function is automatically disabled.
- **OPT_KNOB_20260812_1**: specifies whether to correct the cost when the optimizer generates a nested-loop join path and the foreign table contains the sort operator. This prevents suboptimal plans caused by estimation deviation. If the database is upgraded from an earlier version that does not support this parameter, this function is enabled by default and the parameter does not take effect during the observation period. After the observation period ends, the function is automatically disabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value:

- For a newly installed database:

  'ALL,on(ROWID_OPT,DIST_QUERY_ADAPTIVE_FALLBACK,XC_REMOTE_JOIN_PARAM,MERGE_INTO_SHIP,SUBLINK_QUERY_SHIP,SUBPLAN_SORTED_PATH_FIX,PARAM_PATH_SKIP_CACHING_VARRATIO,PREFER_UNIQUE_INDEX,PREFER_LONGER_INDEX,PPI_COST_MODEL1,PREDICATE_INFER_ENHANCE,HIERARCHY_QRY_CHECK_SUBLINK_PULLUP,ACTUAL_RANGE_BY_INDEX,FIX_PART_INDEX_SCAN_COST,OPT_KNOB_20260703_1,OPT_KNOB_20260807_1,OPT_KNOB_20260728_1,OPT_KNOB_20260812_1),off()'.
- For the **PPI_COST_MODEL1**, **FIX_PART_INDEX_SCAN_COST**, **HIERARCHY_QRY_CHECK_SUBLINK_PULLUP**, **OPT_KNOB_20260703_1**, **OPT_KNOB_20260807_1**, **OPT_KNOB_20260728_1**, and **OPT_KNOB_20260812_1** parameters: If the database is upgraded from an earlier version that does not support these parameters, this function is enabled by default and the parameters do not take effect during the observation period. After the observation period ends, the function is automatically disabled.
- In the PDB scenario, if this parameter is not set, the global setting is inherited.

Setting method: This is a USERSET parameter, and you are advised not to use the gs_guc tool to set it. The setting method is as follows:

First, you can run **SHOW opt_behavior_knob** to check which behavior changes are controllable. For example, the query output result is as follows:

```
gaussdb=# SHOW opt_behavior_knob;
              opt_behavior_knob
-------------------------------------------------

ALL,on(ROWID_OPT,DIST_QUERY_ADAPTIVE_FALLBACK,XC_REMOTE_JOIN_PARAM,MERGE_INTO_SHIP,SUBLINK_QUERY_SHIP,SUBPLAN_SORTED_PATH_FIX,PARAM_PATH_SKIP_CACHING_VARRATIO,PREFER_UNIQUE_INDEX,PREFER_LONGER_INDEX,PPI_COST_MODEL1,PREDICATE_INFER_ENHANCE,HIERARCHY_QRY_CHECK_SUBLINK_PULLUP,ACTUAL_RANGE_BY_INDEX,FIX_PART_INDEX_SCAN_COST,OPT_KNOB_20260703_1,OPT_KNOB_20260807_1,OPT_KNOB_20260728_1,OPT_KNOB_20260812_1),off()
(1 row)
```

Controllable behavior changes are identified by uppercase letters, and the special ID **ALL** is used for setting all behavior changes. You can set it to **on** or **off** to specify whether the behavior change takes effect. To disable the behavior change of skipping cache selectivity for parameterized paths, run the following commands:

```
gaussdb=# SET opt_behavior_knob = 'off(PARAM_PATH_SKIP_CACHING_VARRATIO)';
gaussdb=# SHOW opt_behavior_knob;
              opt_behavior_knob
-------------------------------------------------

ALL,on(ROWID_OPT,DIST_QUERY_ADAPTIVE_FALLBACK,XC_REMOTE_JOIN_PARAM,MERGE_INTO_SHIP,SUBLINK_QUERY_SHIP,SUBPLAN_SORTED_PATH_FIX,PREFER_UNIQUE_INDEX,PREFER_LONGER_INDEX,PPI_COST_MODEL1,PREDICATE_INFER_ENHANCE,HIERARCHY_QRY_CHECK_SUBLINK_PULLUP,ACTUAL_RANGE_BY_INDEX,FIX_PART_INDEX_SCAN_COST,OPT_KNOB_20260703_1,OPT_KNOB_20260807_1,OPT_KNOB_20260728_1,OPT_KNOB_20260812_1),off(PARAM_PATH_SKIP_CACHING_VARRATIO)
(1 row)
```

The behavior change of skipping cache selectivity for parameterized paths is disabled, and other behavior changes remain unchanged.

The **ALL** identifier can be used to quickly enable or disable all behavior changes.

```
gaussdb=# SET opt_behavior_knob = 'off(ALL)';
SET
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SHOW opt_behavior_knob;
              opt_behavior_knob
-------------------------------------------------

ALL,on(),off(ROWID_OPT,DIST_QUERY_ADAPTIVE_FALLBACK,XC_REMOTE_JOIN_PARAM,MERGE_INTO_SHIP,
SUBLINK_QUERY_SHIP,SUBPLAN_SORTED_PATH_FIX,PARAM_PATH_SKIP_CACHING_VARRATIO,PREFER_UNIQ
UE_INDEX,PREFER_LONGER_INDEX,PPI_COST_MODEL1,PREDICATE_INFER_ENHANCE,HIERARCHY_QRY_CHE
CK_SUBLINK_PULLUP,ACTUAL_RANGE_BY_INDEX,FIX_PART_INDEX_SCAN_COST,OPT_KNOB_20260703_1,OPT
_KNOB_20260807_1,OPT_KNOB_20260728_1,OPT_KNOB_20260812_1)
(1 row)

gaussdb=# SET opt_behavior_knob = 'on(ALL)';
SET
gaussdb=# SHOW opt_behavior_knob;
              opt_behavior_knob
-------------------------------------------------

ALL,on(ROWID_OPT,DIST_QUERY_ADAPTIVE_FALLBACK,XC_REMOTE_JOIN_PARAM,MERGE_INTO_SHIP,SUBLI
NK_QUERY_SHIP,SUBPLAN_SORTED_PATH_FIX,PARAM_PATH_SKIP_CACHING_VARRATIO,PREFER_UNIQUE_I
NDEX,PREFER_LONGER_INDEX,PPI_COST_MODEL1,PREDICATE_INFER_ENHANCE,HIERARCHY_QRY_CHECK_S
UBLINK_PULLUP,ACTUAL_RANGE_BY_INDEX,FIX_PART_INDEX_SCAN_COST,OPT_KNOB_20260703_1,OPT_KN
OB_20260807_1,OPT_KNOB_20260728_1,OPT_KNOB_20260812_1),off()
(1 row)
```

Setting suggestion: Retain the default value. Disable specified behavior changes as required.

Risks and impacts of improper settings: If one or all behavior changes are disabled, the optimizer may output an unexpected plan, which may affect the system performance.

NOTE

The related functions take effect only when **cost_model_version** is greater than or equal to **5** or is set to **0** and **opt_behavior_knob** is set to **on**.

exec_behavior_knob

Parameter description: Specifies the behavior change of the executor in specific scenarios.

Parameter type: string.

Unit: none

Value range: You can run the **SHOW exec_behavior_knob** command to view the parameter values.

```
ALL,on(SHOW_BATCH_ERROR_DETAIL,INDEXBY_FUNC_OPTIMIZE,SYS_CB_PATH_NULL_CTRL,PGXC_MERGE_
UPDATE,SORT_KEYS_CACHE,PLSQL_RETURN_EXCEPTION),off(AUTO_INCR_LP_INSERT)
```

- **SHOW_BATCH_ERROR_DETAIL**: In the batch execution scenario, if an exception is thrown during cyclic execution, the number of current batches and the bound parameter values in the error batch are printed.
- **INDEXBY_FUNC_OPTIMIZE**: When the first/last/prior/next function of a set variable with an index is called in a stored procedure, the red-black tree is used to optimize performance. After this function is enabled, the memory usage increases to maintain the red-black tree, and the performance deteriorates in some scenarios.
- **SYS_CB_PATH_NULL_CTRL**: specifies whether to perform special processing when the hierarchical query system function SYS_CONNECT_BY_PATH is executed and a NULL value is read. After this function is enabled, an empty

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

value is output for NULL values. Otherwise, the string **"null"** is output. If the database is upgraded from an earlier version that does not support this parameter, this function is enabled by default and the parameter does not take effect during the observation period. After the observation period ends, the function is automatically disabled.

- **AUTO_INCR_LP_INSERT**: specifies whether to enable light proxy pushdown optimization in simple insertion scenarios where the partition key is an auto-increment column. If it is disabled, the execution plan displayed using EXPLAIN changes from light proxy to PGXC in this scenario. As a result, the performance deteriorates in some scenarios.
- **PGXC_MERGE_UPDATE**: specifies whether to allow multiple updates to the same row of data when the MERGE INTO syntax is executed for a single SQL statement in a PGXC plan. If this parameter is enabled and the **enable_cid_sync** parameter is also enabled, an error is reported.
- **SORT_KEYS_CACHE**: specifies whether to enable the multi-column information cache mechanism of the Sort operator to improve sorting performance. When this function is enabled, information about the sorting key can be dynamically cached, improving the sorting performance of the row executor.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

The Sort operator has the following limitations on multi-column information cache:

- Full sorting only

  This feature is optimized for full sorting scenarios where the row executor performs high-frequency cross comparison. For top-N queries with the LIMIT clause, heap sort is usually used and the overall number of comparisons is small. Therefore, the cost-effectiveness of caching multiple columns of data is low. In this case, this feature does not take effect. If the query with the LIMIT clause uses quick sort, this feature can be enabled. For example, the execution plan output is as follows:

```
gaussdb=# EXPLAIN ANALYZE SELECT * FROM test_sort ORDER BY 1 LIMIT 1;
                                                        QUERY PLAN
---------------------------------------------------------------------------------------------------------------------------
--------
 [Parameterized]
 Limit  (cost=46.03..46.03 rows=1 width=4) (actual time=0.098..0.098 rows=1 loops=1)
   ->  Sort  (cost=46.03..52.04 rows=1 width=4) (actual time=0.095..0.095 rows=1 loops=1)
         Sort Key: a
         Sort Method: top-N heapsort  Memory: 25kB
         ->  Seq Scan on test_sort  (cost=0.00..34.02 rows=2402 width=4) (actual
time=0.045..0.050 rows=10 loops=1)
 Total runtime: 0.397 ms
(7 rows)
gaussdb=# EXPLAIN ANALYZE SELECT * FROM test_sort ORDER BY 1 LIMIT 6;
                                                        QUERY PLAN
---------------------------------------------------------------------------------------------------------------------------
--------
 [Parameterized]
 Limit  (cost=77.08..77.09 rows=6 width=4) (actual time=0.132..0.137 rows=6 loops=1)
   ->  Sort  (cost=77.08..83.08 rows=6 width=4) (actual time=0.127..0.129 rows=6 loops=1)
         Sort Key: a
         Sort Method: quicksort  Memory: 25kB
         ->  Seq Scan on test_sort  (cost=0.00..34.02 rows=2402 width=4) (actual
time=0.060..0.070 rows=10 loops=1)
 Total runtime: 0.551 ms
(7 rows)
```

- Unique key scenario

  On a single DN, secondary comparison will not be performed when there is only one column in the sort key or the data of the first sort key is unique (only the actual data needs to be unique and the UNIQUE constraint is not necessary). In this case, this feature does not take effect.

- Memory overflow protection

  If the memory required by the original sorted data exceeds the working memory (**work_mem**) configured in the system, the data needs to be spilled to disks for processing. Alternatively, if the overall memory usage exceeds the working memory plus the additional memory overhead caused by multi-column caching, the multi-column caching feature automatically becomes invalid.

- Limit on the number of sort keys

  To prevent the cache structure from bloating excessively, the system disables this feature when the total number of sort keys in a query statement exceeds 32.

- **PLSQL_RETURN_EXCEPTION**: specifies whether the EXCEPTION statement can capture the type conversion errors reported by the RETURN statement in the stored procedure in ORA-compatible mode. For details, see **Error Trapping Statements**.

  NOTE

  For an upgrade from a version that does not support this parameter, the setting of this parameter does not take effect during the upgrade observation period. After the upgrade is committed, the setting takes effect and this option is set to **off** by default.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value:

- For a newly installed database:

  'ALL,on(SHOW_BATCH_ERROR_DETAIL,INDEXBY_FUNC_OPTIMIZE,SYS_CB_PATH_NULL_CTRL,PGXC_MERGE_UPDATE,SORT_KEYS_CACHE,PLSQL_RETURN_EXCEPTION),off(AUTO_INCR_LP_INSERT)'.
- This parameter is added in the 507.0.0 version. If the database is upgraded from a version earlier than 507.0.0, the default value is as follows:

  'ALL,on(SHOW_BATCH_ERROR_DETAIL),off(INDEXBY_FUNC_OPTIMIZE,SYS_CB_PATH_NULL_CTRL,AUTO_INCR_LP_INSERT,PGXC_MERGE_UPDATE,SORT_KEYS_CACHE,PLSQL_RETURN_EXCEPTION)'.
- For an upgrade from a version later than 507.0.0 to this version, the original behavior change is the same as that in the source version, and the new behavior change is disabled by default.

Setting method: This is a USERSET parameter, and you are advised not to use the gs_guc tool to set it. The setting method is as follows:

First, you can run **SHOW exec_behavior_knob** to check which behavior changes are controllable. For example, the query output result is as follows:

```
gaussdb=# SHOW exec_behavior_knob;
        exec_behavior_knob
----------------------------------------

ALL,on(SHOW_BATCH_ERROR_DETAIL,INDEXBY_FUNC_OPTIMIZE,SYS_CB_PATH_NULL_CTRL,PGXC_MERGE_
UPDATE,SORT_KEYS_CACHE,PLSQL_RETURN_EXCEPTION),off(AUTO_INCR_LP_INSERT)
(1 row)
```

Controllable behavior changes are identified by uppercase letters, and the special ID **ALL** is used for setting all behavior changes. You can set it to **on** or **off** to specify whether the behavior change takes effect. To disable the **SHOW_BATCH_ERROR_DETAIL** behavior change, run the following commands:

```
gaussdb=# SET exec_behavior_knob='off(SHOW_BATCH_ERROR_DETAIL)';
gaussdb=# SHOW exec_behavior_knob;
        exec_behavior_knob
----------------------------------------

ALL,on(INDEXBY_FUNC_OPTIMIZE,SYS_CB_PATH_NULL_CTRL,PGXC_MERGE_UPDATE,SORT_KEYS_CACHE,PLS
QL_RETURN_EXCEPTION),off(SHOW_BATCH_ERROR_DETAIL,AUTO_INCR_LP_INSERT)
(1 row)
```

The behavior change of **SHOW_BATCH_ERROR_DETAIL** is disabled, and other behavior changes remain unchanged.

The **ALL** identifier can be used to quickly enable or disable all behavior changes.

```
gaussdb=# SET exec_behavior_knob='off(ALL)';
SET
gaussdb=# SHOW exec_behavior_knob;
        exec_behavior_knob
----------------------------------------

ALL,on(),off(SHOW_BATCH_ERROR_DETAIL,INDEXBY_FUNC_OPTIMIZE,SYS_CB_PATH_NULL_CTRL,AUTO_INC
R_LP_INSERT,PGXC_MERGE_UPDATE,SORT_KEYS_CACHE,PLSQL_RETURN_EXCEPTION)
(1 row)
gaussdb=# SET exec_behavior_knob = 'on(ALL)';
SET
gaussdb=# SHOW exec_behavior_knob;
        exec_behavior_knob
----------------------------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
ALL,on(SHOW_BATCH_ERROR_DETAIL,INDEXBY_FUNC_OPTIMIZE,SYS_CB_PATH_NULL_CTRL,AUTO_INCR_LP
_INSERT,PGXC_MERGE_UPDATE,SORT_KEYS_CACHE,PLSQL_RETURN_EXCEPTION),off()
(1 row)
```

When **SHOW_BATCH_ERROR_DETAIL** is enabled, the default values of the parameters that are not explicitly specified in the configuration file are loaded. By default, the **SHOW_BATCH_ERROR_DETAIL** and **SYS_CB_PATH_NULL_CTRL** parameters are enabled. Therefore, when the behavior change of **SHOW_BATCH_ERROR_DETAIL** is enabled, the **SYS_CB_PATH_NULL_CTRL** parameter is enabled, and the **INDEXBY_FUNC_OPTIMIZE**, **AUTO_INCR_LP_INSERT**, **PGXC_MERGE_UPDATE**, and **PLSQL_RETURN_EXCEPTION** parameters are disabled.

```
gs_guc reload -Z datanode -N all -I all -c "exec_behavior_knob='on(SHOW_BATCH_ERROR_DETAIL)'"
gaussdb=# SHOW exec_behavior_knob;
         exec_behavior_knob
-----------------------------------------

ALL,on(SHOW_BATCH_ERROR_DETAIL,SYS_CB_PATH_NULL_CTRL),off(INDEXBY_FUNC_OPTIMIZE,AUTO_INC
R_LP_INSERT,PGXC_MERGE_UPDATE,SORT_KEYS_CACHE,PLSQL_RETURN_EXCEPTION)
(1 row)
```

Setting suggestion:

- **SORT_KEYS_CACHE**: Set this parameter based on the **work_mem** and data distribution configured in the current service system.

  NOTICE

  Enabling multi-column cache increases the memory usage of the Sort operator proportionally. The increased memory size is in strict linear relationship with the number of data rows and the number of extra cached columns. The relationship is evaluated using the following formula:

  Extra memory increment ≈ Total number of rows x 10 x (Total number of sort keys – 1) bytes

  (Note: 10 is the estimated byte overhead base for single-column cache.)

  - Recommended scenarios for enabling:

    High duplicity in leading columns: When the leading columns in ORDER BY have many duplicate values (such as low-cardinality fields like status and department), the optimization payoff of multi-column comparison becomes increasingly pronounced as data duplication grows.

    Trailing columns positioned later internally: When the subsequent columns used for secondary comparison are located further back in the physical table layout (that is, with larger column indexes), the optimization benefit scales directly with the size of the column index.
  - Recommended scenarios for disabling:

    Insufficient memory resources: If the **work_mem** configured for the system is small or the overall memory resources of the database are insufficient, you are advised not to enable this feature.
- Others: You are advised to use the default value.

Risks and impacts of improper settings: Disabling one or all options will change the behavior of the corresponding options.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_cid_sync

Parameter description: If this function is enabled, non-read-only statements delivered to DNs use the CID on the CN during the execution of a top-level distributed PGXC plan. This ensures the visibility of tuple modified in the current transaction.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: This function is enabled.
- **off**: This function is disabled.

Default value:

- For a newly installed database: **on**.
- For a database upgraded from an earlier version that does not support this GUC parameter, the default value is **off** after the upgrade. If the earlier version supports this GUC parameter, the default value is retained after the upgrade.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this function is disabled, the visibility of tuples modified in the current transaction may be affected when a distributed PGXC plan involves delivering multiple statements in a row of SQL statements.

enable_startwith_prior_enhance

Parameter description: Specifies whether to enhance the control over columns modified by PRIOR. If this parameter is enabled, the PRIOR keyword can be used to modify columns in the SELECT query column, DISTINCT clause, WHERE clause, GROUP BY clause, HAVING clause, WINDOW clause, and ORDER BY clause in the SELECT statement with the CONNECT BY clause to display the parent row column value. In addition, an error is reported in the following scenarios:

(1) In the SELECT statement with the CONNECT BY clause, PRIOR is used to modify columns in the ON condition and ORDER SIBLINGS BY clause.

(2) In a statement without the CONNECT BY clause, PRIOR is used to modify columns.

(3) In a clause (SELECT query column, DISTINCT clause, WHERE clause, GROUP BY clause, HAVING clause, WINDOW clause, or ORDER BY clause) that supports PRIOR, PRIOR is used to modify the pseudo columns **LEVEL**, **CONNECT_BY_ISCYCLE**, and **CONNECT_BY_ISLEAF**.

If this parameter is disabled, using the PRIOR keyword to modify a column outside the CONNECT BY clause is equivalent to not adding the PRIOR keyword. In this case, the current row and column values are displayed.

Parameter type: Boolean.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Disable this function when restoring the behavior of the compatibility with earlier versions.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ustore_consistent_read_opt

Parameter description: Specifies the consistent read policy for Ustore sequential scans.

Parameter type: enumerated type

Unit: none

Value range:

- **pbrcr**: page-based row-level consistent read policy.
- **fast_pbrcr**: page-based row-level consistent read policy that optimizes the page lock logic and undo traversal algorithm.

Default value: **pbrcr**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

parser_ignore_hints

Parameter description: Specifies whether to disable the parsing of hints.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The processing of hints is disabled in the parsing phase, so that hints in statements do not take effect. When this parameter is enabled, the effect of SQL patches is affected, but outlines and SPM baselines are not affected.
- **off**: Hints are processed normally in the parsing phase.

Default value: **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this parameter is enabled, statements that have been optimized using hints or SQL patches may be rolled back to poor query plans, resulting in performance deterioration.

enable_partition_statistics_min_rows

Parameter description: Specifies whether to use the **partition_statistics_min_rows** option set for a partitioned table when **partition_mode** is set to **ALL** for the ANALYZE operation on the partitioned table.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The value of **partition_statistics_min_rows** specified for a partitioned table can be used as the minimum number of rows sampled in a non-empty partition.
- **off**: The value of **partition_statistics_min_rows** specified for a partitioned table cannot be used as the minimum number of rows sampled in a non-empty partition.

Default value:

- For a newly installed database: **on**.
- For a database upgraded from an earlier version that does not support this GUC parameter, the default value is **off** after the upgrade. If the earlier version supports this GUC parameter, the default value is retained after the upgrade.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the partition-level statistics obtained by the ANALYZE operation with **partition_mode** set to **ALL** may be inaccurate. For details, see **ANALYZE | ANALYSE**.

6.3.9 Error Reporting and Logging

6.3.9.1 Logging Destination

log_destination

Parameter description: GaussDB supports several methods of logging server messages. Set this parameter to a list of desired log destinations separated by commas. (For example, **log_destination** can be set to **"stderr,csvlog"**.)

Parameter type: string.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range: **"stderr"**, **"csvlog"**, **"syslog"**, and **"eventlog"**.

- **"stderr"** indicates that logs are printed to the screen.
- **"csvlog"** indicates that logs are output in comma separated value (CSV) format. The prerequisite for generating logs in CSV format is that **logging_collector** must be set to **on**. For details, see **Using CSV Log Output**.
- **"syslog"** indicates that logs are recorded using the syslog of the OS. GaussDB can record logs using syslog from **LOCAL0** to **LOCAL7**. For details, see **syslog_facility**. To record logs using syslog, add the following information to syslog daemon's configuration file:
  local0.* */var/log/gaussdb*
- **"eventlog"** indicates that logs are recorded using the event log component. This function is supported only in the Windows OS.

Default value: **"stderr"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

logging_collector

Parameter description: Specifies whether to enable the logger thread to collect logs. This thread captures log messages sent to **stderr** or **csvlog** and redirects them into log files.

This method is more effective than recording logs to syslog because some types of messages cannot be displayed in syslog output, such as messages indicating the loading failures of dynamic link libraries and error messages generated by scripts.

NOTICE

It is possible to log to **stderr** without using the parameter **logging_collector** and the log messages will go to where the server's **stderr** is directed. However, this method is only suitable for low log volumes due to difficulties in rotating log files.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The log collection is enabled.
- **off**: The log collection is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: When this parameter is disabled, CSV logs cannot be used to record logs.

log_directory

Parameter description: Specifies the directory for storing log files when **logging_collector** is set to **on**. The value can be an absolute path, or relative to the data directory. The **log_directory** parameter can be dynamically modified using the **gs_guc reload** command. Only the SYSADMIN user can access this parameter.

NOTICE

- If **log_directory** is set to an invalid path, the cluster cannot be started.
- If you modify the **log_directory** parameter using the **gs_guc reload** command, and the specified path is valid, the log files are output to this new path. If the specified path is invalid, the log files are output to the valid path set last time and the database operation is not affected. The invalid value of **log_directory** is still written into the configuration file.
- In the sandbox environment, the path cannot contain **/var/chroot**. For example, if the absolute path of log is **/var/chroot/var/lib/log/Ruby/gs_log/cn_log**, you only need to set the path to **/var/lib/log/Ruby/gs_log/cn_log**.

NOTE

- Valid path: Users have read and write permissions on the path.
- Invalid path: Users do not have read or write permission on the path.

Parameter type: string.

Unit: none

Value range: a string.

Default value: specified during installation

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_filename

Parameter description: Specifies the names of generated log files when **logging_collector** is set to **on**. The value is treated as a strftime pattern, so %-escapes can be used to specify time-varying file names. Only the SYSADMIN user can access this parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- You are advised to use %-escapes to specify the log file names for efficient management of log files.
- If **log_destination** is set to **csvlog**, log files are output in CSV format with timestamped names, for example, **server_log.1093827753.csv**.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **gaussdb-***%Y-%m-%d_%H%M%S***.log**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_file_mode

Parameter description: Specifies the permissions of log files when **logging_collector** is set to **on**. The value of **log_file_mode** is usually a number in the format acceptable to the **chmod** and **umask** system calls.

NOTICE

- Before setting this parameter, set **log_directory** to store the logs to a directory other than the data directory.
- Do not make the log files world-readable because they might contain sensitive data.

Parameter type: integer.

Unit: none

Value range: an octal integer ranging from 0000 to 0777 (that is, 0 to 511 in the decimal format).

NOTE

- **0600** indicates that log files are readable and writable only to the server administrator.
- **0640** indicates that log files are readable and writable to members of the administrator's group.

Default value: 0600

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_truncate_on_rotation

Parameter description: Specifies the writing mode of the log files when **logging_collector** is set to **on**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: GaussDB overwrites the existing log files of the same name on the server.
- **off**: GaussDB appends the logging messages to the existing log files of the same name on the server.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_rotation_age

Parameter description: Specifies the interval for creating a log file when **logging_collector** is set to **on**. If the duration from the time when the last log file was created to the current time is greater than the value of **log_rotation_age**, a new log file will be generated.

NOTE

Assume that you want logs to be kept for 7 days, a log file generated each day to be named **server_log.Mon** on Monday, **server_log.Tue** on Tuesday, and so forth, and this week's log files to be overwritten by next week's log files.

Setting method: Set **log_filename** to **server_log.%a**, **log_truncate_on_rotation** to **on**, and **log_rotation_age** to **1440**.

Parameter type: integer.

Unit: minute

Value Range: 0 to 35791394. **0** indicates that the time-based creation of new log files is disabled.

Default value: **1440** (that is, 1 day)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

log_rotation_size

Parameter description: Specifies the maximum size of a server log file when **logging_collector** is set to **on**. If the total size of messages in a log file exceeds the maximum size, a log file will be generated.

Parameter type: integer.

Unit: KB

Value Range: 0 to 2097151. **0** indicates that the capacity-based creation of new log files is disabled.

Default value: 20480 (20 MB)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

syslog_facility

Parameter description: Specifies the syslog facility to be used when **log_destination** is set to **syslog**.

Parameter type: enumerated type.

Unit: none

Value range: **local0**, **local1**, **local2**, **local3**, **local4**, **local5**, **local6**, and **local7**.

Default value: local0

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

syslog_ident

Parameter description: Specifies the identifier of GaussDB messages in syslog logs when **log_destination** is set to **syslog**.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **"postgres"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

event_source

Parameter description: This parameter takes effect only in a Windows environment. It specifies the identifier of GaussDB messages in logs when **log_destination** is set to **eventlog**.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **"PostgreSQL"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.9.2 Logging Time

client_min_messages

Parameter description: Specifies the level of messages sent to the client. All messages at or above the specified level are sent.

Parameter type: enumerated type

Unit: none

Value range: The message levels are **debug5**, **debug4**, **debug3**, **debug2**, **debug1**, **log**, **info**, **notice**, **warning**, **error**, **fatal**, and **panic** in ascending order. For details about the parameters, see **Table 6-23**.

NOTICE

- In general, the lower the set level, the more messages will be sent to the client.
- For client log messages, if **client_min_messages** is set to **fatal** or **panic**, its effective value will be forcibly adjusted to **error**. Regardless of the level set in **client_min_messages**, messages at the **info** level are always sent to the client.
- When a client is connected to a CN, debug (**debug5**, **debug4**, **debug3**, **debug2**, and **debug1**) and log messages are not sent to the client due to security and performance reasons. To view these two types of logs, view the server logs of the CN.

Default value: **notice**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_min_messages

Parameter description: Specifies the level of messages written to server logs. All messages at or above the specified level are written.

Parameter type: enumerated type

Unit: none

Value range: The message levels are **debug5**, **debug4**, **debug3**, **debug2**, **debug1**, **log**, **info**, **notice**, **warning**, **error**, **fatal**, and **panic** in ascending order. For details about the parameters, see **Table 6-23**.

NOTICE

- In general, the lower the set level, the more messages will be written into server logs.
- For server log messages, if **log_min_messages** is set to a level not higher than **error**, all messages at the **log** level will be written into server logs.
- Frequent printing affects system performance and stability. Therefore, the debug levels 1–5 are generally used for onsite fault locating. You are advised to enable these levels by module (see **logging_module**) and promptly reset the levels afterward.

Default value: **warning**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_min_error_statement

Parameter description: Specifies the level of messages corresponding to SQL statements written to server logs. All SQL statements corresponding to messages at or above the specified level are written.

Parameter type: enumerated type

Unit: none

Value range: The message levels are **debug5**, **debug4**, **debug3**, **debug2**, **debug1**, **log**, **info**, **notice**, **warning**, **error**, **fatal**, and **panic** in ascending order. For details about the parameters, see **Table 6-23**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- In general, the lower the set level, the more SQL statements will be written into server logs.
- For SQL messages in server logs, if the level is set to **error** or lower, the SQL statements corresponding to messages at the **log** level will be written into server logs.
- If an SQL statement contains personal information and you do not need to record the SQL statement that reports an error, you can set **log_min_error_statement** to **panic**.

Default value: **error**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_min_duration_statement

Parameter description: When the execution time of an SQL statement is greater than or equal to the value of this parameter, the SQL statement and its execution time are recorded.

Setting **log_min_duration_statement** makes it easy to trace query statements that need to be optimized. For clients using extended query protocols, the time required for parsing, binding, and executing steps are logged independently.

Parameter type: integer.

Unit: milliseconds

NOTICE

When you use this parameter together with **log_statement**, the SQL statements that have been logged by **log_statement** will not be repeatedly logged. If syslog is not used, you are advised to use **log_line_prefix** to record PIDs or session IDs so that statement messages and execution time messages can be associated based on PIDs or session IDs.

Value range: –1 to 2147483647

- **-1** indicates that this function is disabled.
- **0** indicates that all SQL statements and their execution time are recorded.
- Positive integer indicates that an SQL statement and its execution time are recorded when the execution time of the SQL statement is greater than or equal to the value of this parameter.

Default value: **3000 (that is, 3s)**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**. If the value is an integer, the default unit is ms. If the value contains a unit, the unit must be ms, s, min, h, or d. For example, if the value is **600** without a unit, **log_min_duration_statement** indicates 600 ms. If the value is **10min**, **log_min_duration_statement** indicates 10 minutes.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is out of the range or the value contains a unit other than ms, s, min, h, and d, an error message is displayed.

backtrace_min_messages

Parameter description: Specifies the level of messages corresponding to function stack information written to server logs. All stack information corresponding to messages at or above the specified level is written.

Parameter type: enumerated type

Unit: none

Value range: The message levels are **debug5**, **debug4**, **debug3**, **debug2**, **debug1**, **log**, **info**, **notice**, **warning**, **error**, **fatal**, and **panic** in ascending order. For details about the parameters, see **Table 6-23**.

NOTICE

- In general, the lower the level, the more stack information is written to the server logs.
- This parameter is used to locate faults on site. Frequent printing of function stacks affects system overhead and stability. Therefore, you are advised to set **backtrace_min_messages** to **fatal** or **panic** during fault locating.

Default value: **panic**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing. If the level is too low, concurrent thread stack printing may affect service execution in high-concurrency scenarios. For example, if this parameter is set improperly when a large number of concurrent connections are established, the processing efficiency of the connection thread pool will be slowed down, or even the database instance will become unreachable.

Table 6-23 explains message severities used by GaussDB. When logs are output to syslog or eventlog (only in the Windows environment, this parameter is not involved in GaussDB), GaussDB performs the conversion as shown in the following table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-23 Message severity levels

Severity	Description	System Log	Event Log
debug[1-5]	Provides detailed debug information.	DEBUG	INFORMATION
log	Reports information of interest to administrators, for example, checkpoint activity.	INFO	INFORMATION
info	Provides information implicitly requested by users, for example, output from **VACUUM VERBOSE**.	INFO	INFORMATION
notice	Provides information that might be helpful to users, for example, truncation of long identifiers and index created as part of the primary key.	NOTICE	INFORMATION
warning	Provides warnings of likely problems, for example, COMMIT outside a transaction block.	NOTICE	WARNING
error	Reports an error that causes a command to terminate.	WARNING	ERROR
fatal	Reports the reason that causes a session to terminate.	ERR	ERROR
panic	Reports an error that caused all database sessions to terminate.	CRIT	ERROR

6.3.9.3 Logging Content

debug_print_parse

Parameter description: Specifies whether to print parsing tree results.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The printing is enabled.
- **off**: The printing is disabled.

Default value: **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

debug_print_rewritten

Parameter description: Specifies whether to print query rewriting results.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The printing is enabled.
- **off**: The printing is disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

debug_print_plan

Parameter description: Specifies whether to print the query execution plan to logs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The query execution plan is printed to logs.
- **off**: The query execution plan is not printed to logs.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- Debugging information about **debug_print_parse**, **debug_print_rewritten**, and **debug_print_plan** is printed only when the log level is set to **log** or higher. When these parameters are set to **on**, their debugging information will be recorded in server logs and will not be sent to client logs. You can change the log level by setting **client_min_messages** and **log_min_messages**.
- Do not call the **gs_encrypt_aes128** and **gs_decrypt_aes128** functions when **debug_print_plan** is set to **on**, preventing the risk of sensitive information disclosure. You are advised to filter parameter information of the **gs_encrypt_aes128** and **gs_decrypt_aes128** functions in the log files generated when **debug_print_plan** is set to **on** before providing the log files to external maintenance engineers for fault locating. After you finish using the logs, delete them as soon as possible.

debug_pretty_print

Parameter description: If this parameter is enabled, logs generated by **debug_print_parse**, **debug_print_rewritten**, and **debug_print_plan** are indented and output in a longer format that is easy to read.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The indentation is enabled.
- **off**: The indentation is disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_checkpoints

Parameter description: Specifies whether the statistics on checkpoints and restart points are recorded in the server logs. When this parameter is set to **on**, statistics on checkpoints and restart points are recorded in the log messages, including the number of buffers written and the time spent in writing them.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Statistics on checkpoints and restart points are recorded in the server logs.
- **off**: Statistics on checkpoints and restart points are not recorded in the server logs

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_connections

Parameter description: Specifies whether to record logs of connection requests from the client. Logs include the IP address, port number, username, database name, and duration of key steps in the GaussDB connection setup process.

NOTE

Some client programs (such as gsql) attempt to connect to the database twice when determining whether a password is required. Therefore, the log message may contain duplicate "connection receive" (indicating that a connection request is received).

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the logs are recorded.
- **off** indicates that the logs are not recorded.

Default value: **off**

Setting method: This is a BACKEND parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_disconnections

Parameter description: Specifies whether to record disconnection request information of the client.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The information is recorded.
- **off**: The information is not recorded.

Default value: **off**

Setting method: This is a BACKEND parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_duration

Parameter description: Specifies whether to record the duration of every completed SQL statement. For clients using extended query protocols, the time required for parsing, binding, and executing steps are logged independently.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The execution time of all completed SQL statements is recorded.
- **off**: The execution time of SQL statements is not recorded.

NOTE

If this parameter is set to **on** or **log_min_duration_statement** is set to **0**, the execution time of each completed SQL statement is recorded. The difference is that **log_min_duration_statement** forcibly records SQL statements.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_error_verbosity

Parameter description: Specifies the content output to the server log for each message that is logged.

Parameter type: enumerated type.

Unit: none

Value range:

- **terse**: The record excludes the DETAIL, HINT, QUERY, and CONTEXT error information.
- **verbose**: The record includes the SQLSTATE error code, the source code file name, function name, and number of the line in which the error occurs.
- **default**: The record includes the DETAIL, HINT, QUERY, and CONTEXT error information, and excludes the SQLSTATE error code, the source code file name, function name, and number of the line in which the error occurs.

Default value: **default**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Set this parameter according to the requirement for the detail level of the log message.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_hostname

Parameter description: By default, connection log messages only show the IP address of the connecting host. If this parameter is enabled, the host name is also recorded. It may take some time to parse the host name. Therefore, the database performance may be affected.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The host name is recorded.
- **off**: The host name is not recorded.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The database performance may be affected when this parameter is enabled.

log_line_prefix

Parameter description: Specifies the prefix format of each log. A prefix is a printf-style string that is output at the beginning of each line of the log. The escape character starting with **%** indicates the status information. For details, see **Table 6-24**.

Table 6-24 Escape characters

Escape Character	Status Information
%a	Application name.
%u	Username.
%d	Database name.
%r	Remote host name or IP address and remote port. If **log_hostname** is set to **off**, only the IP address and remote port are displayed.
%h	Remote host name or IP address. If **log_hostname** is set to **off**, only the IP address is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Escape Character	Status Information
%p	Thread ID.
%t	Timestamp without milliseconds (no time zone in the Windows OS).
%m	Timestamp with milliseconds.
%n	Node to which an error is reported.
%i	Command tag: type of command executed in the current session.
%e	SQLSTATE error code.
%c	Session ID. For details, see the note below the table.
%l	Number of the log line for each session or thread, starting at 1.
%s	Thread startup time.
%v	Virtual transaction ID (backend ID/local XID).
%x	Transaction ID (**0** indicates that no transaction ID is assigned).
%q	Produces no output. If the current thread is a backend thread, this escape sequence is ignored and subsequent escape sequences are processed. Otherwise, this escape sequence and subsequent escape sequences are all ignored.
%S	Session ID.
%T	Trace ID.
%%	The character %.

NOTE

The %c escape character prints a session ID consisting of two 4-byte hexadecimal numbers separated by a period (.). The two hexadecimal numbers indicate the thread startup time and thread ID, respectively. Therefore, %c can also be used to record the thread startup time and thread ID, which saves space. For example, you can use the following query to generate such a hexadecimal session ID from pg_stat_activity:

```
SELECT to_hex(EXTRACT(EPOCH FROM backend_start)::integer) || '.' ||
       to_hex(pid)
FROM pg_stat_activity;
```

- If you set a nonempty value for **log_line_prefix**, you should usually make its last character be a space, to provide visual separation from the rest of the log line. A punctuation character can be used, too.
- Syslog generates its own timestamp and thread ID information. Therefore, you do not need to include those escape characters when you are logging in to syslog.

Parameter type: string.

Unit: none

Value range: a string.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **'%m %n %u %d %h %p %S %x %a '**

NOTE

%m %n %u %d %h %p %S %x %a indicates the session start timestamp, error reporting node, username, database name, remote host name or IP address, thread ID, session ID, transaction ID, and application name.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the requirements for log status information.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_lock_waits

Parameter description: When the time for a session to wait for a lock exceeds the value of **deadlock_timeout**, this parameter specifies whether to record the message in the database log. This helps determine whether the lock wait event causes problems such as deadlock.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The information is recorded.
- **off**: The information is not recorded.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_statement

Parameter description: Specifies which SQL statements are recorded. For clients using extended query protocols, logging occurs when an execute message and values (enclosed by a pair of single quotation marks) of bound parameters are received.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- Statements that contain simple syntax errors are not logged even if **log_statement** is set to **all** because logging occurs only after basic parsing has been completed and the statement type is determined. If an extended query protocol is used, statements that fail before the execution phase (during parsing or planning) are not logged, either. Set **log_min_error_statement** to **ERROR** or lower to log such statements.
- If this parameter is set to a value other than **none**, the statement audit function is enabled. The database administrator can access server logs to view SQL execution records.

Parameter type: enumerated type.

Unit: none

Value range:

- **none**: No SQL statement is recorded.
- **ddl**: All data definition statements, such as CREATE, ALTER, and DROP, are recorded.
- **mod**: All DDL statements and data modification statements, such as INSERT, UPDATE, DELETE, TRUNCATE, and COPY FROM, are recorded.
- **all**: All statements, including the PREPARE, EXECUTE, and EXPLAIN ANALYZE statements, are recorded.

Default value: **none**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on whether SQL statements need to be recorded.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_temp_files

Parameter description: Specifies whether to record the deletion information about temporary files whose size is greater than the value of this parameter. Temporary files can be created for sorting, hashing, and storing temporary querying results. If the recording is enabled, a log entry is generated for each temporary file when it is deleted.

Parameter type: integer.

Unit: KB

Value range: –1 to 2147483647

- A positive value indicates that the deletion information of temporary files whose size is larger than the value specified by **log_temp_files** is recorded.
- **0** indicates that the delete information of all temporary files is recorded.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **-1** indicates that the delete information of any temporary files is not recorded.

Default value: **-1**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**. If the value is an integer, the default unit is KB. If the value contains a unit, the unit must be KB, MB, or GB. For example, if the value is **16384** without a unit, **log_temp_files** indicates 16384 KB. If the value is **16MB**, **log_temp_files** indicates 16 MB.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a value other than –1, more temporary files that meet the recording requirements indicate more logs are recorded, affecting the system performance. If the value is out of the range or the value contains a unit other than KB, MB, and GB, an error message is displayed.

log_timezone

Parameter description: Specifies the time zone used for timestamps written in the server log. Different from **TimeZone**, this parameter takes effect for all sessions in the database.

Parameter type: string.

Unit: none

Value range: a string. You can obtain it by querying the **PG_TIMEZONE_NAMES** view.

Default value: Set this parameter based on the OS time zone.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

The default value will be changed when **gs_initdb** is used to set system environments.

logging_module

Parameter description: Specifies whether module logs are output on the server. This parameter depends on the setting of the **log_min_messages** parameter. Module logs whose levels are lower than the value of **log_min_messages** will not be printed.

Parameter type: string.

Unit: none

Value range:

ALL,on(),off(COMMAND,DFS,GUC,GSCLEAN,HDFS,ORC,SLRU,MEM_CTL,AUTOVAC,CACHE,ADIO,SSL,GDS,TBLSPC,WLM,OBS,INDEX,EXECUTOR,OPFUSION,GPC,GSC,VEC_EXECUTOR,STREAM,LLVM,OPT,OPT_REWRITE,CB

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
OPT_REWRITE,OPT_JOIN,OPT_AGG,OPT_CHOICE,OPT_SUBPLAN,OPT_SETOP,OPT_SKEW,OPT_PLANNER,DIST
OPT_PLANNER,OPT_STAT_EXT,OPT_SHIPPING,OPT_DSL_REWRITE,SPM,SPM_BG,SPM_KEY_FLOW,UDF,COOP_
ANALYZE,WLMCP,ACCELERATE,MOT,PLANHINT,PARQUET,PGSTAT,CARBONDATA,SNAPSHOT,XACT,HANDLE,C
LOG,EC,REMOTE,CN_RETRY,PLSQL,PLSQL_COMPILE,PLSQL_TRANS,TEXTSEARCH,SEQ,REDO,FUNCTION,PARS
ER,INSTR,WDR_SNAPSHOT,GSMONITOR,WDR_REPORT,ASP_REPORT,BLACKBOX,INCRE_CKPT,INCRE_BG_WRI
TER,DBL_WRT,RTO_RPO,HEARTBEAT,COMM_IPC,COMM_PARAM,TIMESERIES,SCHEMA,GTT,SEGMENT_PAGE,L
IGHTPROXY,HOTKEY,THREAD_POOL,OPT_AI,WALRECEIVER,USTORE,UPAGE,UBTREE,UNDO,TIMECAPSULE,GE
N_COL,DCF,AI4DB,DB4AI,ABO,MOD_ABOFEEDBACK,ASCEND,PLDEBUGGER,ADVISOR,SEC,SEC_FE,SEC_LEGER,
SEC_POLICY,SEC_SDD,SEC_TDE,COMM_FRAMEWORK,COMM_PROXY,COMM_POOLER,COMM_STATUS,VACU
UM,JOB,SPI,NEST_COMPILE,RESOWNER,GSSTACK,LOGICAL_DECODE,GPRC,DISASTER_READ,STANDBY_READ,
SQL_APPLY,REPSYNC,SQLPATCH,PARTITION,UBT_NEWPAGE,GS_DEPENDENCY,GPI,LWLOCK,LOCK,UNIQUE_S
QL,GSI,GLC,SRF,DBLINK,BARRIER_CREATOR,EXRTO_PAGE_P,SEQ_TUP_P,BT_TUP_P,DISPATCH_VERIFY,HBKT,HB
KT_CBI,DBE_STATS,DBE_XMLGEN,MESSAGE_QUEUE,DBE_ALERT,UPDATE_DIST_COL,GS_ILM,GSPERF,PARAME
TERIZE,EXEC_REMOTE,UBTREE_PARA,BTREE_PARA,ANTI_CACHE,ANTI_RECYCLER,VERIFYLOG,PARALLEL_BUIL
D,GS_REPAIR,AUTHID,CCINDEX,ONLINE_DDL,MTD,EXRTO_BUFFER,WAL,STREAM_PBE,HTAP,UALT,GTM_LITE_
READ_S,UGIN,LOCAL_CACHE,BIINDEX,ONDEMAND,DATEA,TURBO_COMPRESS,GS_SIGNAL,AUTOHINT,EXRT
O_READ_RECY,CHARACTER,BUFFILE,BUFFER,PAGE,SELCACHE,BYTECODE,UTIL,FUNC_DEPEND,MOD_DBE_LO
B,EXEC_CACHE,ROWID,WAL_IMAGE_CHECK,CHECK_XLOG_GEN,SHARE_STORAGE,MOD_TOOLKIT,RESULT_CH
ECK)
```

Table 6-25 Description of some logging_module modules

Module	Description
COMM_IPC	Specifies whether to print the packet sending and receiving status between nodes. This is a switch for locating communication performance problems.
COMM_PARAM	Specifies whether to print the **session** parameter settings during node communication. This is a switch for locating communication performance problems.

NOTICE

CN_RETRY and **BYTECODE** do not take effect in the current version.

Default value: **"off(ALL)"**. That is, all module logs are disabled by default.

Setting method: This is a session-level parameter, and you are advised not to use the gs_guc tool to set it.

You can run **SHOW logging_module** to check which modules are controllable. For example, the query output result is as follows:

```
gaussdb=# show logging_module;
logging_module
-------------------
```

```
ER,INSTR,WDR_SNAPSHOT,GSMONITOR,WDR_REPORT,ASP_REPORT,BLACKBOX,INCRE_CKPT,INCRE_BG_WRI
TER,DBL_WRT,RTO_RPO,HEARTBEAT,COMM_IPC,COMM_PARAM,TIMESERIES,SCHEMA,GTT,SEGMENT_PAGE,L
IGHTPROXY,HOTKEY,THREAD_POOL,OPT_AI,WALRECEIVER,USTORE,UPAGE,UBTREE,UNDO,TIMECAPSULE,GE
N_COL,DCF,AI4DB,DB4AI,ABO,MOD_ABOFEEDBACK,ASCEND,PLDEBUGGER,ADVISOR,SEC,SEC_FE,SEC_LEGER,
SEC_POLICY,SEC_SDD,SEC_TDE,COMM_FRAMEWORK,COMM_PROXY,COMM_POOLER,COMM_STATUS,VACU
UM,JOB,SPI,NEST_COMPILE,RESOWNER,GSSTACK,LOGICAL_DECODE,GPRC,DISASTER_READ,STANDBY_READ,
SQL_APPLY,REPSYNC,SQLPATCH,PARTITION,UBT_NEWPAGE,GS_DEPENDENCY,GPI,LWLOCK,LOCK,UNIQUE_S
QL,GSI,GLC,SRF,DBLINK,BARRIER_CREATOR,EXRTO_PAGE_P,SEQ_TUP_P,BT_TUP_P,DISPATCH_VERIFY,HBKT,HB
KT_CBI,DBE_STATS,DBE_XMLGEN,MESSAGE_QUEUE,DBE_ALERT,UPDATE_DIST_COL,GS_ILM,GSPERF,PARAME
TERIZE,EXEC_REMOTE,UBTREE_PARA,BTREE_PARA,ANTI_CACHE,ANTI_RECYCLER,VERIFYLOG,PARALLEL_BUIL
D,GS_REPAIR,AUTHID,CCINDEX,ONLINE_DDL,MTD,EXRTO_BUFFER,WAL,STREAM_PBE,HTAP,UALT,GTM_LITE_
READ_S,UGIN,LOCAL_CACHE,BIINDEX,ONDEMAND,DATEA,TURBO_COMPRESS,GS_SIGNAL,AUTOHINT,EXRT
O_READ_RECY,CHARACTER,BUFFILE,BUFFER,PAGE,SELCACHE,BYTECODE,UTIL,FUNC_DEPEND,MOD_DBE_LO
B,EXEC_CACHE,ROWID,WAL_IMAGE_CHECK,CHECK_XLOG_GEN,SHARE_STORAGE,MOD_TOOLKIT,RESULT_CH
ECK)
(1 row)
```

Controllable modules are identified by uppercase letters, and the special ID **ALL** is used for setting all module logs. You can control the output of module logs by setting the parameter to **on** or **off**. Enable log output for SSL:

```
gaussdb=# set logging_module='on(SSL)';
SET
gaussdb=# show
logging_module;
      logging_module
-------------------
```

```
ANALYZE,WLMCP,ACCELERATE,MOT,PLANHINT,PARQUET,PGSTAT,CARBONDATA,SNAPSHOT,XACT,HANDLE,C
LOG,EC,REMOTE,CN_RETRY,PLSQL,PLSQL_COMPILE,PLSQL_TRANS,TEXTSEARCH,SEQ,REDO,FUNCTION,PARS
ER,INSTR,WDR_SNAPSHOT,GSMONITOR,WDR_REPORT,ASP_REPORT,BLACKBOX,INCRE_CKPT,INCRE_BG_WRI
TER,DBL_WRT,RTO_RPO,HEARTBEAT,COMM_IPC,COMM_PARAM,TIMESERIES,SCHEMA,GTT,SEGMENT_PAGE,L
IGHTPROXY,HOTKEY,THREAD_POOL,OPT_AI,WALRECEIVER,USTORE,UPAGE,UBTREE,UNDO,TIMECAPSULE,GE
N_COL,DCF,AI4DB,DB4AI,ABO,MOD_ABOFEEDBACK,ASCEND,PLDEBUGGER,ADVISOR,SEC,SEC_FE,SEC_LEGER,
SEC_POLICY,SEC_SDD,SEC_TDE,COMM_FRAMEWORK,COMM_PROXY,COMM_POOLER,COMM_STATUS,VACU
UM,JOB,SPI,NEST_COMPILE,RESOWNER,GSSTACK,LOGICAL_DECODE,GPRC,DISASTER_READ,STANDBY_READ,
SQL_APPLY,REPSYNC,SQLPATCH,PARTITION,UBT_NEWPAGE,GS_DEPENDENCY,GPI,LWLOCK,LOCK,UNIQUE_S
QL,GSI,GLC,SRF,DBLINK,BARRIER_CREATOR,EXRTO_PAGE_P,SEQ_TUP_P,BT_TUP_P,DISPATCH_VERIFY,HBKT,HB
KT_CBI,DBE_STATS,DBE_XMLGEN,MESSAGE_QUEUE,DBE_ALERT,UPDATE_DIST_COL,GS_ILM,GSPERF,PARAME
TERIZE,EXEC_REMOTE,UBTREE_PARA,BTREE_PARA,ANTI_CACHE,ANTI_RECYCLER,VERIFYLOG,PARALLEL_BUIL
D,GS_REPAIR,AUTHID,CCINDEX,ONLINE_DDL,MTD,EXRTO_BUFFER,WAL,STREAM_PBE,HTAP,UALT,GTM_LITE_
READ_S,UGIN,LOCAL_CACHE,BIINDEX,ONDEMAND,DATEA,TURBO_COMPRESS,GS_SIGNAL,AUTOHINT,EXRT
O_READ_RECY,CHARACTER,BUFFILE,BUFFER,PAGE,SELCACHE,BYTECODE,UTIL,FUNC_DEPEND,MOD_DBE_LO
B,EXEC_CACHE,ROWID,WAL_IMAGE_CHECK,CHECK_XLOG_GEN,SHARE_STORAGE,MOD_TOOLKIT,RESULT_CH
ECK)
(1 row)

gaussdb=# set logging_module='on(ALL)';
SET
gaussdb=# show
logging_module;
       logging_module
-------------------
```

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Logs are printed.
- **off**: Logs are not printed.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

opfusion_debug_mode

Parameter description: Checks whether simple queries are optimized for debugging. If this parameter is set to **log**, you can view the specific reasons why queries are not optimized in the DN execution plans.

Parameter type: enumerated type.

Unit: none

Value range:

- **off**: This function is disabled.
- **log**: This function is enabled. You can view the specific reasons why queries are not optimized in the DN execution plans.

NOTICE

- You need to set **max_datanode_for_plan** to view the DN execution plans.
- To view the reasons why queries are not optimized in the log, set the parameter to **log**, **log_min_messages** to **debug4**, and **logging_module** to **'on(OPFUSION)'**. Note that a large number of log messages may be generated. Therefore, execute only a small number of jobs during debugging.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_debug_vacuum

Parameter description: Specifies whether to allow output of some VACUUM-related logs for problem locating. This parameter is used only by developers. Common users are advised not to use it.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_err_stack

Parameter description: Specifies whether to print stack information for errors whose **ACTION** is "Contact Huawei technical support."

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_show_err_code

Parameter description: Specifies whether to display error codes for frontend connections such as gsql.

Parameter type: Boolean.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: enabled.
- **off**: disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.9.4 Using CSV Log Output

Prerequisites

- The **log_destination** parameter is set to **csvlog**.
- The **logging_collector** parameter is set to **on**.

Definition of csvlog

Log lines are emitted in comma separated values (CSV) format.

An example table definition for storing CSV-format log output is shown as follows:

```
CREATE TABLE gaussdb_log
(
log_time timestamp(3) with time zone,
node_name text,
user_name text,
database_name text,
process_id bigint,
connection_from text,
"session_id" text,
session_line_num bigint,
command_tag text,
session_start_time timestamp with time zone,
virtual_transaction_id text,
transaction_id bigint,
query_id bigint,
module text,
error_severity text,
sql_state_code text,
message text,
detail text,
hint text,
internal_query text,
internal_query_pos integer,
context text,
query text,
query_pos integer,
location text,
application_name text
);
```

For details, see **Table 6-26**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-26 Meaning of each csvlog field

Field	Description	Field	Description
log_time	Timestamp in milliseconds	module	Log module
node_name	Node name	error_severity	ERRORSTATE code
user_name	Username	sql_state_code	SQLSTATE code
database_name	Database name	message	Error message
process_id	Thread ID	detail	Detailed error message
connection_from	Port number of the client host	hint	Prompt message
session_id	Session ID	internal_query	Internal query (This field is used to query the information leading to errors if any.)
session_line_num	Number of lines in each session	internal_query_pos	Pointer for an internal query
command_tag	Command tag	context	Environment
session_start_time	Start time of a session	query	Character count at the position where errors occur
virtual_transaction_id	Regular transaction	query_pos	Pointer at the position where errors occur
transaction_id	Transaction ID	location	Position where errors occur in the GaussDB source code if **log_error_verbosity** is set to **verbose**.
query_id	Query ID	application_name	Application name

Run the **COPY FROM** command to import a log file to this table.

```
COPY postgres_log FROM '/opt/data/gs_log/logfile.csv' WITH csv;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

The log name (**logfile.csv**) here needs to be replaced with the name of a log generated.

Simplifying Input

Simplify importing CSV log files by performing the following operations:

- Set **log_filename** and **log_rotation_age** to provide a consistent and predictable naming solution for log files. By doing this, you can predict when an individual log file is complete and ready to be imported.
- Set **log_rotation_size** to **0** to disable size-based log rollback, as it makes the log file name difficult to predict.
- Set **log_truncate_on_rotation** to **on** so that old log data cannot be mixed with the new one in the same file.

6.3.10 Alarm Reporting

During the running of cluster, error can be reported and informed to users in a timely manner. You can view system_alarm logs in *$GAUSSLOG*/**cm**, *$GAUSSLOG*/**gs_log/gtm**, *$GAUSSLOG*/**gs_log**, or *$GAUSSLOG*/**roach/agent**.

enable_alarm

Parameter description: Specifies whether to enable the alarm reporting thread to report possible fault scenarios in the database.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The alarm reporting thread is enabled.
- **off**: The alarm reporting thread is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, users cannot perceive some fault scenarios in the database.

NOTE

This parameter takes effect only on CNs and DNs.

connection_alarm_rate

Parameter description: Specifies the ratio restriction on the maximum number of allowed parallel connections to the database. The maximum number of concurrent connections to the database is **max_connections** x **connection_alarm_rate**. When the number of concurrent connections exceeds

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

max_connections x **connection_alarm_rate**, the backend prints the connection information log (**connection_log**), you can view the pg_stat_activity view and thread pool view LOCAL_THREADPOOL_STATUS in logs. (To view the pg_stat_activity view, you need to enable the tracing function. To view the thread pool view LOCAL_THREADPOOL_STATUS, you need to enable the thread pool function.)

Parameter type: floating-point number

Unit: none

Value range: 0 to 1

Default value: **0.9**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

alarm_report_interval

Parameter description: Specifies the interval at which an alarm is reported.

Parameter type: integer.

Unit: seconds (s)

Value range: 0 to 2147483647

Default value: **10**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings:

- If the value is too large and the same alarm is detected for multiple times within the specified alarm reporting interval, key alarms will be lost.
- If the value is too small and the same alarm is detected continuously within the specified alarm reporting interval, the alarm is reported frequently.

alarm_component

Parameter description: Certain alarms are suppressed during alarm reporting. That is, the same alarm will not be repeatedly reported by an instance within the period specified by **alarm_report_interval**. Its default value is **10s**. In this case, the parameter specifies the location of the alarm component that is used to process alarm information. Only the SYSADMIN user can access this parameter.

Parameter type: string.

Unit: none

Value range: valid file name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If **--alarm-type** in the gs_preinstall script is set to **5**, no third-party component is connected and alarms are written into system_alarm logs. In this case, the value of **alarm_component** is **/opt/huawei/snas/bin/snas_cm_cmd**.
- If **--alarm-type** in the gs_preinstall script is set to **1**, a third-party component is connected. In this case, the value of **alarm_component** is the absolute path of the executable program of the third-party component.

Default value: **"/opt/huawei/snas/bin/snas_cm_cmd"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

table_skewness_warning_threshold

Parameter description: Specifies the threshold for triggering a table skew alarm.

Parameter type: floating-point number

Unit: none

Value range: 0 to 1

Default value: **1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. You can adjust the value based on the sensitivity to table skew in service scenarios.

Risks and impacts of improper settings: If this parameter is set to a small value, the system may be more sensitive to table skew. There is no significant risk.

table_skewness_warning_rows

Parameter description: Specifies the number of rows for triggering a table skew alarm.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **100000**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. You can adjust the value based on the sensitivity to table skew in service scenarios.

Risks and impacts of improper settings: If this parameter is set to a small value, the system may be more sensitive to table skew. There is no significant risk.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3.11 Statistics During the Database Running

6.3.11.1 Query and Index Statistics Collector

The query and index statistics collector is used to collect statistics during database running. The statistics include the times of inserting and updating a table and index, the number of disk blocks and tuples, and the time required for the last cleanup and analysis on each table. The statistics can be viewed by querying the system view pg_stats and the system catalog pg_statistic. The following parameters are used to set the statistics collection feature in the server scope:

track_activities

Parameter description: Specifies whether to collect statistics about the commands that are being executed in each session. For a stored procedure, if this parameter is enabled, you can view the PERFORM statement, stored procedure calling statement, SQL statement, and OPEN CURSOR statement that are being executed in the stored procedure in the pg_stat_activity view.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The statistics collection function is enabled.
- **off**: The statistics collection function is disabled.

Default value: **on**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **off**, the space recycling capability of the storage engine is affected, causing space expansion.

track_counts

Parameter description: Specifies whether to collect statistics about database activities. Database statistics are required when the autovacuum process checks for databases that need to be vacuumed. Therefore, the default value is set to **on**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The statistics collection function is enabled.
- **off**: The statistics collection function is disabled.

Default value: **on**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **off**, the space recycling capability of the storage engine is affected, causing space expansion.

track_procedure_sql

Parameter description: Specifies whether the SQL statements that are being executed in the stored procedure are printed in the **query** column in the pg_stat_activity system view.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: When a stored procedure is called, the statements that are being executed by the stored procedure are printed in the query column of pg_stat_activity.
- **off**: When a stored procedure is called, only statements for calling the stored procedure are printed in the query column of pg_stat_activity.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

track_io_timing

Parameter description: Specifies whether to collect statistics about I/O timing in the database. The I/O timing statistics can be queried by using the **pg_stat_database** parameter. When this parameter is enabled, the collector repeatedly queries the OS information at the current time, which may cause significant cost on some platforms. Therefore, the default value is **off**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The statistics collection function is enabled.
- **off**: The statistics collection function is disabled.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

track_functions

Parameter description: Specifies whether to collect statistics of the number and duration of function calls.

NOTE

When the SQL functions are set to inline functions queried by the calling, these SQL functions cannot be traced no matter the parameter is enabled or not.

Parameter type: enumerated type

Unit: none

Value range:

- **pl**: Only procedural language functions are traced.
- **all**: SQL language functions are traced.
- **none**: The function tracing function is disabled.

Default value: **none**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

track_activity_query_size

Parameter description: Specifies byte counts of the current running commands used to trace each active session. If the actual number of bytes in a command is greater than the value of this parameter, the command is truncated.

Parameter type: integer

Unit: bytes

Value range: 100 to 102400

Default value: 1024

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the actual service scenario.

Risks and impacts of improper settings: Setting a larger value will increase the shared memory usage. If the value is too large, the memory of other modules may be insufficient. Before modifying this parameter, make sure you understand its meaning and fully verify the change in a test environment.

update_process_title

Parameter description: Specifies whether to control statistics updated with a thread name each time the server receives a new SQL statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The thread name can be viewed on Windows task manager by running the **ps** command.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The statistics collection function is enabled.
- **off**: The statistics collection function is disabled.

Default value: **off**

Setting method: This parameter is a fixed INTERNAL parameter and cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

stats_temp_directory

Parameter description: Specifies the directory for storing temporary statistics. Only the SYSADMIN user can access this parameter.

NOTE

If a RAM-based file system directory is used, the actual I/O cost can be lowered and the performance can be improved.

Parameter type: string.

Unit: none

Value range: valid directory.

Default value: **"pg_stat_tmp"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_gsstat_cache

Parameter description: Specifies whether to enable cache acceleration for PgStat functions.

PgStat functions include pg_stat_get_numscans, pg_stat_get_tuples_returned, pg_stat_get_tuples_fetched, pg_stat_get_tuples_inserted, pg_stat_get_tuples_updated, pg_stat_get_tuples_deleted, pg_stat_get_tuples_hot_updated, pg_stat_get_live_tuples, pg_stat_get_dead_tuples, pg_stat_get_autovacuum_count, pg_stat_get_autoanalyze_count, pg_stat_get_blocks_fetched, pg_stat_get_blocks_hit, pg_timestamp stat_get_last_autoanalyze_time, pg_timestamp stat_get_last_autovacuum_time, and pg_timestamp stat_get_last_updated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The cache acceleration function is enabled for PgStat functions.
- **off**: The cache acceleration function is disabled for PgStat functions.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If this parameter is disabled and the query of PgStat-related views using the preceding functions is slow, you are advised to enable this parameter.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gsstat_cache_size

Parameter description: Specifies the maximum memory occupied by cache acceleration when the statistics of a specific partition are obtained.

Parameter type: integer

Unit: Megabytes

Value range: 1 to 10240

Default value: 1024

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the actual service scenario.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

track_thread_wait_status_interval

Parameter description: Specifies the interval of collecting the thread status information.

Parameter type: integer

Unit: minute

Value range: 0 to 1440

Default value: **30**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **100** without a unit, it indicates 100 minutes. If this parameter is set to **1d**, it indicates 1 day. The unit must be **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_save_datachanged_timestamp

Parameter description: Specifies whether to record the time when **INSERT**, **UPDATE**, **DELETE**, or **EXCHANGE**/**TRUNCATE**/**DROP PARTITION** is performed on table data.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The time when an operation is performed on table data will be recorded.
- **off**: The time when an operation is performed on table data will not be recorded.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

plan_collect_thresh

Parameter description: Collects statistics about the plans that are being executed in each session.

Parameter type: integer

Unit: none

Value range: –1 to 2147483647

- **–1**: Plans that are being executed are not collected.
- **0**: If the current SQL execution time is greater than or equal to the value of **log_min_duration_statement** and the total number of tuples processed by all operators in the plan is greater than or equal to 10000, the system starts to collect plans in running state. Each time the total number of tuples processed by all operators exceeds 10000, a collection is performed.
- Value greater than **0**: When the total number of tuples incrementally returned by all operators in a plan is greater than or equal to the value of this parameter, plans that are being executed are collected once.

Default value: **0**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

track_sql_count

Parameter description: Specifies whether to collect statistics about the statements (SELECT, INSERT, UPDATE, MERGE INTO, and DELETE) that are being executed in a session.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The statistics collection function is enabled.
- **off**: The statistics collection function is disabled.

Default value: **on**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

If this parameter is disabled, the query result is 0.

6.3.11.2 Hotspot Key Statistics

In the distributed architecture, if applications access a node in a short period of time, the resource usage of the node is too high, affecting the normal running of the database. GaussDB provides the function of quickly detecting hotspot keys to quickly determine whether there are hotspot keys and the distribution of hotspot keys.

enable_hotkeys_collection

Parameter description: Specifies whether to automatically collect statistics on the accessed key values in the database.

NOTE

- If you set parameters using gs_guc set, you need to restart the database for the GUC parameters to take effect. During the restart, hotspot key information is cleared.
- When the GUC parameter is disabled, the query result of the hotspot key is empty and a message is displayed indicating that the GUC parameter is disabled. However, when the function is disabled, the hotspot key clearance API can still be used.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The statistics collection function is enabled.
- **off**: The statistics collection function is disabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Enable this parameter when the hotspot detection function is required.

Risks and impacts of improper settings: If this function is enabled, the TPC-C performance slightly deteriorates by less than 5%, and each instance occupies 500 MB memory resources.

6.3.12 Workload Management

If database resource usage is not controlled, concurrent tasks may preempt resources. As a result, the OS will be overloaded and cannot respond to user tasks; or even crash and cannot provide any services to users. The workload and management of GaussDB balance the database workload based on available resources to prevent database overloads.

use_workload_manager

Parameter description: Specifies whether to enable the resource and management function. This parameter must be set to a same value on CNs and DNs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The resource management function is enabled.
- **off**: The resource management function is disabled.

Default value: **on**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this resource management function is disabled, many resource management functions will be affected. Therefore, you are advised not to disable this function.

NOTE

- If method 2 in **Table 6-1** is used to change the parameter value, the new value takes effect only for the threads that are started after the change. In addition, the new value does not take effect for new jobs that are executed by background threads and reused threads. You can make the new value take effect for these threads by using **kill session** or restarting the node.
- After the value of **use_workload_manager** changes from **off** to **on**, statistics about storage resources when **use_workload_manager** was **off** are not collected. To collect statistics about such resources, run the following SQL statement:

```
SELECT gs_wlm_readjust_user_space(0);
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_control_group

Parameter description: Specifies whether to enable the Cgroup function. This parameter must be set to a same value on CNs and DNs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The Cgroups function is enabled.
- **off**: The Cgroups function is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on the corresponding setting method in **Table 6-1**. When you change the parameter value using method 2 listed in the table, the new value takes effect only for the thread that is started after the change.

Setting suggestion: The modification does not take effect for background threads and new jobs that are reused by threads. If you want such threads to identify parameter changes in real time, kill sessions or restart nodes.

Risks and impacts of improper settings: If this parameter and the resource management function (**use_workload_manager**) are enabled at the same time, the user resource management function takes effect, deteriorating the performance of jobs executed by users.

enable_cgroup_switch

Parameter description: Specifies whether a statement is automatically switched to the TopWD group when the statement is executed by control group type.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: A statement is automatically switched to the TopWD group when the statement is executed by control group type.
- **off**: A statement is not automatically switched to the TopWD group when the statement is executed by control group type.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

This parameter is valid only when **enable_control_group** is set to **on**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

cgroup_name

Parameter description: Specifies the name of the Cgroup in use and changes the priority of items in the queue of the Cgroup.

If you set **cgroup_name** and then **session_respool**, the Cgroup associated with **session_respool** takes effect. If you reverse the order, the Cgroup associated with **cgroup_name** takes effect.

If the workload Cgroup level is specified during the **cgroup_name** change, the database does not check the Cgroup level. The level ranges from 1 to 10.

Parameter type: string.

Unit: none

Value range: name of an existing Cgroup.

Default value: **"DefaultClass:Medium"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Do not use both **cgroup_name** and **session_respool**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

DefaultClass:Medium indicates the **Medium** Cgroup that belongs to the **Timeshare** Cgroup under the **DefaultClass** Cgroup.

enable_backend_control

Parameter description: Specifies whether the database permanent thread is bound to the **DefaultBackend** and **HaBackend** Cgroups. This parameter must be set to a same value on CNs and DNs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Permanent threads are bound to the **DefaultBackend** and **HaBackend** Cgroups.
- **off**: Permanent threads are not bound to the **DefaultBackend** and **HaBackend** Cgroups.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

This parameter is valid only when **enable_control_group** is set to **on**. The **HaBackend** Cgroup is used to control the CPU usage of heartbeat threads and WAL-related background threads to ensure high cluster availability.

enable_vacuum_control

Parameter description: Specifies whether the database permanent thread autovacuum worker is bound to the **Vacuum** Cgroup. This parameter must be set to a same value on CNs and DNs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The autovacuum worker thread is bound to the **Vacuum** Cgroup.
- **off**: The autovacuum worker thread is not bound to the **Vacuum** Cgroup.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

This parameter is valid only when **enable_control_group** is set to **on**.

enable_perm_space

Parameter description: Specifies whether to enable the perm space function. This parameter must be set to a same value on CNs and DNs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The perm space management is enabled.
- **off**: The perm space management is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

parctl_min_cost

Parameter description: Specifies the execution cost threshold of a statement. If the execution cost of a statement exceeds the specified value, the statement is subject to the concurrent limit of a resource pool.

Parameter type: integer.

Unit: none

Value range: –1 to 2147483647

- If the value is **-1** or the cost of executing a statement is less than 10, the statement is not subject to the concurrency limit of the resource pool.
- If the value is greater than or equal to **0**, **enable_dynamic_workload** is set to **off**, and the cost of executing a statement exceeds the value and is greater than or equal to 10, the statement is subject to the concurrency limit of the resource pool.

Default value: **100000**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

cpu_collect_timer

Parameter description: Specifies how frequently CPU data is collected during statement execution on DNs.

Parameter type: integer.

Unit: seconds

Value range: 1 to 2147483647

Default value: **30**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: A database administrator should set a proper collection frequency based on system resources (for example, CPU, I/O, and memory resources). A small value will affect the execution efficiency and a large value will reduce the accuracy of exception handling.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

memory_tracking_mode

Parameter description: Specifies the memory information recording mode.

Parameter type: enumerated type

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **none**: Memory statistics are not collected.
- **peak**: Memory statistics on the peak value of the query level are collected. The value is recorded in the database log and can also be output by using **explain analyze**.
- **normal**: Memory statistics are collected in real time but no file is generated.
- **executor**: A statistics file is generated, containing the context information of all allocated memory used on the execution layer.
- **fullexec**: A statistics file is generated, containing the information about all memory contexts requested by the execution layer.

Default value: **none**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

memory_detail_tracking

Parameter description: Specifies the memory context allocation priority of a thread and the plannodeid of the query for which the current thread is running. This parameter is used only in the debug version.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **""**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. You are not allowed to set this parameter.

Risks and impacts of improper settings: none

enable_resource_track

Parameter description: Specifies whether to enable the real-time resource monitoring function. This parameter must be set to a same value on CNs and DNs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The resource monitoring is enabled.
- **off**: The resource monitoring is disabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_resource_record

Parameter description: Specifies whether resource monitoring records are archived. If this parameter is set to **on**, records in the history views (GS_WLM_SESSION_HISTORY and GS_WLM_OPERATOR_HISTORY) are archived to the corresponding info views (GS_WLM_SESSION_INFO and GS_WLM_OPERATOR_INFO) at an interval of 3 minutes. After being archived, the records are deleted from the history views. This parameter must be set to a same value on CNs and DNs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The resource monitoring records are archived.
- **off**: The resource monitoring records are not archived.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_logical_io_statistics

Parameter description: Specifies whether to enable the logical I/O statistics function during resource monitoring. If this function is enabled, the **read_kbytes**, **write_kbytes**, **read_counts**, **write_counts**, **read_speed**, and **write_speed** fields in the PG_TOTAL_USER_RESOURCE_INFO view will collect statistics on the byte count, number of times, and speed of logical read and write. Fields related to logical read and write in the system catalogs GS_WLM_USER_RESOURCE_HISTORY and GS_WLM_INSTANCE_HISTORY will collect statistics on the logical read and write of related users and instances.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The logical I/O statistics function is enabled.
- **off**: The logical I/O statistics function is disabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_instance_metric_persistent

Parameter description: Specifies whether the instance resource monitoring data is dumped. When this parameter is enabled, the instance monitoring data is saved to the system catalog GS_WLM_INSTANCE_HISTORY.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The instance resource monitoring data is dumped.
- **off**: The instance resource monitoring data is not dumped.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

instance_metric_retention_time

Parameter description: Specifies the retention days of the historical monitoring data of instance resources. This parameter is valid only when **enable_instance_metric_persistent** is set to **on**.

Parameter type: integer.

Unit: days

Value range: 0–3650

- If this parameter is set to **0**, the historical monitoring data of instance resources is permanently stored.
- If the value is greater than **0**, the historical monitoring data of instance resources is stored for the specified number of days.

Default value: **7**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The unit must be **d** if required.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

resource_track_level

Parameter description: Specifies the resource monitoring level of the current session. This parameter is valid only when **enable_resource_track** is set to **on**.

Parameter type: enumerated type

Unit: none

Value range:

- **none**: Resources are not monitored.
- **query**: Resources used at the query level are monitored.
- **operator**: Resources used at the query and operator levels are monitored.

Default value: **query**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

resource_track_cost

Parameter description: Specifies the minimum execution cost for resource monitoring on statements in the current session. This parameter is valid only when **enable_resource_track** is set to **on**.

Parameter type: integer.

Unit: none

Value range: –1 to 2147483647

- **-1** indicates that resource monitoring is disabled.
- A value ranging from 0 to 9 indicates that statements whose execution cost is greater than or equal to 10 will be monitored.
- A value greater than or equal to **10** indicates that statements whose execution cost exceeds this value will be monitored.

Default value: **100000**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

resource_track_duration

Parameter description: Specifies the minimum statement execution time that determines whether information about jobs of a statement recorded in the real-time view will be dumped to a historical view after the statement is executed. Job information will be dumped from the real-time view (with the suffix **statistics**) to a historical view (with the suffix **history**) if the statement execution time is no less than this value. This parameter is valid only when **enable_resource_track** is set to **on**.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647

- **0** indicates that historical information about all statements recorded in the real-time resource monitoring view are archived.
- If the value is greater than **0**, historical information about a statement whose execution time exceeds this value will be archived.

Default value: **60**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **3600** without a unit, **resource_track_duration** indicates 3600s. If the value is **1h**, **resource_track_duration** indicates 1 hour. The unit must be **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

dynamic_memory_quota

Parameter description: Specifies the memory quota in adaptive load scenarios, that is, the proportion of maximum available memory to total system memory.

Parameter type: integer.

Unit: none

Value range: 1 to 100

Default value: **80**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

disable_memory_protect

Parameter description: Specifies whether to stop memory protection. To query system views when system memory is insufficient, set this parameter to **on** to stop

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

memory protection. This parameter is used only to diagnose and debug the system when system memory is insufficient. Set it to **off** in other scenarios.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Memory protection is disabled.
- **off**: Memory protection is enabled.

Default value: **off**

Setting method: This parameter is a USERSET parameter and is valid only for the current session. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

query_band

Parameter description: Specifies the user-defined job type of the current session.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **""**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

memory_fault_percent

Parameter description: Specifies the ratio of memory allocation failures during the memory fault test. This parameter is used only in the DEBUG version.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_bbox_dump

Parameter description: Specifies whether to enable the black box function and whether to generate core files when the core mechanism is not configured in the system. This function is valid only for CNs or DNs. For CM Agent, CM Server, GTM, and fenced UDF, the system core mechanism must be configured to capture core files.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The black box function is enabled.
- **off**: The black box function is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

The generation of core files by the black box function depends on the open ptrace API of the OS. If the permission is insufficient (errno = 1), ensure that the **/proc/sys/kernel/yama/ptrace_scope** configuration is correct.

bbox_dump_count

Parameter description: Specifies the maximum number of core files that are generated by GaussDB and can be stored in the path specified by **bbox_dump_path**. If the number of core files exceeds this value, old core files will be deleted. This parameter is valid only when **enable_bbox_dump** is set to **on**.

NOTE

When core files are generated concurrently, the number of files may be larger than the value of **bbox_dump_count**.

Parameter type: integer.

Unit: none

Value range: 1 to 20

Default value: **8**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

bbox_dump_path

Parameter description: Specifies the path where the black box core files are generated. This parameter is valid only when **enable_bbox_dump** is set to **on**.

Parameter type: string.

Unit: none

Value range: valid file path.

Default value: **""**. The default path where the black box core files are generated is **"/proc/sys/kernel/core_pattern"**. If the path is not a directory or you do not have the write permission on the directory, black box core files will be generated under the data directory of the database.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

bbox_blanklist_items

Parameter description: Specifies the anonymized data items of black box core files. This parameter is valid only when **enable_bbox_dump** is set to **on**.

Parameter type: string.

Unit: none

Value range: a string of sensitive data items separated by commas (,).

Currently, the following data items can be anonymized:

- **SHARED_BUFFER**: data buffer.
- **XLOG_BUFFER**: redo log buffer.
- **DW_BUFFER**: doublewrite data buffer.
- **XLOG_MESSAGE_SEND**: buffer for sending primary/standby replication logs.
- **WALRECEIVER_CTL_BLOCK**: buffer for receiving primary/standby replication logs.
- **DATA_MESSAGE_SEND**: buffer for sending primary/standby replication data.
- **DATA_WRITER_QUEUE**: buffer for receiving primary/standby replication data.
- **BUCKET_XLOG_MESSAGE_SEND**: buffer for sending replay logs during bucket scale-out.

Default value: **""**. It indicates that all supported sensitive data items of the core files generated by the black box are anonymized.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_ffic_log

Parameter description: Specifies whether to enable the first failure information capture (FFIC) function. This function is valid only for CNs or DNs. For CM Agent, CM Server, GTM, and fenced UDF, the system core mechanism must be configured to capture core files.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The FFIC function is enabled.
- **off**: The FFIC function is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_dynamic_workload

Parameter description: Specifies whether to enable the memory adaptation function.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The memory adaptation function is enabled.
- **off**: The memory adaptation function is disabled.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If memory adaptation is enabled, you do not need to use **work_mem** to optimize the operator memory usage. The system will generate a plan for each statement based on the current workload, estimating the memory used by each operator and by the entire statement. In a concurrency scenario, statements are queued based on the system workload and their memory usage.
- In some cases, the optimizer cannot accurately estimate the number of rows and thereby underestimates or overestimates memory usage. If the memory usage is underestimated, the allocated memory will be automatically increased during statement running. If the memory usage is overestimated, system memory resources will not be fully used, and the number of statements waiting in a queue will increase, which probably results in low performance. In this case, you need to identify the statements whose estimated memory usage is much greater than the actual DN peak memory and optimize the performance by setting **query_mem**. For details, see "Adjusting Key Parameters During SQL Optimization" in *SQL Optimization Guide*.

enable_reaper_backend

Parameter description: Specifies whether the signal sent by a worker thread when it exits is collected by a separate thread.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: This function is enabled.
- **off**: This function is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

bypass_workload_manager

Parameter description: Specifies whether to enable independent I/O management and control. This parameter must be used together with CNs and DNs.

If **use_workload_manager** is not set to **on**, this parameter can be used to enable the I/O control independently. After the I/O control is enabled, you can set **io_limits** or **io_priority** to configure control details.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The I/O control is enabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **off**: The I/O control is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

io_limits

Parameter description: Specifies the upper limit of input/output operations per second (IOPS).

Parameter type: integer.

Unit: none

Value range: 0 to 1073741823

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

io_priority

Parameter description: Specifies the I/O priority for jobs that consume many I/O resources. It takes effect when the I/O usage reaches 50%.

Parameter type: enumerated type

Unit: none

Value range:

- **None**: no limitation.
- **Low**: The IOPS is reduced to 10% of the original value.
- **Medium**: The IOPS is reduced to 20% of the original value.
- **High**: The IOPS is reduced to 50% of the original value.

Default value: **None**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

io_control_unit

Parameter description: Specifies the unit used to count the number of I/Os during I/O control in row-store scenarios. This parameter must be set to a same value on CNs and DNs.

Set a certain number of I/Os as one unit. This unit is used during the I/O control.

Parameter type: integer.

Unit: none

Value range:1000 to 1000000

Default value: **6000**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

session_respool

Parameter description: Specifies the resource pool associated with the current session.

If you set **cgroup_name** and then **session_respool**, the Cgroup associated with **session_respool** takes effect. If you reverse the order, the Cgroup associated with **cgroup_name** takes effect.

If the workload Cgroup level is specified during the **cgroup_name** change, the database does not check the Cgroup level. The level ranges from 1 to 10.

Parameter type: string.

Unit: none

Value range: name of the resource pool set by running **create resource pool**.

Default value: **"invalid_pool"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Do not use both **cgroup_name** and **session_respool**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

session_statistics_memory

Parameter description: Specifies the memory size of a real-time query view.

Parameter type: integer.

Unit: KB

Value range: 5120 to 2147483647

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: 5120 (that is, 5 MB)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **16384** without a unit, **session_statistics_memory** indicates 16384 KB. If the value is **16MB**, **session_statistics_memory** indicates 16 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: You are advised to set this parameter to a value less than or equal to 50% of the value of **max_process_memory**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

session_history_memory

Parameter description: Specifies the memory size of a historical query view.

Parameter type: integer.

Unit: KB

Value range: 10240 to 2147483647

Default value: **10240** (that is, 10 MB)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **16384** without a unit, **session_history_memory** indicates 16384 KB. If the value is **16MB**, **session_history_memory** indicates 16 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: You are advised to set this parameter to a value less than or equal to 50% of the value of **max_process_memory**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_transaction_parctl

Parameter description: Specifies whether to control transaction block statements and stored procedure statements.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Transaction block statements and stored procedure statements are controlled.
- **off**: Transaction block statements and stored procedure statements are not controlled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

topsql_retention_time

Parameter description: Specifies the retention period of historical TopSQL data in the **gs_wlm_session_query_info_all** and **gs_wlm_operator_info** tables.

Parameter type: integer.

Unit: days

Value range: 0–3650

- If it is set to **0**, the data is stored permanently.
- If the value is greater than **0**, the data is stored for the specified number of days.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

transaction_pending_time

Parameter description: Specifies the maximum queuing time of transaction block statements and stored procedure statements. This parameter is valid only when **enable_transaction_parctl** is set to **on**.

Parameter type: integer.

Unit: seconds

Value range: –1 to 1073741823.

- **-1** or **0** indicates that no queuing timeout is specified for transaction block statements and stored procedure statements. The statements can be executed when resources are available.
- A value greater than **0** indicates that if transaction block statements and stored procedure statements have been queued for a time longer than the specified value, they are forcibly executed regardless of the current resource situation.

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

node_group_mode

Parameter description: Displays the current node group mode.

Parameter type: string.

Unit: none

Value range: name of the node group mode.

Default value: **"node group"**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

resilience_ctrlslot_available_maxpercent

Parameter description: Specifies the maximum percentage of threads in the thread pool that can be occupied by slow SQL statements. This parameter is valid only for SELECT statements executed by non-SYSADMIN/MONADMIN users.

Parameter type: integer.

Unit: none

Value range: 0 to 100

Default value: **100**, indicating that slow SQL statements can occupy all threads in the thread pool.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

resilience_ctrlstmt_detect_timelimit

Parameter description: Specifies the execution time of a normal SQL statement after which it is marked as a slow SQL statement. The value **0** indicates that slow SQL statements are not identified. A value greater than 0 indicates that the SQL statement is marked as a slow SQL statement when its execution time exceeds the value of this parameter. This parameter is valid only for SELECT statements executed by non-SYSADMIN/MONADMIN users.

Parameter type: integer.

Unit: milliseconds

Value range: 0 to 2147483647. **0** indicates that the slow SQL detection function is disabled.

Default value: **0**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **6000** without a unit, **resilience_ctrlstmt_detect_timelimit** indicates 6000 ms. If the value is **6s**, **resilience_ctrlstmt_detect_timelimit** indicates 6s. The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Set this parameter based on the average SQL execution duration of the real-time database.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

resilience_ctrlstmt_control_iopslimit

Parameter description: Specifies the maximum IOPS that can be used by slow SQL statements after normal SQL statements are marked as slow SQL statements. This parameter is valid only for SELECT statements executed by non-SYSADMIN/MONADMIN users.

Parameter type: string.

Unit: none

Value range: **"Low"**, **"Medium"**, **"High"**, **"None"**, or a value ranging from 0 to 2147483647. The lower the level, the stricter the control.

- **"Low"**: low level.
- **"Medium"**: medium level.
- **"High"**: high level.
- **"None"**: Logical I/O control for slow SQL statements is disabled.
- 0 to 2147483647: A smaller value indicates stricter and a larger value indicates looser.

Default value: **"None"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. If this parameter is set to a number, the value must be a character string, for example, **resilience_ctrlstmt_control_iopslimit = '1024'**.

Setting suggestion: Set this parameter to a small value when I/O access is full and to a large value when I/O access is idle.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

session_max_dynamic_memory

Parameter description: Specifies the upper limit of the session-level memory. If the session memory usage exceeds the value of this parameter, an error is reported and the session is interrupted. This parameter is valid only when **enable_memory_limit** is set to **on**.

Parameter type: integer.

Unit: KB

Value range: –1 to 2147483647

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **-1**, indicating no control.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The parameter cannot be set to a small value. If the value is too small, the session fails to execute statements due to insufficient memory. If the parameter is set to a value by using method 3 in **Table 6-2** but the value cannot be set larger due to insufficient memory, you can only exit the session to eliminate the limit. If the value is less than 16 MB, a message is displayed.

max_concurrency

Parameter description: Specifies the upper limit of global concurrency.

Parameter type: integer.

Unit: none

Value range: –1 to 2147483647

Default value: **-1**, indicating no control.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: A database administrator needs to adjust the value of this parameter based on the usage of system resources (such as CPU, I/O, and memory resources) to maximize the number of concurrent jobs and prevent system breakdown caused by excessive concurrent jobs.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_workload_rule

Parameter description: Specifies whether to enable the SQL statement concurrency control.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the SQL statement concurrency control is enabled.
- **off** indicates that the SQL statement concurrency control is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If the SQL statement concurrency control function is not required, set this parameter to **off**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_sysadmin_workload_rule

Parameter description: Specifies whether to enable the SQL statement concurrency control for administrators. This parameter is invalid for superusers, who are never subject to concurrency control rules.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The SQL statement concurrency control for administrators is enabled.
- **off**: The SQL statement concurrency control for administrators is disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value **off**. Set this parameter to **on** only when you need to limit the concurrency of SQL statements executed by administrators.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

workload_uncontrolled_users

Parameter description: Specifies the user whitelist for workload management. Workload management is not performed on users in the whitelist. Currently, this parameter takes effect only for SQL statement concurrency control.

Parameter type: string.

Unit: none

Value range: a list of users separated by commas (,), for example, **user1, user2, user3**.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements. If the users without the SYSADMIN permission do not want to be restricted by workload management when running SQL statements, they can add this parameter.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_emergency_access

Parameter description: Specifies whether to enable the UDS O&M channel, which is called emergency channel.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The emergency channel is enabled.
- **off**: The emergency channel is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after you fully understand the parameter meaning and test the parameter to avoid unexpected results.

max_plancache_memory_ratio

Parameter description: Specifies the dynamic memory usage percentage threshold of the plan cache in session-level prepare statements and **cursor_sharing**.

Parameter type: integer.

Unit: none

Value range: 0 to 100

The value is an integer ranging from 0 to 100, indicating the percentage threshold of the upper limit of the plan cache in the current session to the expected memory reserved for each session. The value **0** indicates that the function is disabled.

The expected dynamic memory size reserved for each session complies with the rules described in **Table 6-27**.

Table 6-27 Dynamic memory reserved for each session

CurConnCount/ max_connections	Dynamic Memory Reserved for Each Session
0 ~ 30%	max_dynamic_memory / (max_connections * 30%)
30% ~ 60%	max_dynamic_memory / (max_connections * 60%)
60% ~ 100%	max_dynamic_memory / (max_connections * 100%)

CurConnCount indicates the number of current sessions. **max_connections** indicates the maximum number of concurrent sessions, which is specified by the GUC parameter **max_connections**. **max_dynamic_memory** indicates the maximum dynamic memory of the system.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

When the system detects that the ratio of the memory occupied by the plan cache to the memory reserved for sessions reaches or exceeds the percentage threshold specified by **max_plancache_memory_ratio**, the cache eviction mechanism is triggered. This mechanism releases plan cache entries using the least recently used (LRU) algorithm until the current plan cache memory usage falls below the threshold. When the cached entries are evicted, the system automatically generates a new execution plan to ensure that the query can be executed properly.

Default value:

- Independent deployment:

  CN: **60** (60-core CPU/480 GB memory, 32-core CPU/256 GB memory, 16-core CPU/128 GB memory); **30** (8-core CPU/64 GB memory); **50** (4-core CPU/32 GB memory); **40** (4-core CPU/16 GB memory)

  DN: **60** (60-core CPU/480 GB memory, 32-core CPU/256 GB memory, 16-core CPU/128 GB memory); **50** (8-core CPU/64 GB memory; 4-core CPU/32 GB memory); **30** (4-core CPU/16 GB memory)

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If this parameter is set to a small percentage, planned cache eviction and rebuilding are frequently triggered, which may cause performance deterioration. Exercise caution when setting this parameter based on the actual memory usage. In the hybrid deployment scenario, you need to adjust the parameter settings based on the memory allocated to a single process. This parameter controls only the dynamic memory usage of the plan cache in session-level prepare statements and **cursor_sharing**. Other memory is not controlled.

The current parameter settings are based on the memory configuration, number of connections, and performance. If the plan memory usage is too high, **max_dynamic_memory** may be used up, or memory eviction may be triggered, causing performance deterioration. In this special scenario, you are advised to adjust the memory ratio based on services.

In addition, if the GUC parameter **enable_opfusion** is set to **on**, the session-level opfusion memory is not specified by this parameter. As a result, this memory may not be controlled. If this problem occurs, you are advised to decrease the value of the GUC parameter **max_plancache_memory_ratio** to alleviate this problem.

Risks and impacts of improper settings: If the ratio set by this parameter is too small, the plan cache eviction and rebuild are frequently triggered, which may deteriorate the performance. If the ratio set by this parameter is too large, memory resources may be insufficient and resources may be unavailable. Before modifying this parameter, make sure you understand its meaning and fully verify the change in a test environment.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This parameter takes effect only when the parameter **enable_memory_limit** is enabled.
- The maximum dynamic memory (**max_dynamic_memory**) can be queried in the pv_total_memory_detail view. The GUC parameter **max_connections** indicates the maximum number of connections.
- Cache eviction is triggered only when the memory size reaches or exceeds the upper limit after the execution plan is created or rebuilt. When a new session is connected and the percentage changes in the range described in **Table 6-27** (for example, from 29% to 31%), the memory reserved for each session changes accordingly. Note that eviction is not triggered.
- After the version upgrade, this parameter is disabled by default, that is, the value is **0**.

global_plancache_memory_ratio

Parameter description: Specifies the percentage threshold of the dynamic memory occupied by the global shared plan cache. The value **0** indicates that the GPC memory control is disabled.

Parameter type: integer

Unit: none

Value range: 0 to 80

Default value:

- Independent deployment:

  CN: **30** (60-core CPU/480 GB memory, 32-core CPU/256 GB memory, 16-core CPU/128 GB memory); **20** (8-core CPU/64 GB memory); **30** (4-core CPU/32 GB memory); **20** (4-core CPU/16 GB memory)

  DN: **30** (60-core CPU/480 GB memory, 32-core CPU/256 GB memory, 16-core CPU/128 GB memory, 8-core CPU/64 GB memory, 4-core CPU/32 GB memory); **30** (4-core CPU/16 GB memory)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If the percentage specified by this parameter is too small, the plan cache may fail to be shared. Set this parameter based on the actual memory usage.

You are advised to adjust the parameter settings based on the memory specifications. For details, see the egress parameter settings.

In the hybrid deployment scenario, you need to adjust the parameter settings based on the memory allocated to a single process.

The current parameter settings are based on the memory configuration and performance. If the plan memory usage is too high, **max_dynamic_memory** may be used up, or memory eviction may be triggered, causing performance deterioration. In this special scenario, you are advised to adjust the memory ratio based on services.

Risks and impacts of improper settings: If the percentage specified by this parameter is too small, a large number of plan caches cannot be shared. If the percentage is too large, memory resources may be insufficient and there is still a

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

risk that resources are unavailable. Before modifying this parameter, make sure you understand its meaning and fully verify the change in a test environment.

NOTE

This parameter takes effect only when the parameter **enable_memory_limit** is enabled.

current_logic_cluster

Parameter description: Displays the name of the current logical database instance.

Parameter type: string.

Unit: none

Value range: logical database instance name.

Default value: ""

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

6.3.13 Autovacuum

The autovacuum thread includes the VACUUM and ANALYZE processes to recycle the record space marked as deleted and update statistics about the table. The **autovacuum_naptime** parameter specifies the interval between them. The default interval is 10 minutes. You can set the interval based on the actual situation. The interval cannot be completely accurate. It depends on the hardware conditions and load of the environment. If a large amount of data needs to be vacuumed, the heavy load may delay the process. It also depends on the values of **autovacuum_naptime** and **autovacuum_max_workers**. **autovacuum_naptime** specifies the execution interval. A smaller value indicates a shorter execution interval. However, data vacuuming and statistics calculation will increase the CPU usage, memory usage, and I/O overhead. **autovacuum_max_workers** specifies the maximum number of concurrent autovacuum threads. A larger value makes the parameter more compatible with **autovacuum_naptime**, but the CPU usage, memory usage, and I/O overhead also increase.

autovacuum

Parameter description: Specifies whether to start the autovacuum thread in the database. Ensure that the **track_counts** parameter is set to **on** before starting the autovacuum thread.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the autovacuum thread is started.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **off** indicates that the autovacuum thread is not started.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- Set the **autovacuum** parameter to **on** if you expect the system to automatically clear up two-phase transactions after recovering from faults.
- If **autovacuum** is set to **on** and **autovacuum_max_workers** to **0**, the system does not perform autovacuum. Instead, the system automatically vacuum two-phase transactions after recovering from faults.
- If **autovacuum** is set to **on** and the value of **autovacuum_max_workers** is greater than **0**, the system will automatically vacuum the two-phase transactions and processes after recovering from faults.

Risks and impacts of improper settings: If this parameter is disabled in an Astore table, too many inactive rows will exist in the table, which may affect the query performance.

autovacuum_mode

Parameter description: Specifies whether the autoanalyze and autovacuum function are enabled. This parameter is valid only when **autovacuum** is set to **on**.

Parameter type: enumerated type.

Unit: none

Value range:

- **analyze** indicates that only autoanalyze is performed.
- **vacuum** indicates that only autovacuum is performed.
- **mix** indicates that both autoanalyze and autovacuum are performed.
- **none** indicates that neither autoanalyze nor autovacuum is performed.

Default value: **mix**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

autoanalyze_timeout

Parameter description: Specifies the timeout interval of autoanalyze. If the duration of autoanalyze on a table exceeds the value of **autoanalyze_timeout**, the autoanalyze operation is automatically canceled.

The timeout check cannot be completely accurate. In principle, the statistics on each CN must be consistent. Therefore, the synchronization between CNs will not

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

be interrupted even if the synchronization times out. As a result, the actual execution time may exceed the user-defined time.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483. The value **0** indicates that no timeout occurs.

Default value: **300**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Set the value depending on the database load and performance requirements. If the database load is light, you can set **autoanalyze_timeout** to a larger value so that the automatic analysis operation can be completed in sufficient time. If the database is heavily loaded, set **autoanalyze_timeout** to a smaller value so that automatic analysis can be completed in a short period of time. This prevents other important database operations from being affected.

Risks and impacts of improper settings: The AUTOANALYZE operation consumes certain system resources. If the value is too large, system resources may be excessively consumed, affecting database performance. If the value is too small, the AUTOANALYZE operation may fail to be completed. As a result, the statistics may be inaccurate, affecting the execution plan and performance of the query.

autovacuum_io_limits

Parameter description: Specifies the upper limit of I/Os triggered by the autovacuum thread per second.

Parameter type: integer.

Unit: none

Value range: –1 to 1073741823. **-1** indicates that the default Cgroup is used.

Default value: **-1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- If the disk I/O resources of server are sufficient, you can set **autovacuum_io_limits** to a large value, which allows the autovacuum thread to complete the cleanup operation more quickly.
- If the disk I/O resources of the server are limited, you can set **autovacuum_io_limits** to a small value. This limits the disk I/O resources used by the autovacuum thread to avoid impact on other threads.

Risks and impacts of improper settings:

- If the value is too large, the autovacuum thread may use too many disk I/O resources, affecting the performance of other threads.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If the value is too small, the autovacuum thread may fail to effectively clear table data. As a result, table data is excessively expanded, affecting database performance.

log_autovacuum_min_duration

Parameter description: Records each step performed by the autovacuum process to the server log when the execution time of the autovacuum process is greater than or equal to a certain value. This parameter helps track the autovacuum behavior.

For example, set the **log_autovacuum_min_duration** parameter to **250ms** to record the information about the autovacuum commands running longer than or equal to 250 ms.

Parameter type: integer.

Unit: ms

Value range: –1 to 2147483647

- **0** indicates that all autovacuum actions are recorded in the log.
- **-1** indicates that all autovacuum actions are not recorded in the log.
- A value greater than or equal to 0 indicates that a message is recorded when an autovacuum action is skipped due to a lock conflict. (The reason for skipping autovacuum is recorded for audit.)

Default value: **-1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **600** without a unit, **log_autovacuum_min_duration** indicates 600 ms. If the value is **10min**, **log_autovacuum_min_duration** indicates 10 minutes. The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: The default value is **-1**. If you want to monitor the performance of automatic VACUUM operations, set **log_autovacuum_min_duration** to a value greater than 0 to record the automatic VACUUM operations whose duration exceeds the value of this parameter.

Risks and impacts of improper settings: If this parameter is set to **0** or the value is too small, a large number of logs may be recorded, affecting system performance. If this value is too large, the system may not detect that the automatic clearance time is too long in a timely manner, causing insufficient clearance speed or exceptions.

autovacuum_max_workers

Parameter description: Specifies the maximum number of autovacuum threads running at the same time.

Parameter type: integer.

Unit: none

Value range: The minimum value is **0**, indicating that autovacuum is not automatically performed. Theoretically, the maximum value is **1024**. If the value is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

greater than 1024, the value is automatically changed to **1024**. However, the value may still exceed the actual upper limit. The actual upper limit of this parameter is related to the GUC parameters **max_connections** and **job_queue_processes**. The calculation formula is as follows: 262143 – **max_inner_tool_connections** – **max_connections** – **max_concurrent_autonomous_transactions** – **job_queue_processes** – Number of auxiliary threads – Number of launcher threads of autovacuum [x(1] – 1. The number of auxiliary threads and the number of launcher threads of autovacuum are specified by two macros and their default values are **20** and **2**, respectively.

Default value: **3**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, more autovacuum processes are created and more CPU and memory resources are occupied. Therefore, you are advised not to set this parameter to a large value. Otherwise, memory cannot be allocated or too many CPU resources are occupied, causing database startup errors or affecting services. If the value is too small, the autovacuum thread may fail to clear and recycle tablespaces that are no longer used in a timely manner, causing database expansion and performance deterioration. The actual value of this parameter needs to be evaluated based on the actual upper limit of the value range. Generally, the value can reach **1024**. If the value exceeds the actual upper limit, the setting fails. This parameter takes effect for new threads but does not take effect for running threads.

autovacuum_naptime

Parameter description: Specifies the interval between activity rounds for the autovacuum process.

Parameter type: integer.

Unit: seconds

Value range: 1 to 2147483

Default value: **600** (that is, 10 minutes)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **600** without a unit, **autovacuum_naptime** indicates 600s. If the value is **10min**, **autovacuum_naptime** indicates 10 minutes. The unit must be **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value when only Ustore tables are used. For Astore tables, the setting value depends on the user's database environment and application load. If a large number of tables are updated or deleted in the database environment, you are advised to set this parameter to a short interval to ensure that useless data is deleted in a timely manner and the database space is sufficient. If there are few update or delete operations in the database environment, you can set this parameter to a long interval to reduce the overhead of the automatic cleanup thread.

Risks and impacts of improper settings:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If **autovacuum_naptime** is set to a short interval, the overhead of the autovacuum thread may be too high, affecting the database performance.
- If **autovacuum_naptime** is set to a long interval, useless data may be accumulated, occupying database space and affecting database performance.

autovacuum_vacuum_threshold

Parameter description: Specifies the reference value for triggering VACUUM operations on Astore and Ustore tables. Threshold for triggering VACUUM is calculated as follows: Value of **autovacuum_vacuum_threshold** + Value of **reltuples** (Number of tuples in the table) x **autovacuum_vacuum_scale_factor**. When the total number of deleted and updated records in a table exceeds the threshold, the VACUUM operation is executed on this table.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **50**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings:

- If the value is too small, VACUUM operations are frequently performed, increasing system overhead and deteriorating system performance.
- If the value is too large, too many dead rows exist in the table, affecting the query performance.

autovacuum_analyze_threshold

Parameter description: Specifies the reference value for triggering ANALYZE operations on Astore and Ustore tables. Threshold for triggering ANALYZE is calculated as follows: Value of **autovacuum_analyze_threshold** + Value of **reltuples** (Number of tuples in the table) x **autovacuum_analyze_scale_factor**. When the total number of deleted, inserted, and updated records in a table exceeds the threshold, the ANALYZE operation is executed on this table. If a global secondary index exists in a table, the ANALYZE operation is performed on the global secondary index only when the total number of deleted, inserted, and updated records in the table exceeds the threshold 20 times.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **50**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings:

- If the value is too small, ANALYZE operations are frequently performed, increasing system overhead and deteriorating system performance.
- If the value is too large, the statistics in the table may be outdated. As a result, the query plan is inaccurate and the query performance is affected.

autovacuum_vacuum_scale_factor

Parameter description: Specifies the scale factor for triggering VACUUM operations on Astore and Ustore tables. Threshold for triggering VACUUM is calculated as follows: Value of **autovacuum_vacuum_threshold** + Value of **reltuples** (Number of tuples in the table) x **autovacuum_vacuum_scale_factor**. When the total number of deleted and updated records in a table exceeds the threshold, the VACUUM operation is executed on this table.

Parameter type: floating-point number

Unit: none

Value range: 0.0 to 100.0

Default value: **0.2**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: The value of this parameter depends on the load and usage of a table. If the write load is high, set this parameter to a smaller value so that the table can be cleared more frequently to prevent performance deterioration. If the read load is high, set this parameter to a larger value to avoid unnecessary waste of resources during read.

Risks and impacts of improper settings:

- If the value is too small, the autovacuum thread may clear tables too frequently, causing performance deterioration.
- If the value is too large, the table may not be cleared in a timely manner. As a result, the table size becomes large and occupies too much disk space.

autovacuum_analyze_scale_factor

Parameter description: Specifies the scale factor for triggering ANALYZE operations on Astore and Ustore tables. Threshold for triggering ANALYZE is calculated as follows: Value of **autovacuum_analyze_threshold** + Value of **reltuples** (Number of tuples in the table) x **autovacuum_analyze_scale_factor**. When the total number of deleted, inserted, and updated records in a table exceeds the threshold, the ANALYZE operation is executed on this table.

Parameter type: floating-point number

Unit: none

Value range: 0.0 to 100.0

Default value: **0.1**, that is, 10% of the number of tuples in the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings:

- If the value is too small, ANALYZE operations are frequently performed, increasing system overhead and deteriorating system performance.
- If the value is too large, the frequency of performing ANALYZE operations may be too low and the statistics in the table may be outdated. As a result, the query plan is inaccurate and the query performance is affected.

autovacuum_analyze_window

Parameter description: Specifies one or more time windows from Monday to Sunday. In the specified time window, **autovacuum_analyze_window_scale_factor** is used as the scale factor to trigger the ANALYZE operation on the Astore and Ustore tables.

Parameter type: string.

Unit: none

Value range: an empty string or a string consisting of one or more specified time windows separated by commas (,).

NOTE

- The string of a specified time window is in the format of *key=value*.
- The key of a time window must be **MONDAY_WINDOW**, **TUESDAY_WINDOW**, **WEDNESDAY_WINDOW**, **THURSDAY_WINDOW**, **FRIDAY_WINDOW**, **SATURDAY_WINDOW**, or **SUNDAY_WINDOW**, which is case-insensitive.
- The value of a time window is in the format of *HH:MM-HH:MM*, which is separated by a hyphen (-). The left part is the start time of the window, and the right part is the end time of the window. The value cannot contain spaces. The minimum time is **00:00**, and the maximum time is **23:59**.
- If the end time of a time window is earlier than the start time, the time window crosses days. For example, **"MONDAY_WINDOW=22:00-02:00"** indicates that the time window starts from 22:00 on Monday to 02:00 on Tuesday.
- Time windows of multiple days cannot overlap. For example, if **"MONDAY_WINDOW=22:00-05:00,TUESDAY_WINDOW=03:00-09:00"** cannot be set, you can set it to **"MONDAY_WINDOW=22:00-02:59,TUESDAY_WINDOW=03:00-09:00"**.
- This parameter is not controlled by the parameter **autovacuum_mode**. As long as this parameter is set successfully, statistics are collected in the specified time window.
- Only one window can be configured for a day. If multiple windows are configured for a day, only the last configured window takes effect.
- The time window is determined based on the local system time. When changing the system time or time zone, check whether the time window specified by this parameter needs to be modified.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value or adjust the value based on service requirements.

Risks and impacts of improper settings: The interval of a single time window must be greater than the value of **autovacuum_naptime**. Otherwise, the time window may not take effect.

autovacuum_analyze_window_scale_factor

Parameter description: Specifies the scale factor for triggering ANALYZE operations on Astore and Ustore tables in the time window. Threshold for triggering ANALYZE is calculated as follows: Value of **autovacuum_analyze_threshold** + Value of **reltuples** (Number of tuples in the table) x Value of **autovacuum_analyze_window_scale_factor**. When the total number of deleted, inserted, and updated records in a table exceeds the threshold, the ANALYZE operation is executed on this table.

Parameter type: floating-point number

Unit: none

Value range: 0.0 to 100.0

Default value: **0.1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings:

- If the value is too small, ANALYZE operations are frequently performed in the time window, increasing system overhead and deteriorating system performance.
- If the value is too large, the frequency of performing ANALYZE operations in the time window may be too low and the statistics in the table may be outdated. As a result, the query plan is inaccurate and the query performance is affected.

autovacuum_freeze_max_age

Parameter description: Specifies the number of transactions after which the VACUUM operation is forcibly performed for the **pg_class.relfrozenxid64** column in Astore and Ustore tables. The system starts the AUTOVACUUM thread even if automatic cleanup is disabled. After the forcible VACUUM operation is complete, the system deletes old files that can be deleted from the **pg_clog** and **pg_csnlog** subdirectories to prevent the **pg_clog** and **pg_csnlog** directories from expanding infinitely.

Parameter type: integer.

Unit: none

Value range: 100000 to 576460752303423487

Default value: **4000000000**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If a large number of update and deletion operations are frequently performed on the Astore table in the database, you are advised to set this parameter to a small value to recycle garbage in a timely manner. If the Astore table in the database is seldom updated or deleted, you can set this parameter to a large value to reduce the frequency of automatic space recycling.

Risks and impacts of improper settings:

- If the value is too small, automatic space recycling is frequently performed, increasing system overhead and deteriorating system performance.
- If this parameter is set to a large value, automatic space recycling is delayed. As a result, a large amount of invalid data may occupy disk space, affecting database performance.

autovacuum_vacuum_cost_delay

Parameter description: Specifies the value of the cost delay used in the autovacuum operation.

Parameter type: integer.

Unit: ms

Value range: –1 to 100. **-1** indicates that the regular **vacuum_cost_delay** is used.

Default value: **20**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

autovacuum_vacuum_cost_limit

Parameter description: Sets the value of the cost limit used in the autovacuum operation.

Parameter type: integer.

Unit: none

Value range: –1 to 10000. **-1** indicates that the regular **vacuum_cost_limit** is used.

Default value: **-1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

twophase_clean_workers

Parameter description: Specifies the maximum number of concurrent cleanup operations that can be performed by the gs_clean tool.

Parameter type: integer.

Unit: none

Value range: 1 to 10

Default value: **3**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

clog_keep_xids

Parameter description: Specifies the number of transactions retained in a Clog transaction file.

Parameter type: integer.

Unit: none

Value range: 0 to 576460752303423487

- If the value is **0**, the original recycling logic is used and no extra transaction files are retained.
- If the value is not 0 and is within the value range, the new recycling logic is used. (The number of retained transactions is calculated based on the latest transaction ID. The retention mechanism is triggered only when the new parameter value range contains the transaction files to be recycled based on the original recycling logic.)

Default value: **0**

Description: Evaluate the disk space and ensure that sufficient disk space is reserved in the environment. The method for estimating the disk space of each node is as follows:

The number of retained transactions is calculated based on the latest transaction ID. The retention mechanism is triggered only when the new parameter value range contains the transaction files to be recycled based on the original recycling logic.

Space reserved by Clog (bytes): **clog_keep_xids**/4

Space reserved by Clog (hash bucket) (bytes): Number of databases x Number of buckets x (**clog_keep_xids**/4)

For example, if **clog_keep_xids** is set to **4294967295**, the space occupied in common scenarios is 1073741823 bytes (4294967295/4), about 1 GB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. (Calculate the disk space usage based on the preceding formula. If the disk space is sufficient and the hash bucket service does not exist, you are advised to set this parameter to **8589934590**. If the hash bucket service exists, you are advised to set this parameter to **8388607**.)

Risks and impacts of improper settings: If the parameter is set improperly, the disk space usage may be high. All nodes are involved. Exercise caution when evaluating the risk.

csnlog_keep_xids

Parameter description: Specifies the number of transactions retained in a CSNlog transaction file.

Parameter type: integer.

Unit: none

Value range: 0 to 576460752303423487

- If the value is **0**, the original recycling logic is used and no extra transaction files are retained.
- If the value is not 0 and is within the value range, the new recycling logic is used. (The number of retained transactions is calculated based on the latest transaction ID. The retention mechanism is triggered only when the new parameter value range contains the transaction files to be recycled based on the original recycling logic.)

Default value: **0**

Description: Evaluate the disk space and ensure that sufficient disk space is reserved in the environment. The method for estimating the disk space of each node is as follows:

The number of retained transactions is calculated based on the latest transaction ID. The retention mechanism is triggered only when the new parameter value range contains the transaction files to be recycled based on the original recycling logic.

csnlog_keep_xids space estimation:

Space occupied by retained CSNlog files (bytes): **csnlog_keep_xids** x 8

For example, if **csnlog_keep_xids** is set to **4294967295**, 34359738360 bytes (4294967295 x 8) are occupied, which is about 34 GB.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. (Calculate the disk space usage based on the preceding formula. If the disk space is sufficient, you are advised to set this parameter to **252645135**.)

Risks and impacts of improper settings: If the parameter is set improperly, the disk space usage may be high. All nodes are involved. Exercise caution when evaluating the risk.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

defer_csn_cleanup_time

Parameter description: Specifies the local recycling interval.

Parameter type: integer.

Unit: ms

Value range: 0 to 2147483647

Default value: **5000** (5s)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Generally, if the system load is light, you can set **defer_csn_cleanup_time** to a long time. If the system load is heavy, you can set **defer_csn_cleanup_time** to a short time. In this case, the system can maintain good performance even when it is busy.

Risks and impacts of improper settings: The **defer_csn_cleanup_time** parameter is used to specify when to delete expired CSN information. It specifies how long the system retains expired CSN information. If this parameter is set to a small value, the system may frequently clear expired CSN information, which deteriorates system performance. If the value is too large, the system may occupy too much memory. As a result, the system slows down or breaks down.

multixact_keep_mxids

Parameter description: Specifies the number of transactions retained in a multixact transaction file.

Parameter type: integer.

Unit: none

Value range: 0 to 576460752303423487

- If the value is **0**, the original recycling logic is used and no extra transaction files are retained.
- If the value is not 0 and is within the value range, the new recycling logic is used. (The number of retained transactions is calculated based on the latest multixact ID. The retention mechanism is triggered only when the new parameter value range contains the transaction files to be recycled based on the original recycling logic.)

Default value: **0**

Description: Evaluate the disk space and ensure that sufficient disk space is reserved in the environment. The method for estimating the disk space of each node is as follows:

The number of retained transactions is calculated based on the latest transaction ID. The retention mechanism is triggered only when the new parameter value range contains the transaction files to be recycled based on the original recycling logic.

multixact_keep_mxids space estimation:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Space reserved for multixact (bytes): **multixact_keep_mxids** x 8 + **multixact_keep_mxids** x **members** x 8

For example, when **multixact_keep_mxids** is set to **4294967295** and there are three concurrent requests (three members) in the multixact scenario, the total size that resources are occupied is calculated as follows:

(4294967295 x 8 + 4294967295 x 3 x 8) = 137438953440 bytes, about 137 GB.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. (Calculate the disk space usage based on the preceding formula. If the disk space is sufficient, you are advised to set this parameter to **63161283**.)

Risks and impacts of improper settings: If the parameter is set improperly, the disk space usage may be high. All nodes are involved. Exercise caution when evaluating the risk.

6.3.14 Default Settings of Client Connection

6.3.14.1 Statement Behavior

This section describes related default parameters involved in the execution of SQL statements.

search_path

Parameter description: Specifies the order in which schemas are searched when an object is referenced with no schema specified.

Parameter type: string.

Unit: none

Value range: a character string consisting of one or more schema names. Different schema names are separated by commas (,).

NOTE

- When this parameter is set to **'"$user",public'**, shared use of a database (where no users have private schemas, and all share use of public), private per-user schemas, and combinations of them are supported. Other effects can be obtained by modifying the default search path setting, either globally or per-user.
- When this parameter is set to a null character string (''), the system automatically converts it into a pair of double quotation marks ("").
- If the content contains double quotation marks, the system considers them as insecure characters and converts each double quotation mark into a pair of double quotation marks.

Default value: **'"$user",public'**

NOTE

$user indicates the name of the schema with the same name as the current session user. If the schema does not exist, **$user** will be ignored.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- If schemas of temporary tables exist in the current session, the schemas can be listed in the search path by using the alias **pg_temp**, for example, **'pg_temp,public'**. The schema of temporary tables has the highest search priority and is always searched before all the other schemas specified in **pg_catalog** and **search_path**. Therefore, do not explicitly specify **pg_temp** to be searched after other schemas in **search_path**. If **pg_temp** is specified in **search_path**, this setting will not take effect and an error message will be displayed. If the alias **pg_temp** is used, the temporary schema will be searched only for tables, views, and data types, and not for functions or operators.
- The system catalog schema, pg_catalog, has the second highest search priority and is the first to be searched among all the schemas, excluding **pg_temp**, specified in **search_path**. Therefore, do not explicitly specify **pg_catalog** to be searched after other schemas in **search_path**. If **pg_catalog** is specified in **search_path**, this setting will not take effect and an error message will be displayed.
- When an object is created without a specific target schema, the object will be placed in the first valid schema listed in **search_path**. An error is reported if the search path is empty.
- The current effective value of the search path can be examined through the SQL function current_schema(). This is different from examining the value of **search_path** because the current_schema() function displays the first valid schema name in **search_path**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

current_schema

Parameter description: Specifies the current schema.

Parameter type: string.

Unit: none

Value range: a character string consisting of one or more schema names. Different schema names are separated by commas (,).

Default value: **'"$user",public'**

NOTE

$user indicates the name of the schema with the same name as the current session user. If the schema does not exist, **$user** will be ignored.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

default_tablespace

Parameter description: Specifies the default tablespace of the created objects (tables and indexes) when a **CREATE** command does not explicitly specify a tablespace.

NOTE

- The value of this parameter is either the name of a tablespace, or an empty string that indicates the use of the default tablespace of the current database. If a non-default tablespace is specified, users must have CREATE privilege for it. Otherwise, creation attempts will fail.
- This parameter is not used for temporary tables. For them, **temp_tablespaces** is used instead.
- This parameter is not used when users create databases. By default, a new database inherits its tablespace setting from the template database.

Parameter type: string.

Unit: none

Value range: Specifies the name of an existing tablespace or an empty string. An empty string indicates that the default tablespace is used.

Default value: ""

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

temp_tablespaces

Parameter description: Specifies one or more tablespaces to which temporary objects (temporary tables and their indexes) will be created when a CREATE command does not explicitly specify a tablespace. Temporary files for sorting large data sets are created in these tablespaces.

The value of this parameter can be a list of names of tablespaces. When there is more than one name in the list, GaussDB chooses a random tablespace from the list upon the creation of a temporary object each time. However, within a transaction, successively created temporary objects are placed in successive tablespaces in the list. If the element selected from the list is an empty character string, GaussDB will automatically use the default tablespace of the current database instead.

Parameter type: string.

Unit: none

Value range: name of an existing tablespace or an empty string. An empty string indicates that all temporary objects are created only in the default tablespace of the current database. For details, see **default_tablespace**.

Default value: ""

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

default_storage_nodegroup

Parameter description: Specifies the node group where a table is created by default. This parameter takes effect only for ordinary tables.

Parameter type: string.

Unit: none

Value range: an existing Node Group name or **"installation"**.

Default value: **"installation"**

- **"installation"** indicates that tables will be created in the node group created during database installation.
- Other character strings indicate that tables will be created in the node group specified by this parameter.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

check_function_bodies

Parameter description: Specifies whether to enable validation of the function body string during the execution of **CREATE FUNCTION**. Verification is occasionally disabled to avoid problems, such as forward references when you restore function definitions from a dump. After the function is enabled, the syntax of the PL/SQL in the stored procedure is verified, including the data type, statement, and expression. The SQL statements in the stored procedure are not checked in the CREATE phase. Instead, they are checked during running.

NOTE

In O-compatible mode, if **check_function_bodies** is set to **off**, a deadlock occurs when a stored procedure or function with a user-defined type and autonomous transaction is called. Set **check_function_bodies** to **on**. After a function or stored procedure is rebuilt, the function or stored procedure can be called properly.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Validation of the function body string is enabled during the execution of CREATE FUNCTION.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **off**: Validation of the function body string is disabled during the execution of CREATE FUNCTION.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

default_transaction_isolation

Parameter description: Specifies the default isolation level of each transaction.

NOTE

The current version does not support the setting of the default transaction isolation level. The default value is **read committed**. Do not change the value.

Parameter type: enumerated type.

Unit: none

Value range:

- **read committed**: The transaction is read and committed.
- **read uncommitted**: The transaction is read but not committed.
- **repeatable read**: The transaction can be read repeatedly.
- **serializable**: Currently, this isolation level is not supported in GaussDB. It is equivalent to **repeatable read**.

Default value: read committed

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

default_transaction_read_only

Parameter description: Specifies whether each new transaction is in read-only state.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the transaction is in read-only state.
- **off** indicates that the transaction is not in read-only state.

Default value: **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **on**, DML cannot be performed and transactions cannot be written.

default_transaction_deferrable

Parameter description: Specifies the default deferrable status of each new transaction. It currently has no effect on read-only transactions or those running at isolation levels lower than serializable.

GaussDB does not support the serializable isolation level. Therefore, the parameter takes no effect.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: delayed by default.
- **off**: no delay by default.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

session_replication_role

Parameter description: Specifies the behavior of replication-related triggers and rules for the current session.

NOTE

Modifying this parameter will discard all the cached execution plans.

Parameter type: enumerated type.

Unit: none

Value range:

- **origin**: The current session is the data source. Triggers are executed, and foreign keys, constraints, and rules are checked. This value is the default setting for all common sessions and applies to common database operations.
- **replica**: The current session is the data receiver (for example, the secondary database). Only triggers, foreign keys, and rules in ENABLE REPLICA mode are enabled. Triggers, rules, and foreign keys in other modes are disabled.
- **local**: The system detects the role that has logged in to the database when using the function to copy operations and performs related operations. This

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

value is used in specific replication function scenarios and is used less frequently. The behavior of this value is the same as that of **origin**.

Default value: **origin**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

statement_timeout

Parameter description: If the statement execution time (starting from the time the server receives the command) is longer than the duration specified by the parameter, error information is displayed and the statement exits.

Parameter type: integer.

Unit: milliseconds

Value range: 0 to 2147483647

Default value: **0**, indicating that the timeout error reporting function is disabled.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **2000** without a unit, **statement_timeout** indicates 2000 ms. If the value is **2s**, **statement_timeout** indicates 2s. The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

vacuum_freeze_min_age

Parameter description: Specifies whether VACUUM replaces the **xmin** column (minimum time) of a record with **FrozenXID** when scanning a table (in the same transaction).

Parameter type: integer.

Unit: none

Value range: 0 to 576460752303423487

NOTE

By default, regardless of the value configured, VACUUM will limit the effective value within 50% of **autovacuum_freeze_max_age**.

Default value: **2000000000**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this parameter is set to a large value, the freeze time is delayed and the recycling efficiency is reduced.

vacuum_freeze_table_age

Parameter description: Specifies when VACUUM scans the whole table and freezes old tuples. VACUUM performs a full table scan if the difference between the current transaction ID and the value of **pg_class.relfrozenxid64** is greater than the specified time.

Parameter type: integer.

Unit: none

Value range: 0 to 576460752303423487

NOTE

By default, regardless of the value configured, VACUUM will limit the effective value within 95% of **autovacuum_freeze_max_age**. Therefore, a periodic manual VACUUM has a chance to run before an anti-wraparound autovacuum is launched for the table.

Default value: **4000000000**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a large value, tablespace expansion may occur.

bytea_output

Parameter description: Specifies the output format for values of the bytea type.

Parameter type: enumerated type.

Unit: none

Value range:

- **hex** indicates the binary data is converted to the two hexadecimal digits per byte.
- **escape** uses ASCII character sequences to represent binary strings, and converts those binary strings that cannot be represented as ASCII characters into special escape sequences. When **sql_compatibility** is set to **'B'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, the binary data is not escaped. Instead, the original bytes are returned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

When **sql_compatibility** is set to **'B'**, **b_format_version** is set to **'5.7'**, and **bytea_output** is set to **'escape'**, the following rules are used for querying data in a new table where data in hex format is inserted using the escape string E'\\x' based on the value of **b_format_dev_version**:

- When **b_format_dev_version** is set to **'s1'**, if the client-side character set is the same as the server-side character set, no error is reported during the query. If the client-side character set is different from the server-side character set and the encoding of the inserted data is invalid in the server-side character set, an error is reported during the query.
- When **b_format_dev_version** is set to **'s2'** and the encoding of the inserted data is invalid in the client-side character set, an error is reported during the query.

Default value: **hex**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

xmlbinary

Parameter description: Specifies the encoding format of binary values in XML files.

NOTE

Currently, this parameter does not support data of the XML type.

Parameter type: enumerated type.

Unit: none

Value range:

- **base64**: The Base64 encoding format is used.
- **hex**: The hex encoding format is used.

Default value: **base64**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

xmloption

Parameter description: Specifies whether XML data is treated as a document or a normal string when it is implicitly converted or serialized.

NOTE

Currently, this parameter does not support data of the XML type.

Parameter type: enumerated type.

Unit: none

Value range:

- **document** indicates an HTML document.
- **content** indicates a common string.

Default value: **content**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_compile_functions

Parameter description: Specifies the maximum number of compilation results of stored procedures and functions that can be stored on a node, including system stored procedures and user-defined stored procedures and functions. If the compilation result exceeds the value of this parameter, the system automatically triggers memory eviction and deletes unused common stored procedures and functions.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483647.

Default value: 1000

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value to reduce memory usage and improve system performance.

Risks and impacts of improper settings: If the number of compilation results exceeds the value of this parameter and the system cannot find the stored procedures and functions that can be deleted, a large amount of memory is occupied, causing performance deterioration.

gin_pending_list_limit

Parameter description: Specifies the threshold for merging the pending lists of GINs and UGINs when **fastupdate** is enabled for GINs or UGINs. If the pending

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

list grows larger than this maximum size, it is cleaned up by moving the entries in it to the main index data structure in batches. This setting can be overridden for a single index by changing index storage parameters.

Parameter type: integer.

Unit: KB

Value range: 64 to 2147483647

Default value: **4096**, that is, 4 MB.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1024** without a unit, **gin_pending_list_limit** indicates 1024 KB. If the value is **1MB**, **gin_pending_list_limit** indicates 1 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Set this parameter to a proper value based on the update frequency to improve the performance of synchronizing indexes.

Risks and impacts of improper settings:

- If this parameter is set to a value greater than **work_mem**, the performance of synchronizing incremental data and scanning data may deteriorate.
- If this parameter is set to a small value, **pending_list** is frequently merged, affecting DML execution performance.

6.3.14.2 Locale and Formatting

This section describes parameters related to the time format setting.

DateStyle

Parameter description: Specifies the display format for date and time values, as well as the rules for interpreting ambiguous date input values.

This variable contains two independent components: the output format declaration (ISO, Postgres, SQL, or German) and the input/output order of year/month/day (DMY, MDY, YMD, Euro, European, US, NonEuro, NonEuropean, or Default). The two components can be set separately or together. The keywords Euro and European are synonyms for DMY. The keywords US, NonEuro, and NonEuropean are synonyms for MDY.

Parameter type: string.

Unit: none

Value range: combination of the output format declaration and the input/output order of year/month/day. (The two components can be set separately.)

- Output format declaration: ISO, Postgres, SQL, or German.
- Input/Output order of year/month/day: DMY (Euro, European), MDY (US, NonEuro, NonEuropean), or YMD

Default value: **"ISO, MDY"**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

gs_initdb will initialize this parameter so that its value is the same as that of **lc_time**.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. If this parameter is modified by running the **gs_guc reload** command and the connection of a session on the current node is not from the client but from another node in the cluster to which the node belongs, this parameter does not take effect immediately on the session after the **gs_guc reload** command is executed. The setting takes effect only after the connection node is disconnected and then reconnected.

Setting suggestion: The ISO format is recommended.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

IntervalStyle

Parameter description: Specifies the display format for interval values.

Parameter type: enumerated type.

Unit: none

Value range:

- **sql_standard**: Output matching SQL standards will be generated.
- **postgres**: Output matching PostgreSQL 8.4 will be generated. It is valid when the **DateStyle** parameter is set to **ISO**.
- **postgres_verbose**: Output matching PostgreSQL 8.4 will be generated. It is valid when the **DateStyle** parameter is set to **non_ISO**.
- **iso_8601**: Output matching the time interval "format with designators" defined in ISO 8601 will be generated.
- **oracle**: The numtodsinterval function generates output matching the Oracle format. For details, see **numtodsinterval(num, interval_unit)**.

NOTE

The **IntervalStyle** parameter also affects the interpretation of ambiguous interval input.

Default value: **postgres**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. If this parameter is modified by running the **gs_guc reload** command and the connection of a session on the current node is not from the client but from another node in the cluster to which the node belongs, this parameter does not take effect immediately on the session after the **gs_guc reload** command is executed. The setting takes effect only after the connection node is disconnected and then reconnected.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

TimeZone

Parameter description: Specifies the time zone for displaying and interpreting timestamps.

Parameter type: string.

Unit: none

Value range: You can query the PG_TIMEZONE_NAMES view to check the value range. For details, see **PG_TIMEZONE_NAMES**.

Default value: **"PRC"**

NOTE

gs_initdb will set a time zone value that is consistent with the system environment.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. If this parameter is modified by running the **gs_guc reload** command and the connection of a session on the current node is not from the client but from another node in the cluster to which the node belongs, this parameter does not take effect immediately on the session after the **gs_guc reload** command is executed. The setting takes effect only after the connection node is disconnected and then reconnected.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

timezone_abbreviations

Parameter description: Specifies the time zone abbreviations that will be accepted by the server.

Parameter type: string.

Unit: none

Value range: **"India"**, **"Australia"**, and **"Default"**.

Default value: **"Default"**

NOTE

- **"Default"** indicates an abbreviation of the universal time zone, which is applicable to most cases. There are also other values, such as **"Australia"** and **"India"** that can be defined for a particular time zone installation.
- You are advised not to modify the corresponding value file. After the time zone abbreviation file is rewritten or replaced, the modification cannot be updated in real time. In some scenarios, you need to restart the database cluster for the modification to take effect.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

extra_float_digits

Parameter description: Adjusts the number of digits displayed for floating-point values, including float4, float8, and geometric data types. The parameter value is added to the standard number of digits (FLT_DIG or DBL_DIG, according to the actual situation).

Parameter type: integer.

Unit: none

Value range: –15 to 3

NOTE

- If the value is greater than 0, some key data bits are included, but the included data bits may not be valid. As a result, the output digits may not be completely valid. This function is applicable to scenarios where floating-point data needs to be accurately restored during dumping.
- If the value is less than or equal to 0, unnecessary data bits are eliminated to ensure that the output digits are valid.
- In M-compatible databases, **extra_float_digits** affects only the number of data bits displayed for the float4 data type without scale. The number of data bits displayed for the float8 data type and the float4 data type with scale is not affected by the GUC parameter.
- In the scenario where JDBC and ODBC are used to establish a connection, the default value of **extra_float_digits** is **0**.

Default value: **3**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

float_shortest_precision

Parameter description: Specifies whether to enable the minimum precision method to replace sprintf() to serialize floating point numbers. The minimum precision method is used to improve performance of a floating-point number serialization system function, that is, shorten execution time of a single thread of two system functions float4out() and float8out().

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The minimum precision method is enabled.
- **off**: The minimum precision method is disabled.

Default value: **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised to set this parameter for new sites or sites that have performance requirements for floating-point serialization.

Risks and impacts of improper settings: After this parameter is enabled, the precision of valid float digits varies according to the value of **extra_float_digits**. The value range of **extra_float_digits** is –15 to 3, the number of valid digits required for each value is as follows.

Value of extra_float_digits	Number of Required Valid Digits	
	Float4	Float8
-15	1	1
-14	1	1
-13	1	2
-12	1	3
-11	1	4
-10	1	5
-9	1	6
-8	1	7
-7	1	8
-6	1	9
-5	1	10
-4	2	11
-3	3	12
-2	4	13
-1	5	14
0	6	15
1	7	16
2	8	17
3	9	18

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- For the original logic that calls snprintf_s(), the number of required valid digits does not mean that all output digits are valid.
  1. When the value of **extra_float_digits** is greater than 0, the output digits are not guaranteed to be completely valid.
  2. When the value of **extra_float_digits** is less than or equal to 0, the output digits are completely valid.
- For the new logic that uses the minimum precision method, only valid digits are output.
  1. When the value of **extra_float_digits** is greater than 0, it is not guaranteed that the number of output digits reaches the number of required valid digits, but all output digits are valid.
  2. When the value of **extra_float_digits** is less than or equal to 0, the output digits are completely valid.

client_encoding

Parameter description: Specifies the client-side encoding (character set).

Parameter type: string.

Unit: none

Value range: all PG-compatible encoding types. **UTF8** indicates that the database encoding is used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- You can run the **locale -a** command to check the system-supported locales and the corresponding encodings, and select one as required. The sorting rules may vary according to OSs or versions of the same OS. When application services are logically migrated between these OSs, differences and changes in OS sorting rules may cause differences in database functions such as indexes, partitions, and sorting operators. For example, data result sets returned by a same range query statement are different, or query results returned by a same sorting query statement are different. The application service needs to check whether the service data contains related characters based on the locale differences released by the OS.
- By default, **gs_initdb** will initialize the setting of this parameter based on the current system environment. You can also run the **locale** command to check the current configuration environment.
- To use consistent encoding for communication within the cluster, you are advised to retain the default value of **client_encoding**. Modification to this parameter in the **gaussdb.conf** file (by using the gs_guc tool, for example) does not take effect.
- **client_encoding** can be set to **GB18030_2022**. When **client_encoding** is set to **gb18030_2022**, ensure that the GB18030 character set supported by the client OS has been upgraded to version 2022.
- If the encoding format of the database character set is UTF8, the encoding format of the region supported by the current system is GB18030, the GB18030 character set supported by the client OS has been upgraded to version 2022, and the database contains historical data stored before the GB18030 character set is upgraded, the following situations may occur:
  1. When **client_encoding** is set to **gb18030**, the character encoding of historical data in the UTF8 database returned to the client is the same as that before the client character set is upgraded. However, the font of 38 characters whose mapping changes during the character set upgrade is the same as that in version 2022.
  2. When **client_encoding** is set to **gb18030_2022**, the character encoding of historical data in the UTF8 database returned to the client are the same as that after the upgrade of the client character set and the font of characters are the same as that before the upgrade. This is because the versions of the GB18030 character set are not completely compatible, which may cause data inconsistency. Therefore, if the GB18030 character set needs to be upgraded on the client and historical data exists, you need to convert the historical data before the upgrade.
- **client_encoding** can be set to **ZHS16GBK**. The following table lists the setting methods of **server_encoding** and **client_encoding** corresponding to all ways that ZHS16GBK, GB18030, and GB18030_2022 character sets can be transformed.

server_encoding	client_encoding	locale	Setting Method
zhs16gbk	utf8	utf8	The database automatically obtains the value of **locale**.

server_encoding	client_encoding	locale	Setting Method
utf8	zhs16gbk	gbk	• Users use gsql to manually set **client_encoding** to **zhs16gbk**. • When JDBC is used for connection, the URL parameter is used to set **characterEncoding** to **zhs16gbk**.
zhs16gbk	zhs16gbk	gbk	• When gsql is used for connection, the database automatically obtains the value of **locale**. After internal processing, **client_encoding** is automatically set to **zhs16gbk**. • When JDBC is used for connection, the URL parameter is used to set **characterEncoding** to **zhs16gbk**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

server_encoding	client_encoding	locale	Setting Method
zhs16gbk	gb18030	gb18030	• When gsql is used for connection, the database automatically obtains the value of **locale**. • When JDBC is used for connection, the URL parameter is used to set **characterEncoding** to **gb18030**.
gb18030	zhs16gbk	gbk	• gsql does not support this setting. • When JDBC is used for connection, the URL parameter is used to set **characterEncoding** to **zhs16gbk**.
zhs16gbk	gb18030-2022	gb18030	• When gsql is used for connection, users manually set **client_encoding** to **gb18030_2022**. • When JDBC is used for connection, the URL parameter is used to set **characterEncoding** to **gb18030_2022**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

server_encoding	client_encoding	locale	Setting Method
gb18030-2022	zhs16gbk	gbk	• gsql does not support this setting. • When JDBC is used for connection, the URL parameter is used to set **characterEncoding** to **zhs16gbk**.
gb18030	utf8	utf8	The database automatically obtains the value of **locale**.
utf8	gb18030	gb18030	• The database automatically obtains the value of **locale**. • When JDBC is used for connection, the URL parameter is used to set **characterEncoding** to **gb18030**.
gb18030_2022	utf8	utf8	The database automatically obtains the value of **locale**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

server_encoding	client_encoding	locale	Setting Method
utf8	gb18030_2022	gb18030	• When gsql is used for connection, users manually set **client_encoding** to **gb18030_2022**. • When JDBC is used for connection, the URL parameter is used to set **characterEncoding** to **gb18030_2022**.
gb18030	gb18030	gb18030	• The database automatically obtains the value of **locale**. • When JDBC is used for connection, the URL parameter is used to set **characterEncoding** to **gb18030**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

server_encoding	client_encoding	locale	Setting Method
gb18030_2022	gb18030_2022	gb18030	• When gsql is used for connection, the database automatically obtains the value of **locale**. After internal processing, **client_encoding** is automatically set to **gb18030_2022**. • When JDBC is used for connection, the URL parameter is used to set **characterEncoding** to **gb18030_2022**.

Default value: **"UTF8"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on frontend services. Try to keep the encoding consistent on the client and server to improve efficiency. You are advised to set this parameter to **"SQL_ASCII"** or **"UTF8"**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

lc_messages

Parameter description: Specifies the language in which messages are displayed.

Parameter type: string.

Unit: none

Value range: The value range is related to the system. You can run the **locale -a** command to view the value range.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- You can run the **locale -a** command to check the system-supported locales and the corresponding encodings, and select one as required.
- On some systems, this value range does not exist. Setting this variable will still work, but there will be no effect. In addition, translated messages for the desired language may not exist. In this case, you can still see the English messages.
- By default, **gs_initdb** will initialize the setting of this parameter based on the current system environment. You can also run the **locale** command to check the current configuration environment.

Default value: **"C"**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

lc_monetary

Parameter description: Specifies the display format of monetary values. It affects the output of functions such as to_char(). Acceptable values are system-related.

Parameter type: string.

Unit: none

Value range: The value range is related to the system. You can run the **locale -a** command to view the value range.

NOTE

- You can run the **locale -a** command to check the system-supported locales and the corresponding encodings, and select one as required.
- By default, **gs_initdb** will initialize the setting of this parameter based on the current system environment. You can also run the **locale** command to check the current configuration environment.
- If this parameter is modified by running the **gs_guc reload** command and the connection of a session on the current node is not from the client but from another node in the cluster to which the node belongs, this parameter does not take effect immediately on the session after the **gs_guc reload** command is executed. The setting takes effect only after the connection node is disconnected and then reconnected.

Default value: **"C"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

lc_numeric

Parameter description: Specifies the display format of numbers. It affects the output of functions such as **to_char**.

Parameter type: string.

Unit: none

Value range: The value range is related to the system. You can run the **locale -a** command to view the value range.

NOTE

- You can run the **locale -a** command to check the system-supported locales and the corresponding encodings, and select one as required.
- By default, **gs_initdb** will initialize the setting of this parameter based on the current system environment. You can also run the **locale** command to check the current configuration environment.
- If this parameter is modified by running the **gs_guc reload** command and the connection of a session on the current node is not from the client but from another node in the cluster to which the node belongs, this parameter does not take effect immediately on the session after the **gs_guc reload** command is executed. The setting takes effect only after the connection node is disconnected and then reconnected.

Default value: **"C"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

lc_time

Parameter description: Specifies the display format of time and locale. It affects the output of functions such as to_char().

Parameter type: string.

Unit: none

Value range: The value range is related to the system. You can run the **locale -a** command to view the value range.

NOTE

- You can run the **locale -a** command to check the system-supported locales and the corresponding encodings, and select one as required.
- By default, **gs_initdb** will initialize the setting of this parameter based on the current system environment. You can also run the **locale** command to check the current configuration environment.
- If this parameter is modified by running the **gs_guc reload** command and the connection of a session on the current node is not from the client but from another node in the cluster to which the node belongs, this parameter does not take effect immediately on the session after the **gs_guc reload** command is executed. The setting takes effect only after the connection node is disconnected and then reconnected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **"C"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

lc_time_names

Parameter description: Specifies the language used to display names and abbreviations of dates and months. The output of the DATE_FORMAT(), DAYNAME(), and MONTHNAME() functions is affected. The performance of STR_TO_DATE() and GET_FORMAT() is not affected.

Parameter type: string.

Unit: none

Value range: IETF language tags in simple format, such as **en_US** or **zh_CN**.

Default value: **"en_US"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Switch the language based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

default_week_format

Parameter description: Specifies the default **mode** value of the week() function. The following table lists the value range and description.

default_week_format	First Day of a Week	Range	First Week Definition
0	Sunday	0 to 53	The first week that contains a Sunday in this year.
1	Monday	0 to 53	The first week that contains four or more days in this year.
2	Sunday	1 to 53	The first week that contains a Sunday in this year.
3	Monday	1 to 53	The first week that contains four or more days in this year.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

default_week_format	First Day of a Week	Range	First Week Definition
4	Sunday	0 to 53	The first week that contains four or more days in this year.
5	Monday	0 to 53	The first week that contains a Monday in this year.
6	Sunday	1 to 53	The first week that contains four or more days in this year.
7	Monday	1 to 53	The first week that contains a Monday in this year.

NOTE

For week 1 of a year, the number 1 indicates a week number. The week number of the first week of a year may be 0, depending on the value of **week_format**, as shown in the third column.

Parameter type: integer.

Unit: none

Value range: –2147483648 to 2147483647

- If the value is less than 0, it is forcibly changed to **0**.
- If the value is greater than 7, it is forcibly changed to **7**.

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Select a mode based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.14.3 Other Default Parameters

This section describes the default database loading parameters.

dynamic_library_path

Parameter description: Specifies the path that the system will search for a shared database file that is dynamically loadable. When a dynamically loadable module needs to be opened and the file name specified in the **CREATE FUNCTION** or **LOAD** command does not contain a directory, the system will search this path for the required file. Only the SYSADMIN user can access this parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: string.

Unit: none

Value range: a list of absolute paths separated by colons (;) in Windows.

When the name of a path starts with the special variable *$libdir*, the variable will be replaced with the directory in which the module provided by GaussDB is installed. For example:

```
dynamic_library_path = '/usr/local/lib/gaussdb:/opt/testgs/lib:$libdir'
```

NOTE

If the value of this parameter is set to an empty character string, the automatic path search is turned off.

Default value: **"$libdir"**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The dynamic library search paths specified by **local_preload_libraries** and **shared_preload_libraries** also depend on this parameter. If this parameter is left blank or set to an invalid path, the dynamic libraries fail to be loaded, which will result in abnormal database services.

gin_fuzzy_search_limit

Parameter description: Specifies the upper soft limit of the size of the set returned by the UGIN or GIN. If this parameter is not set to the default value, an error is reported during UGIN scan.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_astore_gin

Parameter description: Specifies whether to enable the GIN function. If this function is enabled, Astore supports GIN creation. If this function is disabled, the GIN function is unavailable.

Parameter type: Boolean.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: Astore supports the GIN function.
- **off**: Astore does not support the GIN function.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value and adjust the value based on service requirements.

Risks and impacts of improper settings: none

enable_astore_gist

Parameter description: Specifies whether to enable the GiST function. If this function is enabled, Astore supports GiST creation. If this function is disabled, the GiST function is unavailable.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Astore supports the GiST function.
- **off**: Astore does not support the GiST function.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value and adjust the value based on service requirements.

Risks and impacts of improper settings: none

local_preload_libraries

Parameter description: Specifies one or more shared libraries that are to be preloaded at connection start. If multiple shared libraries are separated with commas (,). All library names (except the names in double quotation marks) are converted to lower case.

- Any user can change this option. Therefore, library files that can be loaded are restricted to those saved in the **plugins** subdirectory of the standard library installation directory. It is the database administrator's responsibility to ensure that libraries in this directory are all safe. Entries in **local_preload_libraries** can specify the library directory explicitly, for example, **$libdir/plugins/mylib**, or just specify the library name, for example, **mylib**. (**mylib** is equivalent to **$libdir/plugins/mylib**.)
- Unlike **shared_preload_libraries**, there are no differences in performance between loading a module at session start or doing this during the session. The intent of this feature is to allow debugging or performance-measurement libraries to be loaded into specific sessions without an explicit LOAD command. For example, debugging can be enabled under a given username by setting this parameter to **ALTER USER SET**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Every GaussDB-supported library has a "magic block" that is checked to guarantee compatibility. For this reason, non-GaussDB-supported libraries cannot be loaded in this way.

Parameter type: string.

Unit: none

Value range: a list of shared libraries separated by commas (,), or an empty string.

Default value: ""

Setting method: This is a BACKEND parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the specified library is not found, the connection fails. If the configuration is incorrect (for example, the length exceeds the limit or the directory is invalid), the process cannot be started properly.

enable_driver_param_sync

Parameter description: Specifies whether the GUC parameters specified when the driver creates a database connection are synchronized from CNs to DNs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **off** for an upgraded database instance and **on** for a newly installed database instance.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. For an upgraded database, you are advised to manually set this parameter to **on** after evaluating the impact on services.

Risks and impacts of improper settings: If this parameter is set to **off**, after the **RESET ALL** command is executed, the parameter values set on CNs when the driver creates a connection are restored to the values set during connection, and the parameter values on DNs are restored to the default values. In this case, the parameter values on CNs and DNs are different. Therefore, you are advised to manually set this parameter to **on** for an upgraded database after evaluating the impact on services to avoid parameter inconsistency.

6.3.15 Lock Management

In GaussDB, a deadlock may occur when concurrently executed transactions compete for resources. This section describes parameters used for managing transaction locks.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

deadlock_timeout

Parameter description: Specifies the deadlock timeout interval. When the applied lock exceeds the preset value, the system will check whether a deadlock occurs. This parameter takes effect only for common locks.

Parameter type: integer.

Unit: ms

Value range: 1 to 2147483647

Default value: **1000** (that is, 1s)

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- The check for deadlock is relatively expensive. Therefore, the server does not check it when waiting for a lock every time. Deadlocks seldom occur when the system is running. Therefore, the system just needs to wait on the lock for a while before checking for deadlocks. Increasing the value of **deadlock_timeout** reduces the time wasted in deadlock check, but slows down reporting of real deadlock errors. On a heavily loaded server, you may need to set **deadlock_timeout** to a larger value. It is recommended that this value exceed the transaction time to avoid deadlock check before locks are released.
- When **log_lock_waits** is set to **on**, **deadlock_timeout** determines the wait time threshold beyond which lock wait information is written to the logs during query execution. To study the lock delay, you can set **deadlock_timeout** to a value smaller than the normal value.

Risks and impacts of improper settings: If the value is too large, the speed of reporting deadlock errors is slowed down. If the value is too small, deadlock detection starts before the lock is released.

NOTE

Only partial deadlocks caused by a single node can be detected. Distributed global deadlocks involving multi-node collaboration cannot be detected.

lockwait_timeout

Parameter description: Specifies the maximum wait time for a single lock in the application process of all lock types except **update_lockwait_timeout** and **ddl_lockwait_timeout**. If the waiting time of the preceding locks exceeds the value of this parameter, the system reports an error. This parameter takes effect only for common locks.

Parameter type: integer.

Unit: ms

Value range: 0 to 2147483647

Default value: **1200000** (20 minutes)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **900** without a unit, **lockwait_timeout** indicates 900 ms. If the value is **20min**, **lockwait_timeout** indicates 20 minutes. The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value. Set this parameter based on the lock conflict timeout interval in the service scenario. If the value is less than 1s and not 0, the preceding locks will wait for at least 1s before reporting timeout. To make all lock types wait indefinitely, set this parameter to **0**. (**update_lockwait_timeout** and **ddl_lockwait_timeout** do not specify whether to wait indefinitely.)

Risks and impacts of improper settings: If a lock conflict occurs when the preceding locks are requested, a large value of this parameter will prolong the blocking time, and a small value will increase the number of timeout errors.

update_lockwait_timeout

Parameter description: Specifies the maximum waiting time for applying for a transaction ID lock or a tuple lock when concurrent updates are allowed in a row. If the time spent in waiting for a lock exceeds the specified time, an error is reported. This parameter takes effect only for common locks.

Parameter type: integer.

Unit: ms

Value range: 0 to 2147483647

Default value: **120000** (2 minutes)

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **900** without a unit, **update_lockwait_timeout** indicates 900 ms. If the value is **2min**, **update_lockwait_timeout** indicates 2 minutes. The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value. Set the value based on the workload of transactions that concurrently update the same row. In most TP workloads, transactions complete in less than 2 minutes, so the wait time for concurrent update transactions to commit is usually short. In this case, the default value of 2 minutes rarely causes lock timeouts. If many concurrent transactions update the same row and execution time exceeds 2 minutes, you can increase this value to reduce false lock timeout errors. If the value is less than 1s, the preceding locks will wait for at least 1s before reporting timeout.

Risks and impacts of improper settings: During concurrent data update in the same row, if a lock conflict occurs when the transaction ID lock or tuple lock is applied for, the blocking time is prolonged if the value is too large; more transaction ID locks or tuple locks may time out if the value is too small.

ddl_lockwait_timeout

Parameter description: Specifies the maximum waiting time for a single lock when some DDL services apply for advanced object locks (level-5 to level-8 locks) on table objects, index objects, and stored procedures/packages/functions. If the time spent in waiting for a lock exceeds the specified time, an error is reported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: ms

Value range: 0 to 2147483647

Default value: **120000** (2 minutes)

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **900** without a unit, **ddl_lockwait_timeout** indicates 900 ms. If the value is **2min**, **ddl_lockwait_timeout** indicates 2 minutes. The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value. Set this parameter to the expected maximum waiting time based on the DDL service requirements on the timeout interval of the preceding advanced locks. If the value is less than 1s, the preceding locks will wait for at least 1s before reporting timeout.

Risks and impacts of improper settings: If a lock conflict occurs when the preceding advanced locks are requested for DDL services, a large value of this parameter will prolong the blocking time, and a small value will increase the number of advanced lock timeout errors.

max_locks_per_transaction

Parameter description: Specifies the average number of object locks that each transaction can hold when the number of database connections reaches the maximum.

NOTE

This parameter does not strictly limit the maximum number of object locks that can be held by each transaction. If the number of connections does not reach the maximum value (**max_connections**), the average number of object locks that can be held by each transaction increases.

Parameter type: integer.

Unit: none

Value range: 10 to 2147483647

Default value: **256**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- The maximum number of hash tables that can be locked by a shared lock at any time is calculated based on an assumption: **max_locks_per_transaction** × (**max_connections** + **max_prepared_transactions**) and **max_locks_per_transaction** ≥ Number of concurrent service transactions × Number of object locks added by each service transaction/(**max_connections** + **max_prepared_transactions**). In this case, an upper limit is determined for this parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Within the specified range, objects can be locked simultaneously at any time. You may need to increase the value of this parameter if many different tables are modified in a single transaction.
- Increasing the value of this parameter may cause GaussDB to request more System V-shared memory than the OS's default configuration allows, leading to database startup failure. You are advised to set this parameter to a value less than or equal to **max_process_memory** x 10%/(**max_connections** + **job_queue_processes** + **autovacuum_max_workers** + **max_inner_tool_connections** + **max_concurrent_autonomous_transactions** + 1280 + **max_prepared_transactions** x **TWOPHASE_PART**)/(512 bytes). **1280** is the estimated number of reserved auxiliary threads and other threads. The unit of **max_process_memory** must be converted into bytes.
- When running a standby node, you must set this parameter to a value that is no less than that on the primary node. Otherwise, queries will not be allowed on the standby node.
- If this parameter is set to a smaller value and the standby node is faulty before the restart, the standby node fails to be restored. You are advised to set the parameters to the original values and restart the system to restore the environment.
- In specification degradation and DR scenarios, you are advised not to perform DDL operations frequently when modifying this parameter. Otherwise, a memory error will be reported.

Risks and impacts of improper settings: If the value is too large, the database may fail to be started.

max_pred_locks_per_transaction

Parameter description: Specifies the average number of predicate locks that each transaction can hold when the number of database connections reaches the maximum.

NOTE

This parameter does not strictly limit the maximum number of predicate locks that can be held by each transaction. If the number of connections does not reach the maximum value (**max_connections**), the average number of predicate locks that can be held by each transaction increases.

Parameter type: integer.

Unit: none

Value range: 10 to 2147483647

Default value: **64**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- The size of the shared predicate lock table is calculated under the condition that a maximum of *N* independent objects need to be locked at any time. *N* = **max_pred_locks_per_transaction** x (**max_connections** + **max_prepared_transactions**). Within the specified range, objects can be

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

locked simultaneously at any time. You may need to increase this value if many different tables are modified in a single transaction. This parameter can only be set at server start.

- Increasing the value of this parameter may cause GaussDB to request more System V-shared memory than the OS's default configuration allows.

Risks and impacts of improper settings: If the value is too large, the database may fail to be started.

gs_clean_timeout

Parameter description: Transactions in GaussDB are committed in two phases. An unfinished two-phase transaction may hold a table-level lock, keeping tables from being locked by other connections. In this case, the database needs to call the gs_clean tool to clean unfinished two-phase transactions in cluster. **gs_clean_timeout** is used to specify the interval for the CN to call the gs_clean tool. gs_clean can periodically clear residual temporary tables.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483

Default value: **60**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1** without a unit, **gs_clean_timeout** indicates 1s. If the value is **1min**, **gs_clean_timeout** indicates 1 minute. The unit must be **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a large value, the interval for clearing two-phase transactions and residual temporary tables is prolonged. As a result, other connections may fail to be locked before the clearing.

partition_lock_upgrade_timeout

Parameter description: Specifies the timeout for attempts to upgrade an exclusive lock (read allowed) to an access exclusive lock (read/write blocked) on a partitioned table during the execution of some query statements. If there are concurrent read transactions running, the lock upgrade will need to wait. **partition_lock_upgrade_timeout** sets the waiting timeout for lock upgrade attempts.

Example: When you perform **MERGE PARTITION** and **CLUSTER PARTITION** on a partitioned table, temporary tables are used for data rearrangement and file exchange. To concurrently perform as many operations as possible on the partitions, exclusive locks are acquired for the partitions during data rearrangement and access exclusive locks are acquired during file exchange. The locking mode is attempt locking. If the locking is successful, a response is returned immediately. If the locking fails, the system waits for 50 ms and attempts to lock the next time. The locking timeout interval is specified by the session-level parameter **partition_lock_upgrade_timeout**.

Parameter type: integer.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: seconds

Value range: –1 to 3000. **-1** indicates infinite waiting. That is, the system continuously attempts to upgrade the lock until the upgrade is successful.

Default value: **1800**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **-1**, services may be blocked when the lock cannot be upgraded for a long time.

fault_mon_timeout

Parameter description: Specifies the period for detecting lightweight deadlocks.

Parameter type: integer.

Unit: minute

Value range: 0 to 1440

Default value: **5**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a large value, deadlocks cannot be detected in a timely manner, which slows down the speed of reporting deadlock errors.

enable_online_ddl_waitlock

Parameter description: Specifies whether to block DDL operations to wait for the release of cluster locks, such as **pg_advisory_lock** and **pgxc_lock_for_backup**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is mainly used in online OM operations and you are advised not to modify the settings.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

xloginsert_locks

Parameter description: Specifies the number of locks on concurrent write-ahead logging. This parameter is used to improve the efficiency of writing write-ahead logs.

Parameter type: integer.

Unit: none

Value range: 1 to 1000. If the CPU uses the NUMA architecture, the value must be an integer multiple of the number of NUMA nodes.

Default value: 16

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: When the concurrency pressure of the Arm CPU architecture is high, performance jitter may occur due to atomic operation competition. You can decrease the value of this parameter for stable running performance.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

num_internal_lock_partitions

Parameter description: Specifies the number of internal lightweight lock partitions. It is mainly used for performance optimization in various scenarios.

NOTE

- The parameter content is organized in the KV format of keywords and numbers. Different types of locks are separated by commas (,).
- The sequence does not affect the setting result. For example, **"CLOG_PART=256,CSNLOG_PART=512"** is equivalent to **"CSNLOG_PART=512,CLOG_PART=256"**.
- If you set the same keyword repeatedly, the last setting takes effect. For example, if the setting is **"CLOG_PART=256,CLOG_PART=2"**, the value of **CLOG_PART** is **2**.

Parameter type: string.

Unit: none

Value range:

- **CLOG_PART**: number of Clog file controllers. The value ranges from 1 to 256.
- **CSNLOG_PART**: number of CSNlog file controllers. The value ranges from 1 to 512.
- **LOG2_LOCKTABLE_PART**: the base-2 logarithm of the number of regular lock table partitions. The minimum value is **4**, that is, the number of lock partitions is 16. The maximum value is **12**, that is, the number of lock partitions is 4096.
- **TWOPHASE_PART**: number of partitions of the two-phase transaction lock. The value ranges from 1 to 64.
- **FASTPATH_PART**: maximum number of locks that each thread can obtain without using the main lock table. The value ranges from 20 to 10000.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **FASTPATH_PART_PER_SHARD**: size of a single FASTPATH partition when each thread applies for a lock through FASTPATH. The value ranges from 20 to 10000.
- **FASTPATH_PROCEDURE_PART**: maximum number of stored procedure object locks that can be applied for by each thread without using the main lock table. The value ranges from 0 to 10000.

Default value:
"CLOG_PART=256,CSNLOG_PART=512,LOG2_LOCKTABLE_PART=4,TWOPHASE_PART=1,FASTPATH_PART=20,FASTPATH_PART_PER_SHARD=20,FASTPATH_PROCEDURE_PART=100"

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion:

- **CLOG_PART**: Increasing the value of this parameter improves the Clog writing efficiency and transaction commit performance, but increases the memory usage. Decreasing the value of this parameter reduces the memory usage, but may increase the conflict of writing Clogs and affect the performance.
- **CSNLOG_PART**: Increasing the value of this parameter improves the CSNlog writing efficiency and transaction commit performance, but increases the memory usage. Decreasing the value of this parameter reduces the memory usage, but may increase the conflict of writing CSNlogs and affect the performance.
- **LOG2_LOCKTABLE_PART**: Increasing the value can improve the degree of parallelism of obtaining locks in the normal process, but may increase the memory usage. When wait events occur in **LockMgrLock**, you can increase the value to improve the performance.
- **TWOPHASE_PART**: Increasing the value of this parameter increases the number of concurrent two-phase transaction commits but increases the memory usage. If an error indicating insufficient slots is reported for a two-phase transaction (the "Maximum number of prepared transactions reached, please increase max_parepared_transactions (currently max_parepared_transactions) or wait gs_clean to clean reserved prepared transactions." log is generated on the CN or DN), you are advised to increase the value of this parameter to avoid severe transaction lock conflicts.
- **FASTPATH_PART**: When a partitioned table is read, updated, inserted, or deleted and a wait event is **LockMgrLock**, you can increase the value of this parameter to prevent **LockMgrLock** from being obtained, improving performance. It is recommended that the value be greater than or equal to that calculated using the following formula: Number of partitions x (1 + Number of local indexes) + Number of global indexes + 10. Increasing the value will increase the memory usage. The incremental FASTPATH memory (unit: bytes) is calculated as follows: (Incremental FASTPATH/20 x 8 + Incremental FASTPATH x 12) x Thread pool size.
- **FASTPATH_PART_PER_SHARD**: size of a FASTPATH partition. If the value is large, it takes a long time to traverse FASTPATH partitions, which may deteriorate the performance. You are advised to use the default value.
- **FASTPATH_PROCEDURE_PART**: If the wait event of a stored procedure is **LockMgrLock**, you can increase the value of this parameter to prevent **LockMgrLock** from being obtained, improving performance. It is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

recommended that the value be greater than the maximum depth of nested stored procedure calls. Increasing the value will increase the memory usage. The incremental memory (unit: bytes) is calculated as follows: (Incremental value/20 x 8 + Incremental value x 12) x Thread pool size.

Risks and impacts of improper settings: The database performance may be affected.

enable_wait_exclusive_lock

Parameter description: Specifies whether to enable the hang detection and cure function for the exclusive lock of ProcArrayLock.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

barrier_lock_timeout

Parameter description: Specifies the timeout interval for holding a barrier lock.

Parameter type: integer.

Unit: seconds

Value range: 0 to 3600

Default value: **30**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

track_activity_history_query_number

Parameter description: Specifies the maximum number of historical commands in a transaction that can be printed in the log after a deadlock occurs.

Parameter type: integer.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 1 to 1000

Default value: **1**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the number of commands in a single transaction. You are advised to set this parameter to a value less than 10. The database pre-allocates memory. The memory size is the value of **track_activity_query_size** multiplied by the maximum number of threads and then multiplied by the value of **track_activity_history_query_number**. **max_connections** is the factor that affects the maximum number of threads. If **max_connections** is set to a large value, the maximum number of estimated threads increases (the maximum number of threads may exceed 500,000). As a result, a large memory is allocated. In this case, exercise caution when increasing the number specified by **track_activity_history_query_numbers**.

Risks and impacts of improper settings: If the value is too large, a large number of logs will be printed and a large memory is allocated when a deadlock occurs.

enable_xid_abort_check

Parameter description: Specifies whether to verify transaction ID rollback is enabled when a transaction is committed.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_partition_ddl_lowlevel_lock

Parameter description: Locks a partitioned table during the execution of the DDL statements (ADD, DROP, TRUNCATE, EXCHANGE, MOVE, and RENAME). This parameter specifies the lock level of a table. If this parameter is set to **on**, a level-4 lock is added to the table, and DDL/DML operations can be concurrently performed on other partitions of the table. If this parameter is set to **off**, a level-8 lock is added to the table, and DDL/DML operations cannot be concurrently performed on other partitions of the table.

Parameter type: Boolean.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **off** for an upgraded database instance and **on** for a newly installed database instance.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **on**, DDL/DML operations can be concurrently performed on other partitions of a table, which has high performance. However, due to factors such as network delay, the execution sequences of concurrent DDL and DML operations on different DNs may be different, causing distributed deadlocks between DNs. If this parameter is set to **off**, DDL/DML operations cannot be concurrently performed on other partitions of the table. This has no performance advantage but no deadlock risk.

6.3.16 Version and Platform Compatibility

6.3.16.1 Compatibility with Earlier Versions

This section describes the parameters that control the backward compatibility and external compatibility of GaussDB. A backward compatible database supports applications of earlier versions. This section describes parameters used for controlling backward compatibility of a database.

array_nulls

Parameter description: Specifies whether the array input parser recognizes unquoted NULL as a null array element. This parameter is not supported in M-compatible mode.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Null values can be entered in arrays.
- **off**: Old behaviors are backward compatible. Arrays containing the value **NULL** can still be created when this parameter is set to **off**.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

backslash_quote

Parameter description: Determines whether a single quotation mark can be represented by \' in a string text.

NOTE

When the string text meets the SQL standards, \ has no other meanings. This parameter only affects the handling of non-standard-conforming string texts, including escape string syntax (E'...').

Parameter type: enumerated type.

Unit: none

Value range:

- **on**: The use of \' is always allowed.
- **off**: The use of \' is not allowed.
- **safe_encoding**: The use is allowed only when client-side encoding does not contain a backslash (\) at the end of a multibyte character for an ASCII value.

Default value: **safe_encoding**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

escape_string_warning

Parameter description: Specifies whether to warn against backslash (\) escapes used in ordinary character strings.

- If you need to use a backslash (\) as an escape, you can use escape string syntax (E'...'). This is because the default behavior of ordinary character strings treats the backslash as an ordinary character in each SQL standard.
- This variable can be enabled to help locate codes that need to be changed.
- If E'...' is used as an escape, logs may be incomplete in some scenarios.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Warn against backslash escapes used in ordinary character strings.
- **off**: Do not warn against backslash escapes used in ordinary character strings.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

quote_all_identifiers

Parameter description: Specifies whether to forcibly quote all identifiers even if they are not keywords when the database generates SQL. This will affect the output of **EXPLAIN** and the results of functions, such as pg_get_viewdef. For details, see the **--quote-all-identifiers** parameter in **gs_dump for Exporting Database Information**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The forcible quoting is enabled.
- **off**: The forcible quoting is disabled.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

sql_inheritance

Parameter description: Specifies the access policy of an inherited table. This parameter is set to be compatible with earlier versions.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Subtables can be accessed.
- **off**: Subtables cannot be accessed. That is, the ONLY keyword is used by default.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

standard_conforming_strings

Parameter description: Specifies whether ordinary string texts ('...') regard backslashes (\) as ordinary texts as specified in the SQL standard in a non-M-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

compatible database. Applications can check this parameter to determine how string texts will be processed. You are advised to use the escape string syntax (E'...') to escape characters. This syntax takes precedence over this parameter. Even if this parameter is set to **on**, backslashes (\) in escape strings are still interpreted as escape characters. The setting of this parameter does not take effect in an M-compatible database.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Backslashes are regarded as ordinary texts as specified in the SQL standard.
- **off**: Backslashes are regarded as escape characters in the string literal value.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

synchronize_seqscans

Parameter description: Specifies whether parallel sequence scans are synchronized with each other.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The scan may start from the middle of the table and then overwrite all rows in surround mode. The purpose is to keep synchronization with the scan that is being executed. This synchronization mechanism can improve the performance of parallel sequence scanning, but queries without using ORDER BY may return out-of-order results.
- **off**: The synchronization behavior of order scans is disabled. Each scan starts from the start position of the table. This ensures that queries without using ORDER BY return results that are consistent with the physical storage sequence, but parallel scanning efficiency may be reduced.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_beta_features

Parameter description: Specifies whether to enable some features that are not officially released and are used only for POC verification, such as GDS table join.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The features are enabled for compatibility with earlier versions.
- **off**: The features are disabled.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Exercise caution when enabling these extended features because they may cause errors in some scenarios.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_recordtype_check_strict

Parameter description: Specifies whether to perform strict check on the record type in PL/SQL. For details, see the following description.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The record type created during compilation of stored procedures, functions, and packages is checked for unsupported functions, and the record type created by provided stored procedures and functions is checked for the NOT NULL function.
- **off**: The backward compatibility is performed and the record type is not checked.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **off**, some *record* variables that specify the NOT NULL constraint and define the default value of a column do not take effect during value assignment.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If the current database version is directly installed, the default value of this parameter is **on**. For a database upgraded from an earlier version, if the source version does not support this parameter, the default value of this parameter is **off** after the upgrade. If the source version supports this parameter, the current setting of this parameter is retained after the upgrade.
- After this parameter is enabled, the following three changes occur:
  - For the record type created in a stored procedure or function, the NOT NULL constraint of the columns takes effect.
  - If **NOT NULL** or **DEFAULT** is specified for a column of the record type created in a package, a compilation error is reported. Variables created by accessing the package.rec type do not support **NOT NULL** and **DEFAULT**.
  - If **NOT NULL** or **DEFAULT** is specified for a column of another type with a nested record type, a compilation error is reported. For a variable nested with the record type, the record element of the variable does not support **NOT NULL** and **DEFAULT**.

plsql_block_concat_typename

Parameter description: Specifies the combination mode of user-defined type names in PL/SQL subblocks.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: A user-defined type in the PL/SQL subblock is used. The name of the user-defined type in the SQL statement is in the format of *OID*.[*m.n*···].*typename*, where *OID* is the OID of the stored procedure, *typename* is the name of the user-defined type, and *m* and *n* are sequence numbers of subblocks. When a plurality of layers of subblocks are nested, a plurality of layers of sequence numbers are required. If there is only a single layer of subblocks, only *m* is required.
- **off**: The backward compatibility in original mode is used. The name format is *OID*.*typename*.

Default value:

- If a new version is directly installed, the default value is **on**.
- For a database upgraded from an earlier version, if the source version does not support the GUC parameter, the default value is **off**. If the source version supports the GUC parameter, the default value is **on**.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is **off** and the user-defined types in a subblock have the same name, the SQL statement cannot distinguish the user-defined types of different subblocks.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

default_with_oids

Parameter description: Specifies whether **CREATE TABLE** and **CREATE TABLE AS** include an **OID** field in newly-created tables if neither **WITH OIDS** nor **WITHOUT OIDS** is specified. It also determines whether OIDs will be included in tables created by **SELECT INTO**.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that **CREATE TABLE** and **CREATE TABLE AS** can include an **OID** column in newly-created tables.
- **off** indicates that **CREATE TABLE** and **CREATE TABLE AS** cannot include any **OID** column in newly-created tables.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: It is not recommended that OIDs be used in user tables. Therefore, this parameter is set to **off** by default. When OIDs are required for a particular table, **WITH OIDS** needs to be specified during the table creation.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

system_view_version

Parameter description: Determines the version of the system view. For details, see **Table 1 System view version parameters**. All versions are backward compatible. For example, when **system_view_version** is set to **3**, all features of version 2 and version 1 are also supported.

Parameter type: integer.

Unit: none

Value range: 0 to 9999

Default value: **1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: In the upgrade scenario, do not change the value. In the installation scenario, set this parameter to the latest version listed in **Table 1 System view version parameters**.

Risks and impacts of improper settings: The behavior of some columns in the system view is changed. For details, see **Table 1 System view version parameters**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-28 System view version parameters

Value	Description
0	Default behavior.
1	When no prepared transaction exists in GaussDB, the value of the **PREPARECOUNT** column in the V$GLOBAL_TRANSACTION view changes from **NULL** to **0**.

enable_subtype_inherit_notnull_constraint

Parameter description: Specifies whether the NOT NULL constraint of subtype takes effect when subtype is used as a subelement of the set type and array type.

Parameter type: Boolean

Unit: none

Value range:

- **on**: When subtype is used as a subelement of the set type and array type, the NOT NULL constraint of subtype takes effect and no null value can be assigned to the element.
- **off**: When subtype is used as a subelement of the set type and array type, the NOT NULL constraint of subtype does not take effect and a null value can be assigned to the element.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **off** and subtype is used as a subelement of the set type and array type, the NOT NULL constraint of subtype does not take effect.

NOTE

If the current database version is directly installed, the default value of this parameter is **on**. For a database upgraded from an earlier version, if the source version does not support this parameter, the default value of this parameter is **off** after the upgrade. If the source version supports this parameter, the current setting of this parameter is retained after the upgrade.

enable_create_rule_on_select_to_table

Parameter description: Specifies whether the operation is successful when an "ON SELECT TO ... DO INSTEAD" rule named "_RETURN" is created on a table to convert the table to a view.

Parameter type: Boolean.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **on**: The "ON SELECT TO ... DO INSTEAD" rule named "_RETURN" is successfully created on the table to convert the table to a view.
- **off**: The "ON SELECT TO ... DO INSTEAD" rule named "_RETURN" fails to be created on the table to convert the table to a view.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Creating an "ON SELECT TO ... DO INSTEAD" rule named "_RETURN" on a table to convert the table to a view is an insecure operation and may cause unexpected errors. Views created in this way cannot be recompiled after they become invalid. Before modifying this parameter, make sure you understand its meaning and fully verify the change in a test environment.

6.3.16.2 Platform and Client Compatibility

Many platforms use the database system. External compatibility of the database system provides great convenience for platforms.

a_format_date_timestamp

Parameter description: Specifies whether to return the date and time. In ORA-compatible mode, when a transaction is started, the functions current_date(), current_timestamp(), and localtimestamp() return the timestamp when the current SQL statement is started.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Returns the timestamp when the current SQL statement is started.
- **off**: Returns the date or date and time when the transaction is started.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. This parameter can be enabled when the system time needs to be returned when a transaction is started.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

When the value of **a_format_version** is **10c** and the value of **a_format_dev_version** is **s7**, this GUC parameter can be enabled. For details, see **a_format_dev_version**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

compatiable_o_function_version

Parameter description: Specifies the version number to change the logic for selecting overloaded system functions. This parameter takes effect only in ORA-compatible mode.

Parameter type: string

Unit: none

Value range:

- **"auto"**: indicates that the overloaded system functions added from 507.1.0 are selected.
- *Specific version number*: If the version number is 507.1.0 or later, the overloaded system functions added from 507.1.0 to the specified version are selected.

Default value: **"auto"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Adjusting this parameter will change the logic for selecting overloaded system functions, which may result in the selection of incorrect overloaded system functions. Exercise caution when performing this operation.

NOTE

If the selected version is 507.1.0.SPC1100, set this parameter to **507.1.0.1100**. If the selected version is 507.1.0 (without an SPC version number), set this parameter to **507.1.0.0**.

convert_string_to_digit

Parameter description: Specifies the implicit conversion priority, which specifies whether to preferentially convert strings into numbers.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that strings are preferentially converted into numbers.
- **off** indicates that strings are not preferentially converted into numbers.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is modified, the internal data type conversion rule will be modified, causing unexpected behavior. Exercise caution when performing this operation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

character_set_connection

Parameter description: Specifies the character set of constant strings. If this parameter is modified, **collation_connection** is changed to the default collation of the character set.

- This parameter takes effect when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s2'**.
- This parameter takes effect in M-compatible mode.

Parameter type: string

Unit: none

Value range:

- Character sets supported in the compatibility mode whose **sql_compatibility** is set to **'MYSQL'**. For details, see **COLLATE collation**.
- In M-compatible mode, the value range is the supported character set. For details, see "SQL Reference > Character Sets and Collations" in *M Compatibility Developer Guide*.

Default value: same as the value of **server_encoding**.

Setting method: This is a USERSET parameter, which can only be set using method 3, and cannot be set using the GUC tools. For details about how to set this parameter, see **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the configured character set is inconsistent with the current environment, transcoding may be performed for multiple times, affecting performance.

collation_connection

Parameter description: Specifies the collation of a constant string. If this parameter is modified, **character_set_connection** is changed to the default character set of the collation.

- This parameter takes effect when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s2'**.
- This parameter takes effect in M-compatible mode.

Parameter type: string

Unit: none

Value range:

- Collations supported in the compatibility mode whose **sql_compatibility** is set to **'MYSQL'**. For details, see **COLLATE collation**.
- In M-compatible mode, the value range is the supported collations. For details, see "SQL Reference > Character Sets and Collations" in *M Compatibility Developer Guide*.

Default value: default collation of the current **server_encoding**. If it has no default collation, the value is **default**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter, which can only be set using method 3, and cannot be set using the GUC tools. For details about how to set this parameter, see **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the configured collation is inconsistent with the current environment, transcoding may be performed for multiple times, affecting performance.

character_set_results

Parameter description: Specifies the character set of the returned result.

- This parameter takes effect when **sql_compatibility** is set to **'MYSQL'**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s2'**.
- This parameter takes effect in M-compatible mode.

Parameter type: string

Unit: none

Value range:

- Character sets supported in the compatibility mode whose **sql_compatibility** is set to **'MYSQL'**. For details, see **COLLATE collation**.
- In M-compatible mode, the value range is the supported character set. For details, see "SQL Reference > Character Sets and Collations" in *M Compatibility Developer Guide*.
- You can also set this parameter to **null** or **''**. In this case, the character set of the database is output.

Default value: same as the value of **client_encoding**.

Setting method: This is a USERSET parameter, which can only be set using method 3, and cannot be set using the GUC tools. For details about how to set this parameter, see **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the configured character set is inconsistent with the current environment, garbled characters may be displayed.

enable_multiple_charset

Parameter description: Specifies whether tables, columns, and schemas whose character sets are different from the character set of the current database can be created in M-compatible mode.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: You can create tables, columns, and schemas whose character sets are different from the character set of the current database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **off**: You cannot create tables, columns, and schemas whose character sets are different from the character set of the current database.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_view_invalidation

Parameter description: Specifies whether to enable or disable the view invalidation function. After this parameter is enabled, if the object on which a view depends is changed, the upper-layer views that depend on the object become invalid.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The view invalidation function is enabled.
- **off**: The view invalidation function is disabled.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after you fully understand the parameter meaning and test the parameter to avoid unexpected results.

NOTE

For details about how to recompile an invalid view, see "Feature Usage Guidelines > Recompilation upon Invalidation" in *Developer Guide*.

enable_proc_exec_check

Parameter description: Specifies whether to check the validity of dependent objects when a stored procedure is executed. After this parameter is enabled, if the dependent object is invalid, the current stored procedure is not executed and an error is reported.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The validity check is performed when the stored procedure is executed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **off**: The validity check is not performed when the stored procedure is executed.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the actual scenario.

Risks and impacts of improper settings: Change the parameter value after you fully understand the parameter meaning and test the parameter to avoid unexpected results.

NOTE

If the version is upgraded from an earlier version to 507.0 or later, this parameter is set to **off** by default. If the newly installed version is 507.0 or later, this parameter is set to **on** by default.

enable_show_view_original_definition

Parameter description: Specifies whether to display the original SELECT statement entered by a user when the view definition is queried in an ORA-compatible database.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The original SELECT statement entered by the user is displayed when the view definition is queried.
- **off**: The SELECT statement is displayed by reversely parsing the QUERY TREE statement when querying the view definition.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

In ORA-compatible mode, this parameter affects the display result of functions or views (for example, \d+, dba_views, all_views, user_views, information_schema.views, pg_views, and pg_get_viewdef) that query view information.

fixed_date

Parameter description: Specifies the value of **sysdate** that can be fixed in an ORA-compatible database.

Parameter type: string

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range: all formats supported by timestamp. The value range is the time range (0001 to 9999) represented by **datea**. The date cannot be set to a negative value, which indicates the BC time. To indicate BC time, append **'BC'** to the end of the time.

Default value: **""**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing. This parameter cannot be enabled in the production environment and is used only for debugging.

mapping_date_to_datea

Parameter description: Specifies whether to replace the date type with the datea type in ORA-compatible mode.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The date type is replaced with the datea type.
- **off**: The behavior of the date type is retained in the default state.

Default value: **on** for a newly installed database. If the GUC parameter does not exist in the basic version to be upgraded, the default value is **off**. If the GUC parameter exists in the basic version to be upgraded, the original setting is retained.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. The value can be specified during database installation. If the value is not specified, the default value **on** is used, indicating that the date type is replaced by the datea type. You can enable or disable this parameter during installation as follows:

```
gs_install --dn-guc="mapping_date_to_datea=off" --cn-guc="mapping_date_to_datea=off" --gsinit-parameter="-- mapping_date_to_datea off"
```

Risks and impacts of improper settings: In ORA-compatible mode, the range, precision, and format of the date type may not be as expected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This parameter does not take effect during the upgrade observation period.
- This parameter is specified during database installation. You are advised not to modify this parameter after the database is started. If you modify the parameters during database running, the system views, system catalogs, and data related to the date type before the modification remain unchanged. The new object replaces the date type with the datea type.
- In the upgrade scenario, if the value of this parameter is changed after the upgrade is committed, the data created in the source version remains unchanged. For example, if this parameter is set to **off** in an earlier version and is set to **on** in a later version, the date type data created in the earlier version is still mapped to the timestamp type. To use the new datea type, create an object again according to the description in the data type.

nls_date_format

Parameter description: In Oracle-compatible mode, it is used for format control during the conversion between the DATEA type and the character type.

Parameter type: string

Unit: none

Value range: For details about the formats supported by **nls_date_format**, see **Table 1-12**. The following characters can be used as separators: hyphen (-), slash (/), comma (,), period (.), percent (%), colon (:), or a string enclosed in double quotation marks ("").

Default value: **"DD-MON-RR"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements. You are advised not to change the value frequently.

Risks and impacts of improper settings: The format verification may fail when the datea type is entered. As a result, an error is reported, or the entered time is not as expected.

nls_timestamp_format

Parameter description: Specifies the default timestamp format.

Parameter type: string

Unit: none

Value range: a string.

Default value: **"DD-Mon-YYYY HH:MI:SS.FF AM"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

nls_timestamp_tz_format

Parameter description: Specifies the default timestamp with time zone format in Oracle-compatible mode.

Parameter type: string

Unit: none

Value range: a string.

Default value: **"DD-Mon-YYYY HH:MI:SS.FF AM"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

This parameter is valid only when **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1**.

nls_nchar_characterset

Parameter description: Specifies the national character set. This parameter is used together with the nchr(cvalue int|bigint) system function.

Parameter type: string.

Unit: none

Value range: **"AL16UTF16"** and **"UTF8"** (case-insensitive).

Default value: **"AL16UTF16"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

- The database does not support the national character set. This parameter is used only to be compatible with the Oracle Database and obtain the national character set for users.
- This GUC parameter applies only to the nchr(cvalue int|bigint) function.

group_concat_max_len

Parameter description: This parameter is used together with the group_concat() function to limit the length of the return value. If the length exceeds the limit, the return value is truncated and an error or alarm is generated. If **sql_mode** is set to

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

a value containing **STRICT_TRANS_TABLES**, an error is reported in the DML statement. In other scenarios, an alarm is reported.

Parameter type: integer.

Unit: none

Value range: 0 to 9223372036854775807. Currently, the maximum length that takes effect is 1073741823. If the length exceeds this limit, the out of memory error is reported.

Default value: 1024

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a small value, the result returned by group_concat() may be truncated.

lastval_supported

Parameter description: Specifies whether the lastval() function can be used.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The lastval() function can be used and the nextval() function cannot be pushed down.
- **off**: The lastval() function cannot be used and the nextval() function can be pushed down.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_function_args

Parameter description: Specifies the maximum number of parameters allowed for a function.

Parameter type: integer.

Unit: none

Value range: **8192**

Default value: **8192**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

lower_case_table_names

Parameter description: Specifies whether the user table names (including table aliases), user view names, schema names (internal database names), and values of the columns corresponding to the preceding objects in the information_schema system view are case-sensitive in an M-compatible database. In the current version, the value of this parameter is **0** and cannot be changed.

Parameter type: integer.

Unit: none

Value range:

- **0**: User table names, user view names, and schema names (internal database names) are stored in system catalogs in the capitalization style specified in the CREATE TABLE, CREATE VIEW, and CREATE SCHEMA (CREATE DATABASE) statements. In the information_schema system view, the values of the columns corresponding to the preceding objects are case-sensitive.
- **1**: User table names are stored in lowercase in system catalogs. Names are case-insensitive. During storage and query operations, all user table names are converted to lowercase. This behavior also applies to schema names (internal database names), table aliases, user view names, and values of the corresponding columns in the information_schema system view.

Default value: **0**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

max_subpro_nested_layers

Parameter description: Specifies the maximum nesting depth of nested subprocedures.

Parameter type: integer.

Unit: none

Value range: 0 to 100

NOTE

When this parameter is set to **0**, nested subprocedures are not allowed.

Default value: **3**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the maximum nesting depth.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

transform_null_equals

Parameter description: Specifies whether expressions of the form **expr=NULL** (or **NULL=expr**) are treated as **expr IS NULL**. The correct SQL-standard-compliant behavior of **expr=NULL** is to always return **NULL** (unknown). Filtered forms in Microsoft Access generate queries that appear to use **expr=NULL** to test for null values. After you enable this option, you can use this API to access the database.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Expressions of the form **expr=NULL** (or **NULL=expr**) are treated as **expr IS NULL** They return true if expr evaluates to the **NULL** value, and false otherwise.
- **off**: **expr=NULL** always returns **NULL** (unknown).

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

New users are always confused about the semantics of expressions involving **NULL** values. Therefore, **off** is used as the default value.

support_extended_features

Parameter description: Specifies whether extended database features are supported.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that extended database features are supported.
- **off** indicates that extended database features are not supported.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

sql_compatibility

Parameter description: Specifies the type of mainstream database with which the SQL syntax and statement behavior of the database is compatible.

Parameter type: enumerated type

Unit: none

Value range:

- **ORA**: Oracle-compatible.
- **TD**: Teradata-compatible.
- **MYSQL**: MySQL-compatible.
- **PG**: PG-compatible.
- **M**: M-compatible database.

Default value: **MYSQL**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

NOTE

- This parameter can be set only when you run the **CREATE DATABASE** command to create a database.
- In the database, this parameter must be set to a specific value. It can be set to **ORA** or **TD** and cannot be changed randomly. Otherwise, the setting is not consistent with the database behavior.
- In distributed database scenarios in M-compatible mode, if an alarm is generated on a CN, WARNING information is generated. If an alarm is generated on a DN, WARNING information cannot be generated.

b_format_behavior_compat_options

Parameter description: Specifies a MySQL-compatible database configuration item.

Parameter type: string

Unit: none

Value range: Currently, only B-compatible configuration items listed in **Table 1 MySQL-compatible configuration items** are supported. Use commas (,) to separate multiple compatibility configuration items.

Default value: **""**

NOTE

If **b_format_version** is not set to **''**, **b_format_behavior_compat_options** will be set to **"all"** and cannot be modified.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-29 MySQL-compatible configuration items

Configuration Item	Behavior
default_collation	Specifies whether to enable compatibility with earlier versions of the default collation. • If this item is not specified, the character set or collation of a column of the character type is not explicitly specified, and the collation of a table is empty, the column uses the default collation. • If this item is specified, the collation of a column of the character type inherits the collation of a table (if it is not empty). If the collation of a table is empty, the default collation corresponding to the database is used.
all	Determines whether to enable all syntax. **all** cannot be specified together with other configuration items. In the table, specifying all configuration items except **all** that are separated by commas is equivalent to specifying **all**.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Determines whether some MySQL-compatible features are available. If you want to modify this parameter, make sure you understand its meaning and modify it with caution to avoid risks caused by misoperations.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

m_format_behavior_compat_options

Parameter description: Specifies the configuration items of an M-compatible database.

Parameter type: string

Unit: none

Value range: Currently, only B-compatible configuration items listed in **Table 2 M-compatible configuration items** are supported. Use commas (,) to separate multiple compatibility configuration items.

Default value:

- For a newly installed database, the value is **"grant_database_nomapping"**.
- For a database upgraded from an earlier version, if the earlier version does not support this GUC parameter, the default value is **""** after the upgrade. If the earlier version supports this GUC parameter, the default value is retained after the upgrade.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-30 M-compatible configuration items

Configuration Item	Behavior

grant_database_nomapping	Configuration item of the GRANT | REVOKE ON DATABASE syntax behavior in the M-compatible database.
	• If this configuration item is not set, the GRANT | REVOKE ON DATABASE syntax is mapped to the GRANT | REVOKE ON SCHEMA behavior to grant or revoke the specified user or role's permission to access schema-level objects.
	• If this configuration item is set, the GRANT | REVOKE ON DATABASE syntax retains the behavior of the GRANT | REVOKE ON DATABASE syntax in the non-M-compatible database (that is, original GaussDB) of the default compatibility, and grants or revokes the access permission of a specified user or role on database objects.
	-- Create a user and a database.
	\c postgres
	DROP USER IF EXISTS testUser cascade;
	DROP DATABASE IF EXISTS m_db;
	CREATE USER testUser WITH PASSWORD '********';
	CREATE DATABASE m_db DBCOMPATIBILITY='M';
	\c m_db
	
	-- The option is not enabled and the schema is mapped to the GRANT SCHEMA. A message is displayed, indicating that the schema cannot be found.
	m_db=# SET m_format_behavior_compat_options='';
	SET
	m_db=# GRANT CREATE ON DATABASE m_db TO testUser;
	ERROR: schema "m_db" does not exist
	m_db=# SET SESSION AUTHORIZATION testUser PASSWORD '********';
	SET
	m_db=> CREATE SCHEMA s1;
	ERROR: Permission denied for database m_db.
	DETAIL: N/A.
	m_db=> \c -
	
	-- Enable the option. The CREATE SCHEMA permission is granted.
	m_db=# SET m_format_behavior_compat_options='grant_database_nomapping';
	SET
	m_db=# GRANT CREATE ON DATABASE m_db TO testUser;
	GRANT
	m_db=# SET SESSION AUTHORIZATION testUser PASSWORD '********';
	SET
	m_db=> CREATE SCHEMA s1;
	CREATE SCHEMA
	m_db=> \c -
	
	-- Clean up.
	\c postgres
	DROP USER IF EXISTS testUser cascade;
	DROP DATABASE IF EXISTS m_db;

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

system_function_version

Parameter description: Specifies the configuration item of the database system function behavior.

Parameter type: string

Unit: none

Value range: Currently, only compatibility configuration items listed in **Table 6-31** are supported. Use commas (,) to separate multiple compatibility configuration items, for example, **"SET system_function_version='get_hash_value_fix,pbe_trunc_fix,empty_array_fix';"**.

Default value:

- During installation: **get_hash_value_fix,pbe_trunc_fix,empty_array_fix,regex_like_m_fix,parse_phrase_fix**
- During upgrade: **""**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-31 Compatibility configuration items

Configuration Item	Behavior

get_hash_value_fix	Specifies whether to enable the **dbe_utility.get_hash_value** compatibility with earlier versions.
	● If this configuration item is not set, the last four bytes of the character string entered by the user are ignored. As a result, the hash values of the character strings less than four bytes are the same.
	● If this configuration item is set, the problem is resolved to ensure that the hash values are correct.
	gaussdb=# SET system_function_version = '';
	gaussdb=# SELECT dbe_utility.get_hash_value(1, 1024, 1024);
	get_hash_value
	---------------- 	
	1444
	(1 row)
	
	gaussdb=# SELECT dbe_utility.get_hash_value(2, 1024, 1024);
	get_hash_value
	---------------- 	
	1444
	(1 row)
	
	gaussdb=# SET system_function_version = 'get_hash_value_fix';
	gaussdb=# SELECT dbe_utility.get_hash_value(1, 1024, 1024);
	get_hash_value
	---------------- 	
	1785
	(1 row)
	
	gaussdb=# SELECT dbe_utility.get_hash_value(2, 1024, 1024);
	get_hash_value
	---------------- 	
	1538
	(1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

pbe_trunc_fix	Specifies whether to enable the compatibility with earlier versions of the lpad/rpad/chr/round/trunc/ add_months/instrb function in the PBE scenario.
	● If this configuration item is not set, the decimal points of the input parameters of the preceding functions are rounded off to integers in the PBE scenario.
	● If this configuration item is set, the decimal points of the input parameters of the preceding functions are truncated to integers in the PBE scenario to ensure that the result is the same as the result in ORA-compatible mode.
	`gaussdb=# SET a_format_version='10c';`
	`gaussdb=# SET a_format_dev_version='s1';`
	`gaussdb=# PREPARE pround1 AS SELECT round(123.456,$1);`
	`gaussdb=# EXECUTE pround1('2.6');`
	`round`
	`---------`
	`123.456`
	`(1 row)`
	
	`gaussdb=# SET system_function_version = 'pbe_trunc_fix';`
	`gaussdb=# PREPARE pround1 AS SELECT round(123.456,$1);`
	`gaussdb=# EXECUTE pround1('2.6');`
	`round`
	`---------`
	`123.46`
	`(1 row)`
	NOTE
	This parameter is valid only in an ORA-compatible database.

empty_array_fix	Specifies whether to enable the compatibility with earlier versions of empty arrays.
	● If this configuration item is not set, the empty array constructed by the array_positions function is different from that constructed by **'{}'::int[]**.
	● If this configuration item is set, the empty array constructed by the array_positions function is the same as that constructed by **'{}'::int[]**.
	`gaussdb=# SET system_function_version = 'empty_array_fix';`
	`SET`
	`gaussdb=# SELECT array_eq(array_positions(array[1, 1, 1], 3),`
	`'{}'::int[]);`
	`array_eq`
	`----------`
	`t`
	`(1 row)`
	
	`gaussdb=# SET system_function_version = '';`
	`SET`
	`gaussdb=# SELECT array_eq(array_positions(array[1, 1, 1], 3),`
	`'{}'::int[]);`
	`array_eq`
	`----------`
	`f`
	`(1 row)`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
regex_like_m_fix	Specifies the compatibility with earlier versions switch that controls the behavior of the regex_like_m function. (The regexp_like function calls regex_like_m, which is also affected.) ● If this configuration item is not set, the matching result is incorrect when multiple rows are matched and the regular expression rule contains |. ● If this configuration item is set, the matching result is correct when multiple rows are matched and the regular expression rule contains |. gaussdb=# SET system_function_version = 'regex_like_m_fix'; SET gaussdb=# SELECT 'abc' FROM dual WHERE regex_like_m('111 2 222','^1+$\|^2+$'); ?column? ---------- abc (1 row) gaussdb=# SET system_function_version = ''; SET gaussdb=# SELECT 'abc' FROM dual WHERE regex_like_m('111 2 222','^1+$\|^2+$'); ?column? ---------- (0 row) **NOTE** This parameter takes effect only for the database upgraded from a version where the matching result is incorrect. For a newly installed database, the behavior is the same as that when this parameter is set, regardless of whether this parameter is set.
parse_phrase_fix	Specifies whether to parse phrase operators in tsquery. ● If this parameter is not set, phrase operators are not parsed in operations such as **to_tsquery** that involve tsquery parsing. ● If this parameter is set, phrase operators are parsed in operations such as **to_tsquery** that involve tsquery parsing. gaussdb=# SET system_function_version = 'parse_phrase_fix'; SET gaussdb=# SELECT to_tsquery('hello <-> world'); to_tsquery ---------- 'hello' <-> 'world' (1 row) gaussdb=# SET system_function_version = ''; SET gaussdb=# SELECT to_tsquery('hello <-> world'); ERROR: syntax error in tsquery: "hello <-> world" CONTEXT: referenced column: to_tsquery

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. During the upgrade, the parameter is left empty by default. If you need to enable the setting, you are advised to modify the GUC configuration file to ensure that the setting is not lost when the database is restarted.

Setting suggestion: Specifies whether some system function compatibility features are available. If you want to modify this parameter, make sure you understand its meaning and modify it with caution.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

behavior_compat_options

Parameter description: Specifies the database compatibility configuration item.

Parameter type: string

Unit: none

Value range: Currently, only compatibility configuration items listed in **Table 6-32** are supported. Use commas (,) to separate multiple compatibility configuration items, for example, **set behavior_compat_options='end_month_calculate,display_leading_zero'**.

Default value:

- For a newly installed database, the default value is **"enable_bpcharlikebpchar_compare,enable_crosstype_integer_operator,forbid_skip_tableof_empty_str_elem,enable_ora_joinop_in_updatestmt"**.
- For a database upgraded from an earlier version, if the earlier version does not support this GUC parameter, the default value is **""** after the upgrade. If the earlier version supports this GUC parameter, the default value is retained after the upgrade.

NOTE

When **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s7**, some compatibility configuration items are supported. For details about the parameters, see **a_format_dev_version**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-32 Compatibility configuration items

Configuration Item	Behavior
display_leading_zero	Specifies how floating-point numbers are displayed. It controls the display of zeros before the decimal point of all character string types (such as char, character, nchar, varchar, character varying, varchar2, nvarchar2, text, and clob) and arbitrary-precision types (such as float4, float8, and numeric) in the numeric type, specifies whether the length of the number calculated by **length** contains zeros before the decimal point. ● If this item is not specified, for a non-zero decimal number between –1 and +1, the 0 before the decimal point is not displayed. For example: `gaussdb=# SELECT 0.1231243 as a, 0.1231243::numeric as b,0.1231243::integer(10,3) as c, length(0.1242343) as d;` ` a	b	c	d` `----------+----------+------+---` `.1231243	.1231243	.123	8` `(1 row)` ● If this item is specified, for a non-zero decimal number between –1 and +1, the 0 before the decimal point is displayed. For example: `gaussdb=# select 0.1231243 as a, 0.1231243::numeric as b,0.1231243::integer(10,3) as c, length(0.1242343) as d;` ` a	b	c	d` `----------+----------+-------+---` `0.1231243	0.1231243	0.123	9` `(1 row)`

end_month_calculate	Specifies the calculation logic of the add_months function.
	This parameter does not take effect in M-compatible databases.
	Assume that the two parameters of the add_months function are **param1** and **param2**, and that the month of **param1** and **param2** is **result**.
	● If this item is not specified, and the **Day** of **param1** indicates the last day of a month shorter than **result**, the **Day** in the calculation result will equal that in **param1**. For example:
	`gaussdb=# select add_months('2018-02-28',3) from sys_dummy;`
	`add_months`
	`---------------------`
	`2018-05-28 00:00:00`
	`(1 row)`
	● If this item is specified, and the **Day** of **param1** indicates the last day of a month shorter than **result**, the **Day** in the calculation result will equal that in **result**. For example:
	`gaussdb=# select add_months('2018-02-28',3) from sys_dummy;`
	`add_months`
	`---------------------`
	`2018-05-31 00:00:00`
	`(1 row)`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
compat_analyze_sample	Specifies the sampling behavior of the ANALYZE operation. If this item is specified, the sample collected by the ANALYZE operation will be limited to around 30,000 records, controlling CN memory consumption and maintaining the stability of ANALYZE.
bind_schema_tablespace	Binds a schema with the tablespace with the same name. If a tablespace name is the same as *schema_name*, **default_tablespace** will also be set to *schema_name* if **search_path** is set to *schema_name*.
bind_procedure_searchpath	Specifies the search path of the database objects in a stored procedure for which no schema name is specified. If the schema name is implicitly specified for a stored procedure, the schema to which the stored procedure belongs is searched preferentially. If the stored procedure is not found, the following operations are performed: ● If this item is not specified, the system reports an error and exits. ● If this item is specified, the search continues based on the settings of **search_path**. If the issue persists, the system reports an error and exits.
skip_insert_gs_source	If this parameter is enabled, no data is inserted into the dbe_pldeveloper.gs_source table when the PL/SQL objects are created.
a_hash_bpchar	If this parameter is enabled, the hashbpchar function retains spaces at the end of a string to calculate the hash value.

correct_to_number	Specifies the compatibility of the to_number() result.
	● If this item is not set, the result of the to_number() function is the same as that in the Oracle Database by default.
	`gaussdb=# select '' AS to_number_14, to_number('34,50','999,99');`
	`ERROR:  invalid data.`
	`CONTEXT:  referenced column: to_number`
	● If this item is specified, the result of the to_number() function is the same as that of pg11.
	`gaussdb=# select '' AS to_number_14, to_number('34,50','999,99');`
	` to_number_14 | to_number`
	`---------------+-----------`
	`  |  3450`
	`(1 row)`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

unbind_divide_bound	Specifies the range check on the result of integer division.
	This parameter does not take effect in M-compatible databases.
	● If this item is not specified, the range of the division result is verified. For example, an out-of-bounds error is reported because the output result of *INT_MIN*/**(-1)** is greater than *INT_MAX*.
	`gaussdb=# select (-2147483648)::int4 / (-1)::int4;`
	`ERROR:  integer out of range`
	● If this item is specified, the range of the division result does not need to be verified. For example, the output result of *INT_MIN*/**(-1)** is *INT_MAX*+**1**.
	`gaussdb=# select (-2147483648)::int4 / (-1)::int4;`
	`  ?column?`
	`------------`
	` 2147483648`
	`(1 row)`

convert_string_digit_to_numeric	Specifies whether to convert numeric constants of the character string type to those of the numeric type before these two types are compared.
	This parameter does not take effect in M-compatible databases.
	● If this item is not specified, the numeric constants of the character string type are not converted to those of the numeric type.
	● If this item is specified, the numeric constants of the character string type are converted to those of the numeric type.
	`gaussdb=# create table test1 (c1 int, c2 varchar);`
	`gaussdb=# insert into test1 values (2, '1.1');`
	`gaussdb=# set behavior_compat_options='';`
	`gaussdb=# select * from test1 where c2 > 1;`
	`ERROR:  invalid input syntax for type bigint: "1.1"`
	
	`gaussdb=# set behavior_compat_options='convert_string_digit_to_numeric';`
	`gaussdb=# select * from test1 where c2 > 1;`
	` c1 | c2`
	`----+-----`
	`  2 | 1.1`
	`(1 row)`
	
	`gaussdb=# drop table test1;`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

return_null_string	Specifies how to display the empty result (empty string '') of the lpad() and rpad() functions.
	This parameter does not take effect in M-compatible databases.
	● If this item is not specified, the empty string is displayed as **NULL**.
	`gaussdb=# select length(lpad('123',0,'*')) from sys_dummy;`
	`length`
	`--------`
	`(1 row)`
	● If this item is specified, the empty string is displayed as single quotation marks ('').
	`gaussdb=# select length(lpad('123',0,'*')) from sys_dummy;`
	`length`
	`--------`
	`0`
	`(1 row)`

compat_concat_variadic	Specifies the compatibility of variadic results of the concat() and concat_ws() functions. The MySQL database does not have the variadic type. Therefore, this option has no impact on the MySQL database.
	● If this item is not set and the concat function parameter is of the variadic type, the results of the Oracle and Teradata databases in compatibility mode are the same by default.
	`gaussdb=# select concat(variadic NULL::int[]) is null;`
	` ?column?`
	`----------`
	` t`
	`(1 row)`
	● If this item is set and the concat function parameter is of the variadic type, different result formats of the ORA and TD databases in compatibility mode are retained.
	`-- In the Oracle Database:`
	`gaussdb=# select concat(variadic NULL::int[]) is null;`
	` ?column?`
	`----------`
	` t`
	`(1 row)`
	`-- In the TD database:`
	`gaussdb=# select concat(variadic NULL::int[]) is null;`
	` ?column?`
	`----------`
	` f`
	`(1 row)`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
merge_update_multi	Controls the UPDATE operation if a data record in the destination table conflicts with multiple source data records when MERGE INTO ... WHEN MATCHED THEN UPDATE (see **MERGE INTO**) and INSERT ... ON DUPLICATE KEY UPDATE (see **INSERT**) are used. • If this item is specified and the preceding scenario exists, the system performs multiple UPDATE operations on the conflicting row. • If this item is not specified (the default value is retained), an error is reported, indicating that the MERGE or INSERT operation fails.
plstmt_implicit_savepoint	Determines whether the execution of an UPDATE statement in a stored procedure has an independent subtransaction. If this parameter is set, the implicit savepoint is enabled before executing each UPDATE statement in the stored procedure, and the subtransaction is rolled backed to the latest savepoint in the EXCEPTION block by default, ensuring that only the modification of failed statements is rolled back. This option is used to be compatible with the EXCEPTION behavior of the Oracle Database.
hide_tailing_zero	Configuration item for numeric display. • If this item is specified, the trailing zeros after the decimal point are hidden in all scenarios where numeric values are output, even if the precision format is specified. • If this item is not specified, numeric data is displayed in the specified precision. For example: gaussdb=# set behavior_compat_options='hide_tailing_zero'; gaussdb=# select cast(123.123 as numeric(15,10)) as a, to_char(cast(123.123 as numeric(15,10)), '999D999999'); a | to_char ---------+---------- 123.123 | 123.123 (1 row) gaussdb=# set behavior_compat_options=''; gaussdb=# select cast(123.123 as numeric(15,10)) as a, to_char(cast(123.123 as numeric(15,10)), '999D999999'); a | to_char ----------------+------------- 123.1230000000 | 123.123000 (1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
truncate_numeric_tail_zero	Configuration item for numeric display. ● If this item is specified, the trailing zeros after the decimal point are hidden in all numeric output scenarios except explicit precisions, such as to_char(numeric, format). ● If this item is not specified, numeric data is displayed in the default precision. For example: gaussdb=# set behavior_compat_options='truncate_numeric_tail_zero'; gaussdb=# select cast(123.123 as numeric(15,10)) as a, to_char(cast(123.123 as numeric(15,10)), '999D999999'); a | to_char ---------+------------- 123.123 | 123.123000 (1 row) gaussdb=# set behavior_compat_options=''; gaussdb=# select cast(123.123 as numeric(15,10)) as a, to_char(cast(123.123 as numeric(15,10)), '999D999999'); a | to_char ----------------+------------- 123.1230000000 | 123.123000 (1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

char_coerce_compat	Controls the behavior when char(n) types are converted to other variable-length string types. This parameter is valid only when the **sql_compatibility** parameter is set to **ORA**. After this parameter is enabled, spaces at the end are not omitted in implicit conversion, explicit conversion, or conversion by calling the text(bpchar) function.
	● If this item is specified, spaces at the end are not omitted in conversion. In addition, if the length of the char(n) type exceeds that of other variable-length string types, an error is reported.
	● If this item is not specified, spaces at the end are omitted when the char(n) type is converted to another variable-length string type.
	gaussdb=# set behavior_compat_options='';
	gaussdb=# create table tab_1(col1 varchar(3));
	gaussdb=# create table tab_2(col2 char(3));
	gaussdb=# insert into tab_2 values(' ');
	gaussdb=# insert into tab_1 select col2 from tab_2;
	gaussdb=# select * from tab_1 where col1 is null;
	col1
	------ 	
	
	(1 row)
	gaussdb=# select * from tab_1 where col1=' ';
	col1
	------ 	
	(0 rows)
	gaussdb=# delete from tab_1;
	gaussdb=# set behavior_compat_options = 'char_coerce_compat';
	gaussdb=# insert into tab_1 select col2 from tab_2;
	gaussdb=# select * from tab_1 where col1 is null;
	col1
	------ 	
	(0 rows)
	gaussdb=# select * from tab_1 where col1=' ';
	col1
	------ 	
	
	(1 row)
	
	gaussdb=# drop table tab_1;
	gaussdb=# drop table tab_2;

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
aformat_regexp_match	Determines the matching behavior of regular expression functions. When this parameter is set and **sql_compatibility** is set to **ORA** or **MYSQL**, the meanings of the options supported by the **flags** parameter of the regular expression are as follows: 1. By default, the character **'\n'** cannot be matched. 2. If **flags** contains the **n** option, **'.'** can match the character **'\n'**. 3. The regexp_replace(source, pattern replacement) function replaces all matching substrings. 4. The regexp_replace(source, pattern, replacement, flags) function returns **null** when the value of **flags** is **null**. 5. When the value of **sql_compatibility** is **ORA** and the value of **flags** in the regexp_replace(source, pattern, replacement, flags) function is **''**, the return value is **null**. Otherwise, the meanings of the options supported by the **flags** parameter of the regular expression are as follows: 1. By default, the character **'\n'** can be matched. 2. The **n** option in **flags** indicates that the multi-line matching mode is used. 3. The regexp_replace(source, pattern replacement) function replaces only the first matched substring. 4. If the value of **flags** is **null**, the return value of the regexp_replace(source, pattern, replacement, flags) function is the character string after replacement. 5. When the value of **sql_compatibility** is **ORA** and the value of **flags** in the regexp_replace(source, pattern, replacement, flags) function is **''**, the return value is the character string after replacement. **NOTICE** You are advised not to change the status of this parameter during service execution. The status change of this parameter may change the return result of the regexp_replace function with the same input parameter. If the regexp_replace function is referenced in the expression index, the status change of this parameter may cause data exceptions.
disable_emptystr2null	In the ORA-compatible parameter binding scenario, if this parameter is enabled, the function of converting the values of parameters of the character type from an empty string to **null** by default, if any, is disabled. text, clob, blob, raw, bytea, varchar, nvarchar2, bpchar, char, name, byteawithoutorderwithqualcol, and byteawithoutordercol. This parameter is reserved for emergency. Do not set it unless necessary.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

select_into_return_null	If this parameter is enabled in PG-compatible mode, **NULL** values can be assigned to the variables in the stored procedure statement **SELECT** *select_expressions* **INTO [STRICT]** *target* **FROM...** without specifying **STRICT** when the query result is empty.
	After this parameter is enabled, if the returned result set is empty, the **NULL** value is assigned to the variable. For example:
	gaussdb=# SET behavior_compat_options = 'select_into_return_null';
	SET
	gaussdb=# CREATE TABLE t1 (a int, b int) DISTRIBUTE BY hash(a);
	CREATE TABLE
	gaussdb=# DECLARE
	gaussdb-# var int;
	gaussdb-# BEGIN
	gaussdb$# SELECT b INTO var FROM t1;
	gaussdb$# RAISE INFO '%', var;
	gaussdb$# END;
	gaussdb$# /
	INFO: <NULL>
	ANONYMOUS BLOCK EXECUTE
	gaussdb=# DROP TABLE t1;
	DROP TABLE

plsql_security_definer	• After this parameter is enabled, the definer permission is used by default when a stored procedure is created. • The default permission on a package is SECURITY INVOKER. To change the default permission to SECURITY DEFINER, set this parameter.
proc_outparam_override	• Determines the overloading of output parameters of a stored procedure. After this parameter is enabled, the stored procedure can be properly created and called even if only the output parameters of the stored procedure are different. Currently, this parameter can be used only when gsql and JDBC are used to connect to the database. If this parameter is enabled for other tools to connect to the database, stored procedures with the **out** parameter cannot be called. • It supports the functions that contain the **out** output parameter and returns data of the record type. Besides, a value is assigned to the **out** parameter. • Setting **behavior_compat_options** to **'proc_outparam_override'** is applicable only in ORA-compatible mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

proc_uncheck_default_param	When a function is called, the system does not check whether the default parameter is omitted.
	• If this item is not specified and a function with default parameters is called, input parameters are added to the function from left to right. If inputs of non-default parameters are missing, an error is reported. For example:
	gaussdb=# create or replace function test(f1 int, f2 int default 20, f3 int, f4 int default 40, f5 int default 50) return int
	gaussdb-# as
	gaussdb$# begin
	gaussdb$# raise info 'f1:%',f1;
	gaussdb$# raise info 'f2:%',f2;
	gaussdb$# raise info 'f3:%',f3;
	gaussdb$# raise info 'f4:%',f4;
	gaussdb$# raise info 'f5:%',f5;
	gaussdb$# return 1;
	gaussdb$# end;
	gaussdb$# /
	CREATE FUNCTION
	gaussdb=# select test(1,2);
	ERROR: function test(integer, integer) does not exist
	LINE 1: select test(1,2);
	^
	HINT: No function matches the given name and argument types. You might need to add explicit type casts.
	CONTEXT: referenced column: test
	
	gaussdb=# drop function test;
	• If this item is set and a function with default parameters is called, the inputs are added to the function from left to right. The number of default inputs matches the number of default parameters. If any input of a non-default parameter is missing, the parameter will be filled with a misplaced default value. For example:
	gaussdb=# create or replace function test(f1 int, f2 int default 20, f3 int, f4 int default 40, f5 int default 50) return int
	gaussdb-# as
	gaussdb$# begin
	gaussdb$# raise info 'f1:%',f1;
	gaussdb$# raise info 'f2:%',f2;
	gaussdb$# raise info 'f3:%',f3;
	gaussdb$# raise info 'f4:%',f4;
	gaussdb$# raise info 'f5:%',f5;
	gaussdb$# return 1;
	gaussdb$# end;
	gaussdb$# /
	CREATE FUNCTION
	gaussdb=# select test(1,2);
	INFO: f1:1
	CONTEXT: referenced column: test
	INFO: f2:2
	CONTEXT: referenced column: test
	INFO: f3:20
	CONTEXT: referenced column: test
	INFO: f4:40
	CONTEXT: referenced column: test
	INFO: f5:50
	CONTEXT: referenced column: test
	test

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
	------ 1 (1 row) gaussdb=# drop function test; As shown above, f3 is filled with an incorrect default value. **NOTE** In this scenario, a non-default parameter is filled with the previous default value.
dynamic_sql_compat	After this parameter is enabled: ● Duplicate parameters in the SQL statement template are not regarded as the same parameter, but match variables in the USING clause in sequence. ● If a stored procedure is called during dynamic statement execution, the IN and OUT attributes of the stored procedure and the USING clause are checked. ● Application scenarios: Execute immediate, open cursor for, return query, and for var in execute sql binding parameters. **NOTE** If a stored procedure is called when a dynamic statement executes an anonymous block statement, only the **IN** parameters are corrected. If **OUT** parameters need to be checked, set the **proc_outparam_override** parameter.
dynamic_sql_check	After this parameter is enabled, an error is reported during dynamic statement execution if the number of different template parameters in the dynamic statement template SQL is different from that of variables in the USING clause. Application scenarios: Execute immediate, open cursor for, return query, and for var in execute sql binding parameters. **NOTE** ● If the **dynamic_sql_compat** option is enabled, the **dynamic_sql_check** option does not take effect. ● If a stored procedure is called when a dynamic statement executes an anonymous block statement, only the **IN** parameters are checked. If **OUT** parameters need to be checked, set the **proc_outparam_override** parameter. ● If a stored procedure is called when a dynamic statement executes an anonymous block statement and the parameter is enabled, the **IN** and **OUT** attributes in a stored procedure and the USING clause are not checked.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

enable_funcname_with_argsname	If the parameter is enabled, the projection alias displays the complete function when SELECT is used to call a function.
	• If this item is not specified, the projection alias displays only the function name when SELECT is used to call a function. For example:
	gaussdb=# select power(2,3);
	power
	------- 	
	8
	(1 row)
	
	gaussdb=# select count(*) from db_ind_columns;
	count
	------- 	
	611
	(1 row)
	
	gaussdb=# select count(index_name) from db_ind_columns;
	count
	------- 	
	611
	(1 row)
	
	gaussdb=# select left('abcde', 2);
	left
	------ 	
	ab
	(1 row)
	
	gaussdb=# select pg_client_encoding();
	pg_client_encoding
	-------------------- 	
	UTF8
	(1 row)
	• If this item is specified, the projection alias displays the complete function when SELECT is used to call a function. For example:
	gaussdb=# set behavior_compat_options =
	'enable_funcname_with_argsname';
	SET
	gaussdb=# select power(2,3);
	power(2,3)
	------------ 	
	8
	(1 row)
	
	gaussdb=# select count(*) from db_ind_columns;
	count(*)
	---------- 	
	611
	(1 row)
	
	gaussdb=# select count(index_name) from db_ind_columns;
	count(index_name)
	------------------- 	
	611
	(1 row)
	
	gaussdb=# select left('abcde', 2);
	left('abcde',2)
	----------------- 	

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

	ab
	(1 row)
	
	gaussdb=# select pg_client_encoding();
	pg_client_encoding()
	---------- 	
	UTF8
	(1 row)
	• Currently, only the projection aliases in the func_name(args_list), func_name(), and func_name(*) formats can display complete functions. The **args** parameter can only be a character constant, numeric constant, column name, or function. A column name can contain a table name, but func_name(*) cannot have a table name before the asterisk. A function name can contain a schema or package name. The parameter cannot be an expression. The parameter can contain only the DISTINCT keyword. If the parameter contains other keywords, the function cannot be displayed completely.
	• When the input parameter of the function is in the following scenarios, the projection alias does not display the complete function:
	-- Create a database in ORA-compatible mode.
	gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA' ENCODING = 'UTF8';
	CREATE DATABASE
	gaussdb=# \c ora_db
	ora_db=# SET behavior_compat_options = 'enable_funcname_with_argsname';
	SET
	-- The input parameter of the function contains null values or empty strings in ORA-compatible mode.
	ora_db=# SELECT length(null);
	length
	-------- 	
	
	(1 row)
	
	ora_db=# SHOW sql_compatibility;
	sql_compatibility
	------------------- 	
	ORA
	(1 row)
	
	ora_db=# SELECT length('');
	length
	-------- 	
	
	(1 row)
	
	-- The input parameter of the function is a forcible type cast with double colons.
	ora_db=# SELECT length('abc'::text);
	length
	-------- 	
	3
	(1 row)
	
	-- The input parameter of the function is of a specified type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

	ora_db=# SELECT length(text'abc');
	 length
	-------- 	
	 3
	(1 row)
	
	-- The input parameter of the function is a PBE placeholder.
	ora_db=# PREPARE p1 AS SELECT length($1);
	PREPARE
	ora_db=# EXECUTE p1('abc');
	 length
	-------- 	
	 3
	(1 row)
	
	ora_db=# DEALLOCATE p1;
	DEALLOCATE
	
	-- The input parameter of the function is an array constructor.
	ora_db=# SELECT array_ndims(ARRAY[1, 2, 3]);
	 array_ndims
	------------- 	
	 1
	(1 row)
	• The projection alias varies with the clause following the function:
	– When the input parameter of the function contains other clauses (such as the ORDER BY clause), the projection alias displays only the function name.
	– When a function is followed by an OVER or WITHIN clause, the projection alias displays the function name and input parameter, but the OVER or WITHIN clause is not displayed.
	– When a function is followed by the KEEP clause, the projection alias displays the function name, but the input parameter and KEEP clause are not displayed.
	gaussdb=# \c ora_db
	ora_db=# SET behavior_compat_options =
	'enable_funcname_with_argsname';
	SET
	ora_db=# DROP TABLE IF EXISTS tbl;
	ora_db=# CREATE TABLE tbl (id int, lower_col text, col1 int, col2
	numeric);
	ora_db=# INSERT INTO tbl VALUES (1, 'hello', 42, 3.14);
	
	-- The input parameter contains the ORDER BY clause.
	ora_db=# SELECT string_agg(lower_col, '1' ORDER BY id) FROM tbl;
	 string_agg
	------------ 	
	 hello
	(1 row)
	
	-- OVER clause
	ora_db=# SELECT min(col1) OVER(PARTITION BY id ORDER BY col2)
	FROM tbl;
	 min(col1)
	----------- 	
	 42

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

	(1 row)
	
	-- WITHIN clause
	ora_db=# SELECT listagg(lower_col, ',') WITHIN GROUP(ORDER BY id) FROM tbl;
	 listagg(lower_col,',')
	----------------------- 	
	 hello,world,test
	(1 row)
	
	-- KEEP clause.
	ora_db=# SELECT min(col1) KEEP (DENSE_RANK FIRST ORDER BY col2) FROM tbl;
	 min
	----- 	
	 42
	(1 row)
	
	ora_db=# DROP TABLE IF EXISTS tbl;
	• Some functions can both be called by syntax or as system functions, causing the difference in the display of projection aliases. For details about these functions, see **Table 1-29**.
	– When a function is called only by function name, the projection alias does not display the complete function.
	– When a function is called with the pg_catalog prefix or double quotation marks, the projection alias displays the complete function.
	• If the projection column contains functions that can only be called by syntax, the projection alias does not display the complete function. For details about these functions, see **Table 1-29**. The projection alias of a function that can only be called by syntax may not display the function name itself. Instead, it displays the output identifier derived by the function during syntax parsing.
	– If the outermost function is a function that can only be called by syntax, only the name of the function is displayed in the projection alias.
	– If the outermost function supports the display of projection alias, but its parameters contain functions that can only be called by syntax, only the name of the outermost function is displayed.
	• If some secure encryption and decryption functions and masking functions are displayed completely by projection aliases, it may bring security problems. Therefore, only function names are displayed here, including: gs_encrypt_aes128, gs_decrypt_aes128, gs_encrypt, gs_decrypt, gs_encrypt_bytea, gs_decrypt_bytea, aes_encrypt, aes_decrypt, pg_create_physical_replication_slot_extern, creditcardmasking, basicemailmasking, fullemailmasking,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

	alldigitsmasking, shufflemasking, randommasking, regexpmasking, gs_digest, desencrypt, desdecrypt, es3encrypt, and des3decrypt functions.
	• Parameter passed using => is not supported for the projection alias to display the complete function. The projection alias cannot contain double quotation marks (""), for example, **select "power"(2,3)**.
	• To enable the projection alias to display the complete function, this function is not affected by parameters such as removing 0 at the end.
	• If a projection alias contains more than 63 bytes, the excess part will be truncated.
	• If the **args** parameter is a numeric constant, the projection alias cannot display leading zeros.
	`gaussdb=# SET behavior_compat_options = 'enable_funcname_with_argsname';`
	`SET`
	`gaussdb=# SELECT abs(0010);`
	`abs(10)`
	`---------`
	`10`
	`(1 row)`
	• In the following special scenarios, the projection alias is different:
	– Scenario 1: Some functions can be called as a system function or called by syntax. For details about these functions, see **Table 1-29**. In scenarios where functions can be called without parentheses, the pg_catalog prefix is omitted when the functions are called.
	– Scenario 2: When XMLType is called with the pg_catalog prefix or enclosed in double quotation marks, it is treated as a system function, and the projection alias can display the complete function. When only XMLType is used for calling, it is treated as a syntax function, and the projection alias does not display the complete function.
	`gaussdb=# \c ora_db`
	`ora_db=# SET behavior_compat_options = 'enable_funcname_with_argsname';`
	`SET`
	`-- Scenario 1 (the current user is omm):`
	`ora_db=# SELECT pg_catalog.current_user;`
	`current_user`
	`--------------`
	`omm`
	`(1 row)`
	
	`-- Scenario 2:`
	`ora_db=# SELECT pg_catalog.xmltype('<a>1</a>');`
	`pg_catalog.xmltype('<a>1</a>')`
	`--------------------------------`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

	<a>1
	(1 row)
	
	ora_db=# SELECT "xmltype"('<a>1');
	xmltype('<a>1')
	---------- 	
	<a>1
	(1 row)
	
	ora_db=# SELECT xmltype('<a>1');
	xmltype
	---------- 	
	<a>1
	(1 row)
	
	ora_db=# \c postgres
	gaussdb=# DROP DATABASE ora_db;

proc_outparam_transfer_length	After this parameter is enabled, the length of the **out** output parameter can be passed to the stored procedure and function, and an error will be reported in the inner stored procedure or function if any. For example:
	gaussdb=# SET behavior_compat_options='proc_outparam_override,proc_outparam_transfer_length';
	SET
	gaussdb=# CREATE OR REPLACE PROCEDURE out_param_test1(m in int, v inout varchar2,v1 inout varchar2) is
	gaussdb$# begin
	gaussdb$# v := 'aaaddd';
	gaussdb$# v1 := 'aaaddd';
	gaussdb$# end;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# CREATE OR REPLACE PROCEDURE call_out_param_test1 is
	gaussdb$# v varchar2(5) := 'aabbb';
	gaussdb$# v1 varchar2(6) := 'aabb';
	gaussdb$# begin
	gaussdb$# out_param_test1(5,v,v1);
	gaussdb$# end;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# CALL call_out_param_test1();
	ERROR: value too long for type character varying(5)
	CONTEXT: PL/SQL function out_param_test1(integer,character varying,character varying) line 3 at assignment
	PL/SQL function call_out_param_test1() line 4 at SQL statement
	
	gaussdb=# DROP PROCEDURE call_out_param_test1;
	gaussdb=# DROP PROCEDURE out_param_test1;

compat_cursor	Specifies the compatibility of error code variables. In this option is set in the ORA-compatible mode, error code variables (*SQLCODE* and *SQLSTATE*) are modified. In earlier versions, they are valid only for the currently executed function; while in this version, they are valid for the subfunctions called by this function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

tableof_elem_constraints	After this parameter is enabled:
	1. For the collection type, the validity of elements will be verified.
	2. If the index is of the varchar collection type, the validity of the index length is also verified.
	For details about behavior differences and examples, see **Collections**.
	NOTE
	• Do not switch and use this parameter in the same session.
	• After a database is upgraded:
	  • For collection types that have been defined in stored procedures or packages, enabling the **tableof_elem_constraints** parameter does not take effect. You need to rebuild the collection type. That is, you need to rebuild the stored procedure or package by performing the CREATE OR REPLACE or DROP+CREATE operation. The parameter takes effect only after the collection type is rebuilt.
	  • If a collection type is created using the syntax: CREATE TYPE type_name IS TABLE OF data_type, you need to drop the type and rebuild it for the function to take effect.
	The following is an example of verifying the validity of elements:
	If this parameter is not enabled, the elements are not verified when the following stored procedure is called:
	`gaussdb=# CREATE OR REPLACE procedure p1 is`
	`gaussdb$#  type t1 is table of varchar(5);`
	`gaussdb$#  v t1 := t1();`
	`gaussdb$# begin`
	`gaussdb$#  v.extend();`
	`gaussdb$#  v(1) := '123456';`
	`gaussdb$#  raise info '%', v;`
	`gaussdb$# end;`
	`gaussdb$# /`
	`CREATE PROCEDURE`
	`gaussdb=# CALL p1();`
	`INFO:  {123456}`
	` p1`
	`----`
	`(1 row)`
	`gaussdb=# DROP procedure p1;`
	If this parameter is enabled, the elements are verified and an error may be reported when the following stored procedure is called:
	`gaussdb=# SET behavior_compat_options = 'tableof_elem_constraints';`
	`SET`
	`gaussdb=# CREATE OR REPLACE procedure p1 is`
	`gaussdb$#  type t1 is table of varchar(5);`
	`gaussdb$#  v t1 := t1();`
	`gaussdb$# begin`
	`gaussdb$#  v.extend();`
	`gaussdb$#  v(1) := '123456';`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

	gaussdb$# raise info '%', v;
	gaussdb$# end;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# CALL p1();
	ERROR: value too long for type character varying(5)
	CONTEXT: PL/SQL function p1() line 5 at assignment
	The following is an example of verifying the index value:
	If this parameter is not enabled, no error is reported when the index value exceeds the defined length.
	gaussdb=# CREATE OR REPLACE procedure p1 is
	gaussdb$# type t1 is table of int index by varchar(5);
	gaussdb$# v t1;
	gaussdb$# begin
	gaussdb$# v('123456') := 1;
	gaussdb$# raise info '%', v;
	gaussdb$# end;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# CALL p1();
	INFO: {indexbyType:varchar,"123456"=>1}
	p1
	---- 	
	
	(1 row)
	
	gaussdb=# DROP procedure p1;
	If this parameter is enabled, an error is reported when the index value exceeds the defined length.
	gaussdb=# SET behavior_compat_options = 'tableof_elem_constraints';
	SET
	gaussdb=# CREATE OR REPLACE procedure p1 is
	gaussdb$# type t1 is table of int index by varchar(5);
	gaussdb$# v t1;
	gaussdb$# begin
	gaussdb$# v('123456') := 1;
	gaussdb$# raise info '%', v;
	gaussdb$# end;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# CALL p1();
	ERROR: value too long for type character varying(5)
	CONTEXT: PL/SQL function p1() line 4 at assignment

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
allow_rownum_alias	After this parameter is enabled, ROWNUM can be used as a column alias in SQL statements using the AS syntax. ROWNUM is used as a common identifier and cannot be used as a pseudocolumn. This parameter does not take effect in M-compatible databases. **NOTE** You are advised not to change the status of this parameter during service execution. When the parameter is enabled, database objects (such as table names, column names, and database names) are created using ROWNUM as the name in the database. If the parameter is disabled, ambiguity occurs and the behavior is unpredictable. When the parameter is disabled, the behavior of using ROWNUM as a pseudocolumn in the database becomes invalid after the parameter is enabled and the behavior is unpredictable.

current_sysdate	If this parameter is enabled, the current OS time is obtained when the **sysdate** command is executed. After the parameter is disabled, **sysdate** returns the system date and time when the current SQL statement is executed.
	`gaussdb=# set behavior_compat_options='current_sysdate';`
	`SET`
	`gaussdb=# select sysdate;`
	`current_sysdate`
	`---------------------`
	`2023-06-20 20:15:27`
	`(1 row)`
	After the GUC parameter **mapping_date_to_datea** is enabled, **sysdate** returns the datea type by default in ORA-compatible mode. In this case, **current_sysdate** does not take effect.
	`gaussdb=# set behavior_compat_options='';`
	`SET`
	`gaussdb=# show mapping_date_to_datea;`
	`mapping_date_to_datea`
	`----------------------`
	`on`
	`(1 row)`
	`gaussdb=# set NLS_DATE_FORMAT = 'YYYY-MM-DD HH24:MI:SS';`
	`SET`
	`gaussdb=# begin`
	`gaussdb$# dbe_output.print_line('time1:' || sysdate);`
	`gaussdb$# pg_sleep(2);`
	`gaussdb$# dbe_output.print_line('time2:' || sysdate);`
	`gaussdb$# end;`
	`gaussdb$# /`
	`time1:2025-02-27 10:45:57`
	`time2:2025-02-27 10:45:59`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
allow_function_procedure_replace	In the default scenario, for ORA and PG compatibility modes, when there is a stored procedure (or function) outside the package, you cannot create a function (or stored procedure) with the same name because they are of different object types, and an error is reported. After this parameter is enabled, you can use the CREATE OR REPLACE method to replace stored procedures and functions with the same names outside the package. gaussdb=# create or replace function proc_test return varchar2 as gaussdb$# begin gaussdb$# return '1'; gaussdb$# end; gaussdb$# / CREATE FUNCTION gaussdb=# create or replace procedure proc_test as gaussdb$# begin gaussdb$# null; gaussdb$# end; gaussdb$# / ERROR: cannot change routine kind DETAIL: "proc_test" is a function. gaussdb=# \df+ proc_test List of functions Schema | Name | Result data type | Argument data types | Type | Volatility | Owner | Language | Source code | Description | fencedmode | propackage | prokind --------+-----------+------------------+---------------------+---------+------------+----------+-------------+-------------+------------+--------- public | proc_test | character varying | | normal | volatile | wangxinyu | plpgsql | DECLARE +| | f | f | f | | | \

Configuration Item	Behavior
	gaussdb=# drop procedure proc_test;

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

collection_exception_backcompat	Exception value thrown when an error related to the collection type is reported in the PL/SQL. Currently, three exception values are controlled. The mapping is as follows:
	Parameter not enabled Parameter enabled
	collection_is_null program_limit_exceeded
	subscript_beyond_count program_limit_exceeded
	subscript_outside_limit program_limit_exceeded
	Example:
	gaussdb=# create or replace procedure p1 is
	gaussdb$# type t1 is table of int;
	gaussdb$# v t1;
	gaussdb$# v_int int;
	gaussdb$# begin
	gaussdb$# v_int := v.count();
	gaussdb$# exception when collection_is_null then
	gaussdb$# raise info '%', sqlerrm;
	gaussdb$# end;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# call p1();
	INFO: Reference to uninitialized collection
	p1
	---- 	
	
	(1 row)
	
	gaussdb=# create or replace procedure p1 is
	gaussdb$# type t1 is table of int;
	gaussdb$# v t1 := t1(1, 2, 3);
	gaussdb$# v_int int;
	gaussdb$# begin
	gaussdb$# v_int := v(4);
	gaussdb$# exception when subscript_beyond_count then
	gaussdb$# raise info '%', sqlerrm;
	gaussdb$# end;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# call p1();
	INFO: Subscript beyond count
	p1
	---- 	
	
	(1 row)
	
	gaussdb=# create or replace procedure p1 is
	gaussdb$# type t1 is table of int;
	gaussdb$# v t1 := t1(1, 2, 3);
	gaussdb$# v_int int;
	gaussdb$# begin
	gaussdb$# v_int := v(-1);
	gaussdb$# exception when subscript_outside_limit then
	gaussdb$# raise info '%', sqlerrm;
	gaussdb$# end;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# call p1();
	INFO: Subscript outside of limit
	p1
	---- 	
	
	(1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

	gaussdb=# set behavior_compat_options = 'collection_exception_backcompat';
	SET
	gaussdb=# create or replace procedure p1 is
	gaussdb$# type t1 is table of int;
	gaussdb$# v t1;
	gaussdb$# v_int int;
	gaussdb$# begin
	gaussdb$# v_int := v.count();
	gaussdb$# exception when program_limit_exceeded then
	gaussdb$# raise info '%', sqlerrm;
	gaussdb$# end;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# call p1();
	INFO: Reference to uninitialized collection
	 p1
	---- 	
	
	(1 row)
	
	gaussdb=# create or replace procedure p1 is
	gaussdb$# type t1 is table of int;
	gaussdb$# v t1 := t1(1, 2, 3);
	gaussdb$# v_int int;
	gaussdb$# begin
	gaussdb$# v_int := v(4);
	gaussdb$# exception when program_limit_exceeded then
	gaussdb$# raise info '%', sqlerrm;
	gaussdb$# end;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# call p1();
	INFO: Subscript beyond count
	 p1
	---- 	
	
	(1 row)
	
	gaussdb=# create or replace procedure p1 is
	gaussdb$# type t1 is table of int;
	gaussdb$# v t1 := t1(1, 2, 3);
	gaussdb$# v_int int;
	gaussdb$# begin
	gaussdb$# v_int := v(-1);
	gaussdb$# exception when program_limit_exceeded then
	gaussdb$# raise info '%', sqlerrm;
	gaussdb$# end;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# call p1();
	INFO: Subscript outside of limit
	 p1
	---- 	
	
	(1 row)
	
	gaussdb=# drop procedure p1;

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

enable_case_when_alias	If this parameter is enabled, the aliases of the CASE WHEN and DECODE syntaxes are character strings starting with __unnamed_.
	Example:
	gaussdb=# set behavior_compat_options='enable_case_when_alias';
	SET
	gaussdb=# create table test(c1 varchar2);
	CREATE TABLE
	gaussdb=# insert into test values('x');
	INSERT 0 1
	gaussdb=# select decode(c1,'x','0','default') from test;
	__unnamed_decode__
	-------------------- 	
	0
	(1 row)
	
	gaussdb=# select (case c1 when 'x' then '0' else 'default' end) from test;
	__unnamed_case_when__
	---------------------- 	
	0
	(1 row)
	
	gaussdb=# drop table test;

plsql_rollback_keep_user	Specifies whether ROLLBACK and ROLLBACK TO SAVEPOINT in PL/SQL change the current user, and whether SAVEPOINT in ROLLBACK TO a definer-privileged function changes the current user. If this parameter is enabled, ROLLBACK does not change the current user.
	Example:
	gaussdb=# create user plsql_rollback1 password '********';
	gaussdb=# create user plsql_rollback2 password '********';
	gaussdb=# grant plsql_rollback1 to plsql_rollback2;
	gaussdb=# create or replace procedure plsql_rollback1.p1 () authid definer
	gaussdb-# as
	gaussdb$# va int;
	gaussdb$# begin
	gaussdb$# raise info 'current usr:%', current_user;
	gaussdb$# rollback;
	gaussdb$# raise info 'current usr:%', current_user;
	gaussdb$# end;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# set session AUTHORIZATION plsql_rollback2 PASSWORD '********';
	SET
	gaussdb=> set behavior_compat_options = 'plsql_rollback_keep_user';
	SET
	gaussdb=> call plsql_rollback1.p1 ();
	INFO: current usr:plsql_rollback1
	INFO: current usr:plsql_rollback1
	p1
	---- 	
	
	(1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

enable_bpcharlikebpchar_compare	Enables or disables the bpcharlikebpchar and bpcharnlikebpchar operators.
	• This parameter is enabled by default for a newly installed database.
	• After the database of a version earlier than 505.1.0 is upgraded, this parameter is disabled by default.
	gaussdb=# SELECT bpcharlikebpchar('455'::BPCHAR(10), '455 '::BPCHAR);
	 bpcharlikebpchar
	------------------ 	
	 f
	(1 row)
	gaussdb=# SELECT bpcharlikebpchar('455 '::BPCHAR(10), '455 '::BPCHAR(10));
	 bpcharlikebpchar
	------------------ 	
	 t
	(1 row)
	gaussdb=# SELECT bpcharlikebpchar('455 '::BPCHAR(10), '455'::BPCHAR(10));
	 bpcharlikebpchar
	------------------ 	
	 t
	(1 row)
	gaussdb=# SELECT bpcharlikebpchar('455 '::BPCHAR(10), '455'::BPCHAR(11));
	 bpcharlikebpchar
	------------------ 	
	 f
	(1 row)
	gaussdb=# CREATE TABLE op_test (
	  col BPCHAR(2) DEFAULT NULL
	);
	CREATE TABLE
	gaussdb=# CREATE INDEX op_index ON op_test(col);
	CREATE INDEX
	gaussdb=# INSERT INTO op_test VALUES ('a');
	INSERT 0 1
	gaussdb=# INSERT INTO op_test VALUES ('1');
	INSERT 0 1
	gaussdb=# INSERT INTO op_test VALUES ('11');
	INSERT 0 1
	gaussdb=# INSERT INTO op_test VALUES ('12');
	INSERT 0 1
	gaussdb=# INSERT INTO op_test VALUES ('sd');
	INSERT 0 1
	gaussdb=# INSERT INTO op_test VALUES ('aa');
	INSERT 0 1
	gaussdb=# SHOW behavior_compat_options;
	 behavior_compat_options
	------------------------- 	
	(1 row)
	-- If **behavior_compat_options** does not contain **enable_bpcharlikebpchar_compare**, the latest bpcharlikebpchar operator is not enabled and the result set returned by the matching between bpchars is not the same as expected (all data should be returned in normal cases).
	gaussdb=# EXPLAIN (COSTS OFF) SELECT * FROM op_test WHERE col LIKE col::BPCHAR ORDER BY col;
	  QUERY PLAN
	-------------------------------------- 	
	 Sort
	  Sort Key: col
	  -> Seq Scan on op_test
	  Filter: (col ~~ (col)::text)
	(4 rows)
	gaussdb=# SELECT * FROM op_test WHERE col LIKE col::BPCHAR ORDER BY

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

	col;
	 col
	----- 	
	 11
	 12
	 aa
	 sd
	(4 rows)
	gaussdb=# SET behavior_compat_options =
	'enable_bpcharlikebpchar_compare';
	SET
	gaussdb=# SHOW behavior_compat_options;
	 behavior_compat_options
	--------------------------------- 	
	 enable_bpcharlikebpchar_compare
	(1 row)
	-- After this parameter is enabled, the latest bpcharlikebpchar operator is enabled, and the returned behavior meets the expected behavior during matching.
	gaussdb=# EXPLAIN (COSTS OFF) SELECT * FROM op_test WHERE col LIKE col::BPCHAR ORDER BY col;
	 QUERY PLAN
	------------------------------ 	
	 Sort
	 Sort Key: col
	 -> Seq Scan on op_test
	 Filter: (col ~~ col)
	(4 rows)
	gaussdb=# SELECT * FROM op_test WHERE col LIKE col::BPCHAR ORDER BY col;
	 col
	----- 	
	 1
	 11
	 12
	 a
	 aa
	 sd
	(6 rows)
	gaussdb=# DROP TABLE op_test;
	DROP TABLE
	NOTE
	• This parameter takes effect only when **sql_compatibility** is set to **ORA**.
	• If this parameter is enabled, the result set and execution plan for LIKE or NOT LIKE pattern matching between BPCHAR types are affected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

cursor_asensitive	If this parameter is enabled, the data sensitive behavior of cursors is compatible with versions earlier than 505.1. This parameter is disabled by default, indicating that cursors are data-insensitive in the database.
	• If this parameter is set, the data sensitive behavior of cursors is compatible with versions earlier than 505.1. Modification after some cursors are opened is displayed.
	gaussdb=# set behavior_compat_options='cursor_asensitive';
	SET
	gaussdb=# create table t1(c1 int, c2 varchar,c3 varchar) distribute by hash(c1);
	CREATE TABLE
	gaussdb=# insert into t1 values(1,'Tom','Jones');
	INSERT 0 1
	gaussdb=# create or replace function func1() returns sys_refcursor
	gaussdb-# as $$
	gaussdb$# declare
	gaussdb$# x sys_refcursor;
	gaussdb$# cursor cur1 for select count(1) from t1 where c2 = 'Joe';
	gaussdb$# begin
	gaussdb$# open cur1;
	gaussdb$# update t1 set c2='Joe' where c3='Jones';
	gaussdb$# fetch cur1 into x;
	gaussdb$# close cur1;
	gaussdb$# return x;
	gaussdb$# end;$$
	gaussdb-# LANGUAGE plpgsql;
	CREATE FUNCTION
	gaussdb=# call func1();
	func1
	------- 	
	1
	(1 row)
	
	gaussdb=# drop function func1;
	gaussdb=# drop table t1;
	• If this parameter is not set, the result sets of the cursors are determined when they are opened and the modification after they are opened is not displayed.
	gaussdb=# show behavior_compat_options;
	behavior_compat_options
	------------------------ 	
	
	(1 row)
	gaussdb=# create table t1(c1 int, c2 varchar,c3 varchar) distribute by hash(c1);
	CREATE TABLE
	gaussdb=# insert into t1 values(1,'Tom','Jones');
	INSERT 0 1
	gaussdb=# create or replace function func1() returns sys_refcursor
	gaussdb-# as $$
	gaussdb$# declare
	gaussdb$# x sys_refcursor;
	gaussdb$# cursor cur1 for select count(1) from t1 where c2 = 'Joe';
	gaussdb$# begin
	gaussdb$# open cur1;
	gaussdb$# update t1 set c2='Joe' where c3='Jones';
	gaussdb$# fetch cur1 into x;
	gaussdb$# close cur1;
	gaussdb$# return x;
	gaussdb$# end;$$

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

	gaussdb-# LANGUAGE plpgsql;
	CREATE FUNCTION
	gaussdb=# call func1();
	func1
	------- 	
	0
	(1 row)
	
	gaussdb=# drop function func1;
	gaussdb=# drop table t1;

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

enable_crosstype_integer_operator	Enables or disables the cross-type integer operator.
	• This parameter is enabled by default for a newly installed database.
	• After the database of a version earlier than 505.1.0 is upgraded, this parameter is disabled by default.
	• Involved operators: =, <>, <, >, <=, and >=
	• After this parameter is enabled, the involved cross-type integers can be compared directly without implicit conversion. Here is an example that uses (int1 op int2):
	ora_db=# SET max_datanode_for_plan=1;
	SET
	ora_db=# DROP TABLE IF EXISTS implicit_index;
	DROP TABLE
	ora_db=# CREATE TABLE implicit_index(c1 int1);
	NOTICE: The 'DISTRIBUTE BY' clause is not specified. Using 'c1' as the distribution column by default.
	HINT: Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
	CREATE TABLE
	ora_db=# SET behavior_compat_options='enable_crosstype_integer_operator';
	SET
	ora_db=# EXPLAIN (COSTS OFF) SELECT * FROM implicit_index WHERE c1 > 1::int2;
	QUERY PLAN
	---------- 	
	Data Node Scan
	Node/s: All datanodes
	
	Remote SQL: SELECT c1 FROM public.implicit_index WHERE c1 > 1::smallint
	Datanode Name: datanode1-- The node information is about the execution plan node of the current test instance, which may be different from the actual information. The actual execution information is subject to the actual information.
	Seq Scan on implicit_index
	Filter: (c1 > 1::smallint)
	
	(8 rows)
	
	ora_db=# SET behavior_compat_options='';
	SET
	ora_db=# EXPLAIN (COSTS OFF) SELECT * FROM implicit_index WHERE c1 > 1::int2;
	QUERY PLAN
	---------- 	
	Data Node Scan
	Node/s: All datanodes
	
	Remote SQL: SELECT c1 FROM public.implicit_index WHERE c1::bigint > 1::smallint
	Datanode Name: datanode1-- The node information is about the execution plan node of the current test instance, which may be different from the actual information. The actual execution information is subject to the actual information.
	Seq Scan on implicit_index
	Filter: ((c1)::bigint > 1::smallint)
	
	(8 rows)
	
	ora_db=# DROP TABLE implicit_index;
	DROP TABLE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

	NOTE
	• This parameter takes effect only when **sql_compatibility** is set to **ORA**.
	• If this parameter is enabled, the execution plan of operators involved in the following scenarios will be affected:
	  • (int1 op int2)
	  • (int1 op int4)
	  • (int1 op int8)
	  • (int1 op int16)
	  • (int1 op numeric)
	  • (int2 op int1)
	  • (int2 op int16)
	  • (int2 op numeric)
	  • (int4 op int1)
	  • (int4 op int16)
	  • (int4 op numeric)
	  • (int8 op int1)
	  • (int8 op int16)
	  • (int8 op numeric)
	  • (int16 op int1)
	  • (int16 op int2)
	  • (int16 op int4)
	  • (int16 op int8)
	  • (numeric op int1)
	  • (numeric op int2)
	  • (numeric op int4)
	  • (numeric op int8)

forbid_package_function_with_prefix	After this parameter is enabled, an error is reported if a function with a prefix is created in a package.
	For example:
	`gaussdb=# SET behavior_compat_options='forbid_package_function_with_prefix';`
	`SET`
	`gaussdb=# CREATE SCHEMA schema1;`
	`gaussdb=# CREATE OR REPLACE package pck1`
	`  IS`
	`  procedure schema1.proc1();`
	`  END pck1;`
	`  /`
	`ERROR:  not support procedure name use *.* format in package at or near ";"`
	`CONTEXT:  compilation of PL/pgSQL package "pck1" near line 2`
	
	`gaussdb=# DROP SCHEMA schema1;`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
time_constexpr_compact	If this parameter is enabled, in a scenario where the time expression is executed, the type automatically returned is with timezone or without timezone depending on whether the constant carries the time zone. Currently, the timestamp and time types are supported. -- Execute a timestamp expression without a time zone. gaussdb=# SELECT timestamp '1999-03-15 8:00:00'; timestamp --------------------- 1999-03-15 08:00:00 (1 row) -- The type timestamp with time zone is returned after a timestamp expression with a time zone is executed. gaussdb=# SELECT timestamp '1999-03-15 8:00:00 -8:00:00'; timestamptz ------------------------ 1999-03-16 00:00:00+08 (1 row)
enable_use_ora_timestamptz	After it is enabled, the date and time obtained by the TIMESTAMP AT TIME ZONE syntax are the same as those in ORA-compatible database. GaussDB can display only the session time zone, and the ORA-compatible database can display the entered time zone. If the time zones are different, the displayed results are different. After the date and time are converted to the same time zone, the same time point is displayed. -- Before enabling a parameter gaussdb=# SELECT timestamp '2024-03-20 01:30:00' at time zone 'Europe/Moscow' FROM dual; timezone ---------------------- 2024-03-20 06:30:00+08 (1 row) -- After enabling a parameter gaussdb=# SET behavior_compat_options='enable_use_ora_timestamptz'; SET gaussdb=# SELECT timestamp '2024-03-20 01:30:00' at time zone 'Europe/Moscow' FROM dual; timezone ---------------------- 2024-03-20 01:30:00+08 (1 row) **NOTE** This parameter is valid only in an ORA-compatible database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

forbid_skip_tableof_empty_str_elem	It is used to control the input behavior of the collection type without indexes for the character string in the format of **'{1,,3}'**.
	● After this parameter is enabled, the collection without an index does not support the input of character strings in the format of **'{1,,3}'**.
	gaussdb=# CREATE TYPE t1 IS TABLE OF int;
	CREATE TYPE
	gaussdb=# CREATE OR REPLACE PROCEDURE p1(v t1) IS
	gaussdb$# BEGIN
	gaussdb$# FOR i IN 1..v.count LOOP
	gaussdb$# dbe_output.print_line('v(' || i || ') is ' || v(i));
	gaussdb$# END LOOP;
	gaussdb$# END;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# SET behavior_compat_options =
	'forbid_skip_tableof_empty_str_elem';
	SET
	gaussdb=# CALL p1('{1,,3}');
	ERROR: Malformed nesttable literal: "{1,,3}".
	LINE 1: call p1('{1,,3}');
	^
	gaussdb=# CALL p1('{1,}');
	ERROR: Malformed nesttable literal: "{1, }".
	LINE 1: call p1('{1, }');
	^
	gaussdb=# CALL p1('{,3}');
	ERROR: Malformed nesttable literal: "{ ,3}".
	LINE 1: call p1('{ ,3}');
	^
	gaussdb=# DROP PROCEDURE p1;
	DROP PROCEDURE
	gaussdb=# DROP TYPE t1;
	DROP TYPE
	● After this parameter is disabled, the collection without an index supports the input of character strings in the format of **'{1,,3}'**. Empty strings are automatically filtered out during conversion.
	gaussdb=# CREATE TYPE t1 IS TABLE OF int;
	CREATE TYPE
	gaussdb=# CREATE OR REPLACE PROCEDURE p1(v t1) IS
	gaussdb$# BEGIN
	gaussdb$# FOR i IN 1..v.count LOOP
	gaussdb$# dbe_output.print_line('v(' || i || ') is ' || v(i));
	gaussdb$# END LOOP;
	gaussdb$# END;
	gaussdb$# /
	CREATE PROCEDURE
	gaussdb=# SET behavior_compat_options = '';
	SET
	gaussdb=# CALL p1('{1,,3}');
	v(1) is 1
	v(2) is 3
	p1
	---- 	
	
	(1 row)
	
	gaussdb=# CALL p1('{1,}');
	v(1) is 1

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

	p1
	---- 	
	
	(1 row)
	
	gaussdb=# CALL p1('{,3}');
	v(1) is 3
	p1
	---- 	
	
	(1 row)
	
	gaussdb=# DROP PROCEDURE p1;
	DROP PROCEDURE
	gaussdb=# DROP TYPE t1;
	DROP TYPE

sys_function_without_brackets	After this parameter is enabled, system functions without input parameters in ORA-compatible mode can be called without parentheses. gaussdb=# SET behavior_compat_options='sys_function_without_brackets'; gaussdb=# SELECT systimestamp; systimestamp ------------------------------- 2024-04-16 15:55:21.614342+08 (1 row)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
array_count_compat	Specifies whether the return value of the array_count function is **NULL** or **0** when the variable of the variable-length array type is **NULL**. • After the parameter is enabled, if the variable of the variable-length array type is **NULL**, the array_count function returns **0**. gaussdb=# create type t1 as table of int; CREATE TYPE gaussdb=# set behavior_compat_options='array_count_compat'; SET gaussdb=# gaussdb=# declare gaussdb-# type typ is varray(5) of int; gaussdb-# va typ; gaussdb-# begin gaussdb$# raise info 'va count:%',va.count; gaussdb$# if 1 > va.count then gaussdb$# raise info ' 1 > null is true'; gaussdb$# else gaussdb$# raise info ' 1 > null is false'; gaussdb$# end if; gaussdb$# end; gaussdb$# / INFO: va count:0 INFO: 1 > null is true ANONYMOUS BLOCK EXECUTE • After the parameter is disabled, if the variable of the variable-length array type is **NULL**, the array_count function returns **NULL**. gaussdb=# set behavior_compat_options=''; SET gaussdb=# create type t1 as table of int; CREATE TYPE gaussdb=# declare gaussdb-# type typ is varray(5) of int; gaussdb-# va typ; gaussdb-# begin gaussdb$# raise info 'va count:%',va.count; gaussdb$# if 1 > va.count then gaussdb$# raise info ' 1 > null is true'; gaussdb$# else gaussdb$# raise info ' 1 > null is false'; gaussdb$# end if; gaussdb$# end; gaussdb$# / INFO: va count:<NULL> INFO: 1 > null is false ANONYMOUS BLOCK EXECUTE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

search_path_behavior_optimization	Enables or disables the path search enhancement function for stored procedures.
	● This parameter is disabled by default for a newly installed database.
	● If this option is enabled, the capability of the **bind_procedure_searchpath** option is included. In a stored procedure, schemas in the session-level **search_path** setting are bound to form a schema list. If no schema is explicitly specified for an object in a stored procedure, the object is searched in the schema list in sequence. If both **bind_procedure_searchpath** and **search_path_behavior_optimization** are enabled, **search_path_behavior_optimization** is used by default.
	● The differences between enabling the option and only enabling the **bind_procedure_searchpath** option are that the refresh logic of *$user* in the search path compilation product changes and the pg_temp schema in the search path compilation product changes.
	● After this option is enabled, the behavior changes are as follows:
	– The SECURITY DEFINER function is called in nested mode to refresh the search path. In the compilation product, *$user* is the owner of the innermost nested function.
	– The namespace of the local temporary table is not added to the search path compilation product.
	– Stored procedures or functions cannot use local temporary tables *%type* and *%rowtype*. (This limitation does not apply to the scenario where local temporary tables *%type* and *%rowtype* are used as input and output parameters.) For example:
	`gaussdb=# SET behavior_compat_options='search_path_behavior_optimization';`
	`SET`
	
	`-- Create the local temporary table **temp_table**.`
	`gaussdb=# CREATE TEMP TABLE tmp_table(c1 int);`
	`CREATE TABLE`
	
	`-- An error is reported when the local temporary table *%rowtype* is used in the stored procedure.`
	`gaussdb=# CREATE or REPLACE PROCEDURE test_temp_tab_proc AS`
	`gaussdb$# a tmp_table%ROWTYPE;`
	`gaussdb$# BEGIN`
	`gaussdb$# NULL;`
	`gaussdb$# END;`
	`gaussdb$# /`
	`ERROR: UNSUPPORTED FEATURE with temporary table when setting behavior_compat_options: search_path_behavior_optimization.`
	`CONTEXT: compilation of PL/pgSQL function "test_temp_tab_proc" near line 2`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

enable_ora_joinop_in_updatestmt	Specifies whether the subquery supports the + operator when the SET clause in the UPDATE statement contains subqueries.
	After this parameter is enabled, the subquery returns a result with +.
	gaussdb=# CREATE DATABASE a_db dbcompatibility = 'ora';
	CREATE DATABASE
	gaussdb=# \c a_db
	a_db=# CREATE TABLE t1(a1 int, b int);
	NOTICE: The 'DISTRIBUTE BY' clause is not specified. Using 'a1' as the distribution column by default.
	HINT: Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
	CREATE TABLE
	a_db=# CREATE TABLE t2(c1 int, d int);
	NOTICE: The 'DISTRIBUTE BY' clause is not specified. Using 'c1' as the distribution column by default.
	HINT: Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
	CREATE TABLE
	a_db=# CREATE TABLE t3(a2 int, c2 int, e int);
	NOTICE: The 'DISTRIBUTE BY' clause is not specified. Using 'a2' as the distribution column by default.
	HINT: Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
	CREATE TABLE
	a_db=# INSERT INTO t1 VALUES(1, 11),(2, 12);
	INSERT 0 2
	a_db=# INSERT INTO t2 VALUES(3, 23),(4, 24);
	INSERT 0 2
	a_db=# INSERT INTO t3 VALUES(1, 3, 31),(5, 6, 32);
	INSERT 0 2
	a_db=# CREATE TABLE result(col1 int, col2 int, col3 int, col4 int, col5 int,col6 int, col7 int);
	NOTICE: The 'DISTRIBUTE BY' clause is not specified. Using 'col1' as the distribution column by default.
	HINT: Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
	CREATE TABLE
	a_db=# INSERT INTO result SELECT * FROM t1, t2, t3 WHERE t1.a1=t3.a2 AND t2.c1=t3.c2;
	INSERT 0 1
	a_db=# UPDATE result SET (col2, col3, col4 ,col5,col6,col7) = (SELECT b,c1,d,a2,c2,e FROM t1, t2, t3 WHERE t1.a1=t3.a2(+) AND t3.c2=t2.c1(+));
	ERROR: The subquery in the expression returns more than one row of result.
	CONTEXT: referenced column: col2

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

disable_insert_void_value	Specifies whether the return value of the void type can be used to execute INSERT or UPDATE. This parameter takes effect in Oracle-compatible mode.
	If this parameter is enabled, INSERT or UPDATE fails to be executed for the void type. If this parameter is disabled, INSERT or UPDATE can be executed.
	gaussdb=# SELECT pg_typeof(dbe_application_info.set_action(''::text));
	pg_typeof
	----------- 	
	void
	(1 row)
	gaussdb=# SET behavior_compat_options = 'disable_insert_void_value';
	gaussdb=# CREATE TABLE insert_void (id int,col text) DISTRIBUTE BY HASH(id);
	gaussdb=# INSERT INTO insert_void (id,col) VALUES(1,dbe_application_info.set_client_info(''::text));
	ERROR: column "col" is of type text but expression is of type void
	LINE 1: INSERT INTO insert_void (id,col) VALUES(1,dbe_application_in...
	^
	HINT: You will need to rewrite or cast the expression.
	CONTEXT: referenced column: col
	gaussdb=# UPDATE insert_void SET col = dbe_application_info.set_client_info(''::text);
	ERROR: column "col" is of type text but expression is of type void
	LINE 1: UPDATE insert_void SET col = dbe_application_info.set_clie...
	^
	HINT: You will need to rewrite or cast the expression.
	CONTEXT: referenced column: col

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior

ignore_like_default_escape	Specifies whether backslashes (\) in a pattern string are treated as literal characters during LIKE pattern matching.
	This parameter can be set only when **standard_conforming_strings** is set to **on**.
	• After this parameter is enabled, the backslash (\) in the pattern string can be treated as a literal character during LIKE pattern matching. The backslash (\) does not escape and cannot eliminate the special meaning of the wildcard.
	`gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA';`
	`CREATE DATABASE`
	`gaussdb=# \c ora_db`
	`ora_db=# SET standard_conforming_strings = on;`
	`SET`
	`ora_db=# SET behavior_compat_options = 'ignore_like_default_escape';`
	`SET`
	`ora_db=# CREATE TABLE test(c1 text) DISTRIBUTE BY HASH(c1);`
	`CREATE TABLE`
	`ora_db=# INSERT INTO test VALUES('aa\a');`
	`INSERT 0 1`
	`ora_db=# SELECT * FROM test WHERE c1 LIKE 'aa\a';`
	`  c1`
	`------`
	` aa\a`
	`(1 row)`
	
	`ora_db=# DROP TABLE test;`
	`DROP TABLE`
	`ora_db=# \c postgres`
	`gaussdb=# DROP DATABASE ora_db;`
	• After this parameter is disabled, the backslash (\) in the pattern string is used as an escape character to remove the special meaning of the wildcard.
	`gaussdb=# CREATE DATABASE ora_db DBCOMPATIBILITY = 'ORA';`
	`CREATE DATABASE`
	`gaussdb=# \c ora_db`
	`ora_db=# SET behavior_compat_options = '';`
	`SET`
	`ora_db=# CREATE TABLE test(c1 text) DISTRIBUTE BY HASH(c1);`
	`CREATE TABLE`
	`ora_db=# INSERT INTO test VALUES('aaa');`
	`INSERT 0 1`
	`ora_db=# SELECT * FROM test WHERE c1 LIKE 'aa\a';`
	`  c1`
	`------`
	` aaa`
	`(1 row)`
	
	`ora_db=# DROP TABLE test;`
	`DROP TABLE`
	`ora_db=# \c postgres`
	`gaussdb=# DROP DATABASE ora_db;`

forbid_update_multi_same_tables	It does not take effect in distributed scenarios.
disable_rewrite_nesttable	It does not take effect in distributed scenarios.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
aformat_null_test	It does not take effect in distributed scenarios.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Adjust the parameter value based on the database-compatible objects.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

Some options of this parameter affect the cache plan. If this parameter is modified in the prepared statement, the execution result does not change.

a_format_version

Parameter description: Specifies the compatibility configuration item of the database platform.

Parameter type: string

Unit: none

Value range: **Table 6-33** lists the supported compatibility configuration items.

Default value: **"10c"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

Table 6-33 Compatibility configuration item

Configuration Item	Behavior
10c	The ORA platform is compatible with the 10C version. **a_format_version** and **a_format_dev_version** are used together. For details about behavior control, see **Table 6-34**.

a_format_dev_version

Parameter description: Specifies the database platform minor version compatibility configuration item.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: string

Unit: none

Value range: **Table 6-34** lists the supported compatibility configuration items.

Default value: **"s7"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-34 Compatibility configuration items

Configuration Item	Behavior
s1	• A minor version of ORA-compatible iteration, which affects the following functions: round(date, fmt), nvl2, lpad, rpad, add_months, months_between, regexp_replace, regexp_count, empty_clob, instrb, trunc(numeric, int), greatest, least, mod, round(number, int), cast, to_date, to_timestamp, chr, to_char, to_number, and to_timestamp_tz. – round(date, fmt): This function, with input parameters of the TIMESTAMP WITHOUT TIME ZONE type, takes effect only when the value of **a_format_version** is **10c** and that of **a_format_dev_version** is **s1** in an ORA-compatible database. – nvl2: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode. – lpad: When **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode: – The **length** parameter indicates the display length of a character string. The display length of a single character is processed based on ORA-compatible requirements. – During the function execution, if the remaining length is 1 and the next character is of the full-width type (2 bytes), a space character is added to the left of the string. – If the value of **length** is a decimal, the value is truncated instead of being rounded off. – The **string** and **fill** parameters do not comply with the encoding specifications. – rpad: When **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode: – The **length** parameter indicates the display length of a character string. The display length of a single character is processed based on ORA-compatible requirements. – During the function execution, if the remaining length is 1 and the next character is of the full-width type (2 bytes), a space character is added to the right of the string. – If the value of **length** is a decimal, the value is truncated instead of being rounded off. – The **string** and **fill** parameters do not comply with the encoding specifications. – If the length of the **string** parameter exceeds 1 GB after padding, an error is reported. – add_months: When the GUC parameter **mapping_date_to_datea** is set to **off**, the database parses

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
	the date type in the input parameter into the TIMESTAMP(0) WITHOUT TIME ZONE type. In this case, the return value is of the timestamp type. When the input parameter type of the function is parsed as TIMESTAMP(0) WITHOUT TIME ZONE, the function is in Oracle-compatible mode, and **a_format_version** is set to **10c** and **a_format_dev_version** is set to **s1**: – If the calculation result is greater than 9999, an error will be reported. – If the input parameter **n** is a decimal, it will be truncated rather than rounded off. – months_between: When the GUC parameter **mapping_date_to_datea** is set to **off**, the database parses the date type in the input parameter to the TIMESTAMP(0) WITHOUT TIME ZONE type. This function takes effect in the scenario where this function input parameter is parsed to the TIMESTAMP(0) WITHOUT TIME ZONE type in Oracle-compatible mode, the value of **a_format_version** is **10c**, and that of **a_format_dev_version** is **s1**. – regexp_replace: When the function is in ORA-compatible mode, **a_format_version** is set to **10c**, and **a_format_dev_version** is set to **s1**, the default value of **occurrence** is **0**, indicating that all matched substrings are replaced. In addition, the **pattern** parameter that ends with a backslash (\) is valid. – regexp_count: When the function is in ORA-compatible mode, the value of **a_format_version** is **10c**, and the value of **a_format_dev_version** is **s1**, the **pattern** parameter ending with a backslash (\) is valid. – empty_clob: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode. – instrb: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode. – trunc(numeric, int): When **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode, if the input parameter s is a decimal, the parameter is truncated instead of being rounded off. – greatest: When **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode: – If any parameter is null, the function will return null. – If the first parameter is of the numeric type, the function converts this parameter and other parameters to the numeric type, and then compares them. If the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
	parameters cannot be converted, an error message is displayed. If the first parameter is of another type, the function converts other parameters to the type of the first parameter for comparison. If the parameters cannot be converted, an error message is displayed. – least: When **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode: – If any parameter is null, the function will return null. – If the first parameter is of the numeric type, the function converts this parameter and other parameters to the numeric type, and then compares them. If the parameters cannot be converted, an error message is displayed. If the first parameter is of another type, the function converts other parameters to the type of the first parameter for comparison. If the parameters cannot be converted, an error message is displayed. – mod: When **a_format_version** is **10c** and **a_format_dev_version** is **s6** in ORA-compatible mode, this function supports overloading with mod(text, int). When the mod function is called with (text, int) as the input type, mod(text, int) will take precedence. – round(numeric, int): – When the control parameter **s** is a decimal: In ORA-compatible databases where **a_format_version** is **10c** and **a_format_dev_version** is **s1**, the function truncates **s** to the integer. In other setups, it will round off **s** to the nearest integer. – If the value of **a_format_version** is **10c** and the value of **a_format_dev_version** is **s1**, the round function supports round(timestamp, text) overloading. When **(text, text)** or **(text, '')** is used as the input parameter to call the round function, round(timestamp, text) is preferred. – cast: When **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode, this function supports the optional syntax DEFAULT z ON CONVERSION ERROR and optional parameter **fmt**. – to_date: When **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode, if the year exceeds 9999, the system may not report an error. For example, executing **to_date('99999-12-12', 'yyyy-mm-dd hh24:mi:ss')** will yield **9999-09-12 12:00:00**. Any numbers that come after 9999 (the maximum allowed year) will be interpreted as the next **fmt**. – to_timestamp: When **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode, **fmt**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
	supports FF[7-9]. When FF[7-9] is applied, the number of characters that can be converted from the target position in the string is limited by the number following FF, with only 6 characters ultimately being retained in the conversion result. – chr: When **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode, the value is truncated instead of being rounded off if the value of *integer* is a decimal. – to_char(numeric/smallint/integer/bigint/double precision/ real[, fmt]): When **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode, the function supports the $, C, TM, TM9, TME, and U formats. However, it does not support TH, PL, or SG. – to_number: When **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible database, the supported formats are described in the to_number function description in *Developer Guide*. – to_timestamp_tz: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s1** in ORA-compatible mode. ● Data type casting: A decimal character string is rounded off when it is converted to an integer (int1/int2/int4/int8/int16). ● Data type conversion: Implicit conversions from timestamp with time zone to timestamp without time zone and datea, and from timestamp to datea are supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
s2	• Compatible minor version of platform ORA, which affects functions, such as dump, to_single_byte, to_multi_byte, nls_upper, nls_lower, initcap, ascii2, asciistr, unistr, vsize, cosh, remainder, sinh, tanh, nanvl, current_date, current_timestamp, dbtimezone, numtodsinterval, numtoyminterval, new_time, sessiontimezone, sys_extract_utc, tz_offset, to_binary_double, to_binary_float, to_dsinterval, to_yminterval, lnnvl, and ora_hash. – dump: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – to_single_byte: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – to_multi_byte: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – nls_upper: This function is supported when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – nls_lower: This function is supported when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – initcap: When **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode, if a case-insensitive character, such as Chinese, is followed by a case-sensitive character, only the first letter of the case-sensitive character is capitalized. Therefore, you are advised to set **a_format_version** to **10c** and **a_format_dev_version** to **s2**. – ascii2: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – asciistr: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – unistr: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – vsize: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – cosh: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
	– remainder: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – sinh: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – tanh: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – nanvl: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – current_date: When **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode, the return type of this function is timestamp. – current_timestamp: When **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode, the precision parameter supports integer values of the numeric type. Otherwise, only int input is supported. – dbtimezone: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – numtodsinterval: When **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode, an error is reported if **interval_unit** is set to **'DAY'** and **num** is set to a value greater than **1000000000**. – numtoyminterval: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – new_time: When the GUC parameter **mapping_date_to_datea** is set to **off**, the database parses the date type in the input parameter to the TIMESTAMP(0) WITHOUT TIME ZONE type. In this case, the return value is of the timestamp type. This function takes effect in the scenario where this function input parameter is parsed to the TIMESTAMP(0) WITHOUT TIME ZONE type in Oracle-compatible mode, the value of **a_format_version** is **10c**, and that of **a_format_dev_version** is **s2**. – sessiontimezone: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – sys_extract_utc: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
	– tz_offset: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – to_binary_double: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – to_binary_float: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – to_dsinterval: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – to_yminterval: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – lnnvl: This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. – ora_hash: This function takes effect only when **a_format_version** is **10c** and **a_format_dev_version** is **s2** in ORA-compatible mode. ● Supports all behaviors when the compatibility configuration item is set to **s1**.
s3	● Compatible minor version of platform ORA. If the parameter is enabled, nested calling of functions without parameters is supported. ● Supports all behaviors when the compatibility configuration item is set to **s2**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
s4	• Compatible minor version of platform ORA, which affects functions such as nchr(cvalue int|bigint), to_timestamp_tz, getclobval(xml), and getstringval(xml). – nchr(cvalue int|bigint): This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s4** in ORA-compatible mode. – to_timestamp_tz(string [DEFAULT return_value ON CONVERSION ERROR], fmt, nlsparam): This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s4** in ORA-compatible mode. – getclobval(xml): This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s4** in ORA-compatible mode. – getstringval(cml): This function takes effect when **a_format_version** is **10c** and **a_format_dev_version** is **s4** in ORA-compatible mode. • Supports all behaviors when the compatibility configuration item is set to **s3**.
s5	• Compatible minor version of the ORA platform. If the parameter is enabled, the composite type with the same name as a function is preferentially parsed. The sys_guid() function is affected. – This function takes effect only when **a_format_version** is **10c** and **a_format_dev_version** is **s5** in ORA-compatible mode. • Supports all behaviors when the compatibility configuration item is set to **s4**.
s6	• The ORA platform is compatible with the iteration minor version. The parameter is enabled, and the function mod(text, int) is affected. – mod(text, int): When **a_format_version** is **10c** and **a_format_dev_version** is **s6** in ORA-compatible mode, the mod function supports overloading with mod(text, int). When the mod function is called with (text, int) as the input type, mod(text, int) will take precedence. • Supports all behaviors when the compatibility configuration item is set to **s5**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
s7	• The ORA platform is compatible with the minor iteration version and this parameter is enabled. • When this parameter is enabled, the following behavior is compatible: 1. Supports all behaviors when the GUC parameter **behavior_compat_options** is set to '**display_leading_zero, end_month_calculate, plstmt_implicit_savepoint, truncate_numeric_tail_zero, char_coerce_compat, aformat_regexp_match, compat_cursor, plsql_security_definer, proc_outparam_override, dynamic_sql_compat, proc_outparam_transfer_length, tableof_elem_constraints, current_sysdate, plsql_rollback_keep_user, enable_bpcharlikebpchar_compare, show_full_error_lineno, enable_crosstype_integer_operator, forbid_package_function_with_prefix, enable_case_when_alias, time_constexpr_compact, enable_use_ora_timestamptz, forbid_skip_tableof_empty_str_elem, sys_function_without_brackets**'. 2. Supports all behaviors when the GUC parameter **a_format_date_timestamp** is set to **on**. • Supports all behaviors when the compatibility configuration item is set to **s6**. • If this parameter needs to be set in the scenario where the JDBC driver is used to connect to the database, the JDBC driver must be updated accordingly.

b_format_version

Parameter description: Specifies the database platform compatibility behavior configuration item that controls the compatibility with earlier versions in MySQL-compatible mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- **b_format_version** takes effect only when **sql_compatibility** is set to **MySQL**.
- If this parameter is set to a non-empty string, **b_format_behavior_compat_options** is set to **"ALL"** and **bytea_output** is set to **"escape"** simultaneously. If this parameter is set to an empty string, **b_format_behavior_compat_options** and **bytea_output** are set to the original values.
- This parameter must be set on both CNs and DNs.
- To ensure data consistency, after a table with a table collation or column collation is created, this parameter must be used together with **b_format_version='5.7'** and **b_format_dev_version='s2'**, and cannot be changed during the running of the production environment.

Parameter type: string

Unit: none

Value range: an empty string **""** or **"5.7"**

Default value: **""**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter together with **b_format_dev_version** immediately after the installation and deployment. You do not need to change the value unless necessary.

Risks and impacts of improper settings: The behaviors of some SQL statements are affected, causing compatibility issues.

b_format_dev_version

Parameter description: Specifies the database platform minor version compatibility configuration item.

NOTICE

- **b_format_dev_version** takes effect only when **sql_compatibility** is set to **MYSQL** and **b_format_version** is set to **"5.7"**.
- This parameter must be set on both CNs and DNs.
- To ensure data consistency, after a table with a table collation or column collation is created, this parameter must be used together with **b_format_version='5.7'** and **b_format_dev_version='s2'**, and cannot be changed during the running of the production environment.

Parameter type: string

Unit: none

Value range: an empty string **""** or a compatibility configuration item in **Table 6-35**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-35 Compatibility configuration items

Configuration Item	Behavior
s1	1. The following parameters are influenced: If this parameter is enabled, the **datetime** option in the **disable_keyword_options** parameter is removed, indicating that DATETIME is used as a non-reserved keyword. *datetime* can be used as a table name, column name, or alias, but cannot be used as a function name, stored procedure name, or parameter name of a function or stored procedure. 2. The following functions are influenced: curdate, from_days, date_format, str_to_date, current_date, datediff, timestampdiff, date_add, subtime, month, time_to_sec, to_days, to_seconds, dayname, monthname, convert_tz, sec_to_time, addtime, adddate, date_sub, timediff, last_day, weekday, from_unixtime, unix_timestamp, subdate, day, year, weekofyear, dayofmonth, dayofyear, week, yearweek, dayofweek, time_format, hour, minute, second, microsecond, quarter, utc_date, get_format, extract, makedate, period_add, timestamppadd, period_diff, utc_time, utc_timestamp, sysdate, current_timestamp, maketime, curtime, current_time, localtime, localtimestamp, now, lc_time_names, default_week_format, json_object, and substring. 3. The following types are influenced: a. Be compatible with the integer type tinyint with the value range changed to –128 to 127. b. Be compatible with the character string types char and varchar. The precision *n* of char(*n*) and varchar(*n*) is changed from the byte length to the character length. c. Be compatible with the text types tinytext, mediumtext, and longtext. These types are mapped to the text type after the parameter is enabled. d. Be compatible with binary types tinyblob, blob, mediumblob, and longblob. These types are mapped to the bytea type after the parameter is enabled. e. Be compatible with floating-point types double, float, and real. Among them, double is mapped to the float8 type, double(p, s) is mapped to the numeric type, float(p, s) is mapped to the numeric type, real is mapped to float8, and real(p, s) is mapped to the numeric type. f. Be compatible with high-precision types numeric, dec, and decimal. If the precision and scale are not specified, the default precision is changed to **10** and the scale is changed to **0**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
	g. Be compatible with time types datetime, timestamp, time, and date. Replace datetime[(*p*)] with a timestamp without time zone[(*p*)]. Replace timestamp[(p)] with a timestamp with time zone[(p)]. The input, output, range, and precision specifications of these types are changed. For details, see related sections in **Data Types**. h. Be compatible with integer types tinyint, smallint, int, and bigint. The display width and ZEROFILL attributes of these types take effect after the parameter is enabled. 4. The following syntax is influenced: The column constraint ON UPDATE update_expr can be specified. 5. The following operators are influenced: a. An error is reported when the || operator between bytea types is used. b. The REGEXP operator is supported. 6. The following GUC parameter is influenced: When the **bytea_output** parameter is set to **'escape'**, the processing logic changes. For details, see **bytea_output**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
s2	The compatibility behavior controlled by s1 is included. In addition, the following impacts are included: 1. The following parameters are influenced: a. When this option is enabled, **standard_conforming_strings** and **escape_string_warning** are set to **off**. b. The **collation_connection** and **character_set_connection** parameters take effect only when this option is enabled. 2. The following functions are affected: a. The like operator does not report an error when an escape character is at the end of the matching string. b. The priorities of the collation rules of the character set and collation are changed. For details, see **Rules for Combining Character Sets and Collations**. c. After the parameter is enabled, the collation of foreign keys cannot be inconsistent with that of columns. 3. The following syntax is influenced: a. The CREATE TABLE table_name LIKE source_table syntax is supported. b. The syntaxes CREATE TABLE table_name LIKE source_table and CREATE TABLE table_name (LIKE source_table) cannot specify **INCLUDING** and **EXCLUDING** options. By default, **INCLUDING ALL** is specified. c. The LOAD DATA syntax is supported. d. The syntaxes for changing table names, such as ALTER TABLE and RENAME TABLE, have impact in the following scenario: If the character string corresponding to the new table name starts with **"#MySQL50#"** and is followed by other characters, **"#MySQL50#"** will be ignored. If the old and new table names are the same, no error will be reported. e. The collate clause can be specified by set names.

Default value: **""**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter together with **b_format_version** immediately after the installation and deployment. You do not need to change the value unless necessary.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: The behaviors of some SQL statements are affected, causing compatibility issues.

m_format_dev_version

Parameter description: Specifies the database platform minor version compatibility configuration item.

NOTE

m_format_dev_version takes effect only when **sql_compatibility** is set to **M**.

Parameter type: string

Unit: none

Value range: compatibility configuration items in **Table 6-36**. Currently, you can set the value to **s1** only.

Table 6-36 Compatibility configuration items

Configuration Item	Behavior
s1	The following syntax is influenced: 1. Disable the SELECT FETCH FIRST ROWS ONLY syntax. 2. The TRUNCATE CASCADE/RESTRICT syntax is disabled. 3. USING INDEX TABLESPACE tablespace_name syntax cannot be specified when a primary key is defined using CREATE TABLE. 4. The semantics of CASCADE/RESTRICT syntax in DROP TABLE/VIEW/COLUMN is ignored, and the default behavior is RESTRICT. 5. If the CREATE TABLE REFERENCES syntax is used as a column constraint, its semantics is changed to ignore the foreign key definition.

Default value: **"s1"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter immediately after the installation and deployment. You do not need to change the value unless necessary.

Risks and impacts of improper settings: The behaviors of some SQL statements are affected, causing compatibility issues.

sql_mode

Parameter description: Specifies the SQL behavior control configuration item in MySQL-compatible and M-compatible modes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- **sql_mode** takes effect when **sql_compatibility** is set to **MYSQL**, **b_format_version** is set to **'5.7'**, and **b_format_dev_version** is set to **'s1'**, or when **sql_mode** takes effect only when **sql_compatibility** is set to **M**.
- When **SHOW sql_mode** and **SELECT @@sql_mode** are used for query, the values are displayed in uppercase based on the sequence as follows: **REAL_AS_FLOAT**, **ANSI_QUOTES**, **ONLY_FULL_GROUP_BY**, **NO_UNSIGNED_SUBTRACTION**, **NO_AUTO_VALUE_ON_ZERO**, **NO_BACKSLASH_ESCAPES**, **STRICT_TRANS_TABLES**, **STRICT_ALL_TABLES**, **NO_ZERO_IN_DATE**, **NO_ZERO_DATE**, **ALLOW_INVALID_DATES**, **ERROR_FOR_DIVISION_BY_ZERO**, **TRADITIONAL**, **NO_AUTO_CREATE_USER**, **NO_ENGINE_SUBSTITUTION**, and **PAD_CHAR_TO_FULL_LENGTH**.

Parameter type: string

Unit: none

Value range:

- MySQL-compatible mode: an empty string **""** or a value in the range of **'strict_trans_tables,only_full_group_by,no_zero_in_date,no_zero_date,error_for_division_by_zero,no_auto_create_user,no_engine_substitution,pad_char_to_full_length,no_auto_value_on_zero'**.
- M-compatible mode: parameters supported in MySQL-compatible mode, **no_unsigned_subtraction**, **ansi_quotes**, **allow_invalid_dates**, **real_as_float**, **strict_all_tables**, and **traditional**.

The meaning of each option value is described in **Table 6-37**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-37 Compatibility configuration items

Configuration Item	Behavior	MySQL-compatible Parameter Configuration	M-compatible Parameter Configuration
strict_trans_tables	Currently, only data types and system functions in MySQL-compatible and M-compatible databases can be verified. If this parameter is set, its format and range are strictly verified. If an invalid value is entered or the value exceeds the range, an error is reported during parsing. If this parameter is not set, the format and range of the input parameter are loosely verified. If an invalid value is entered or the value exceeds the range, a warning is reported during parsing and the value after loose processing is returned. Troubleshooting: 1. Abnormal data When a string, rather than a numeric value, is inserted, the string is truncated from the first non-digit character, for example, **'1a'** is truncated to **1**. If the target type is a time type, the zero value of the corresponding data is returned. 2. Exceeding the maximum range The maximum or minimum value of the corresponding data is returned.	It can be set and canceled independently.	It can be set and canceled independently.
only_full_group_by	Projection columns that contain non-GROUP BY keys and that are not constants and aggregate functions are not allowed.	It can be set and canceled independently.	It can be set and canceled independently.
no_zero_in_date	The year, month, and day of the DATE cannot be **0**.	It can be set and canceled independently, but it does not take effect.	It can be set and canceled independently.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior	MySQL-compatible Parameter Configuration	M-compatible Parameter Configuration
no_zero_date	The value of DATE cannot be **0** (**0000-00-00**).	It can be set and canceled independently, but it does not take effect.	It can be set and canceled independently.
error_for_division_by_zero	The value cannot be divided by 0.	It can be set and canceled independently, but it does not take effect.	It can be set and canceled independently.
no_auto_create_user	This item has no actual function. It is used only for compatibility that an error is not reported when the SET SQL_MODE statement contains this option.	It can be set and canceled independently.	It can be set and canceled independently.
no_engine_substitution	This item has no actual function. It is used only for compatibility that an error is not reported when the SET SQL_MODE statement contains this option.	It can be set and canceled independently.	It can be set and canceled independently.
pad_char_to_full_length	Be used for formatted output of CHAR columns in a table. If there are CHAR columns, a string with spaces at the end is output. Otherwise, a string without spaces at the end is output.	It can be set and canceled independently.	It can be set and canceled independently.
no_auto_value_on_zero	If this option is included, the value **0** inserted into the **AUTO_INCREMENT** column does not trigger auto-increment.	It can be set and canceled independently.	It can be set and canceled independently.
no_unsigned_subtraction	Unsigned numbers cannot be subtracted. If unsigned integers are subtracted and the result is negative, an error is returned.	It can be set and canceled independently, but it does not take effect.	It can be set and canceled independently.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior	MySQL-compatible Parameter Configuration	M-compatible Parameter Configuration
ansi_quotes	The ANSI_QUOTES mode is enabled, in which double quotation marks are treated as identifier quotes, not string quotes.	It can be set and canceled independently, but it does not take effect. Double quotation marks ("") are always considered as identifier reference symbols.	It can be set and canceled independently.
allow_invalid_dates	If this option is enabled, MySQL allows invalid dates to be inserted, for example, '0000-00-00'.	It can be set and canceled independently, but it does not take effect.	It can be set and canceled independently.
real_as_float	By default, the REAL type is regarded as the DOUBLE type. If this option is enabled, the REAL type is regarded as the FLOAT type.	It can be set and canceled independently, but it does not take effect.	It can be set and canceled independently.
strict_all_tables	The effect is the same as that of **strict_trans_tables**.	It can be set and canceled independently.	It can be set and canceled independently.
traditional	The effect is the same as that of setting **strict_trans_tables**, **strict_all_tables**, **no_zero_in_date**, **no_zero_date**, **error_for_division_by_zero**, **no_auto_create_user**, and **no_engine_substitution** together.	It can be set and canceled independently. **NOTE** Whether this parameter takes effect depends on whether the MySQL-compatible mode is supported.	It can be set and canceled independently.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **"strict_trans_tables,only_full_group_by,no_zero_in_date,no_zero_date,error_for_division_by_zero,no_auto_create_user,no_engine_substitution"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

auto_increment_increment

Parameter description: Specifies the auto-increment step of an auto-increment column. The auto-increment value is calculated by the following formula: **auto_increment_offset** + *N* × **auto_increment_increment**. *N* is a positive integer.

Parameter type: integer.

Unit: none

Value range: 1 to 65535

Default value: **1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter as required.

Risks and impacts of improper settings: The self-increment value may not meet the expectation. If the parameter value is smaller than that of **auto_increment_offset**, an error occurs when the values in the auto-increment column automatically increase.

auto_increment_offset

Parameter description: Specifies the initial value of an auto-increment column. The auto-increment value is calculated by the following formula: **auto_increment_offset** + *N* × **auto_increment_increment**. *N* is a positive integer.

Parameter type: integer.

Unit: none

Value range: 1 to 65535

Default value: **1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter as required.

Risks and impacts of improper settings: The self-increment value may not meet the expectation. If the parameter value is greater than that of **auto_increment_increment**, an error occurs when the values in the auto-increment column automatically increase.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

auto_increment_cache

Parameter description: Specifies the number of reserved auto-increment cache values when auto-increment is triggered by batch insertion or import of auto-increment columns. When auto-increment values are reserved, the auto-increment counter value is updated to the maximum auto-increment cache value. Before the cache values are used up, the auto-increment counter value remains unchanged, and the triggered auto-increment uses the cache values.

NOTE

- The reserved cache values are valid only in the statement. If the reserved auto-increment cache values are used up and subsequent INSERT statements trigger auto-increment based on the auto-increment counter, the values in the auto-increment column in the table are discontinuous.
- When auto-increment is triggered by parallel import or insertion of auto-increment columns, the cache value reserved for each parallel thread is used only in the thread. If the cache value is not used up, the values of auto-increment columns in the table are discontinuous.
- When you add an auto-increment column to a table with data or modify a column to an auto-increment column, the existing data triggers auto-increment. The reserved auto-increment cache value is also affected by this parameter.
- This parameter does not affect the auto-increment column in the local temporary table.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

- If this parameter is set to **0**, the auto-increment cache values are automatically reserved.
  - When auto-increment is triggered for the first time, if the number of rows to be inserted into the auto-increment column is known, the number is the reserved value.
    - For example, the number of rows to be inserted cannot be obtained as the value of auto-increment that is triggered by INSERT INTO table SELECT ... or COPY FROM. When the ALTER TABLE statement is executed to rewrite table data, if auto-increment is triggered, reltuples in the statistics is used as the number of rows to be reserved. INSERT INTO table VALUES(...),(...),... is distributed to different DNs. Therefore, in some execution plans, DNs cannot obtain the number of rows to be inserted.
  - If the number of rows is unknown, 2^n values are reserved each time. For example, one value is reserved in the first auto-increment, two values are reserved in the second auto-increment, four values are reserved in the third auto-increment, and eight values are reserved for in fourth auto-increment. However, if the number of reserved values exceeds 65535, 65535 values are reserved.
- If this parameter is not set to **0**, the number of reserved cache values is the value of this parameter.
  - When auto-increment is triggered for the first time, if the number of rows to be inserted into the auto-increment column is known, the number is the reserved value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

  - If the number of rows is unknown, the value of **auto_increment_cache** is the number of auto-increment values reserved each time.

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: The default value is recommended. However, if the auto-increment values are discontinuous in this case, you can adjust the parameter value based on the amount of data to be inserted in batches.

Risks and impacts of improper settings: A larger parameter value indicates better batch insertion performance and more likely to generate discontinuous self-increment values.

auto_increment_mode

Parameter description: When creating a table in an M-compatible database, if **LOCAL** and **GLOBAL** are not specified in the **AUTO_INCREMENT** column, you can use the **auto_increment_mode** parameter to specify the column as a global or local auto-increment column.

Parameter type: string

Unit: none

Value range:

- **'global'**: global auto-increment column.
- **'local'**: local auto-increment column.

Default value: **'global'**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter as required.

Risks and impacts of improper settings: The self-increment value may not meet the expectation.

disable_keyword_options

Parameter description: Specifies database compatibility behavior. Multiple items are separated by commas (,). An identifier with this parameter set will not be used as a keyword.

Parameter type: string

Unit: none

Value range: The keywords vary depending on compatibility modes of databases. For details, see **Table 6-38**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-38 Keywords of the disable_keyword_options option that can be set for databases in different compatible modes

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
absolute	Supported	Not supported	Not supported
access	Supported	Not supported	Supported
account	Supported	Not supported	Not supported
action	Supported	Not supported	Not supported
active	Not supported	Not supported	Supported
add	Supported	Not supported	Not supported
adddate	Not supported	Supported	Not supported
admin	Not supported	Not supported	Supported
advanced	Supported	Supported	Not supported
algorithm	Supported	Not supported	Not supported
always	Supported	Not supported	Not supported
app	Supported	Not supported	Not supported
append	Supported	Not supported	Not supported
archive	Supported	Not supported	Not supported
array	Not supported	Not supported	Supported
assertion	Supported	Not supported	Not supported
assignment	Supported	Not supported	Not supported
async	Supported	Not supported	Not supported
at	Supported	Not supported	Not supported
audit	Supported	Not supported	Not supported
authentication	Not supported	Not supported	Supported
authid	Not supported	Not supported	Supported
auto	Supported	Supported	Not supported
auto_increment	Supported	Supported	Not supported
autoextend	Supported	Not supported	Not supported
autohint	Supported	Supported	Supported
automapped	Supported	Not supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
automatic	Supported	Supported	Not supported
backward	Supported	Not supported	Not supported
bad_path	Supported	Not supported	Not supported
barrier	Supported	Not supported	Not supported
before	Supported	Not supported	Not supported
begin_non_anoyblock	Supported	Not supported	Not supported
blanks	Supported	Not supported	Not supported
blockchain	Supported	Not supported	Not supported
body	Supported	Not supported	Not supported
buckets	Not supported	Not supported	Supported
bulk	Not supported	Not supported	Supported
byte	Supported	Supported	Not supported
byteawithoutorder	Supported	Not supported	Not supported
byteawithoutorderwithequal	Supported	Not supported	Not supported
cache	Supported	Not supported	Not supported
called	Supported	Not supported	Supported
cancelable	Supported	Not supported	Supported
catalog	Supported	Not supported	Not supported
chain	Supported	Not supported	Not supported
challenge_response	Not supported	Not supported	Supported
change	Supported	Supported	Not supported
characteristics	Supported	Not supported	Not supported
charset	Supported	Supported	Not supported
class	Supported	Not supported	Not supported
clean	Supported	Not supported	Not supported
clone	Not supported	Not supported	Supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
client	Supported	Not supported	Not supported
client_master_key	Supported	Not supported	Not supported
client_master_keys	Supported	Not supported	Not supported
columns	Supported	Supported	Not supported
column_encryption_key	Supported	Not supported	Not supported
column_encryption_keys	Supported	Not supported	Not supported
colview	Supported	Supported	Not supported
comment	Supported	Not supported	Not supported
comments	Supported	Not supported	Not supported
compact	Supported	Not supported	Not supported
compatible_illegal_chars	Supported	Not supported	Not supported
compile	Supported	Supported	Not supported
complete	Supported	Not supported	Not supported
completion	Supported	Supported	Not supported
component	Not supported	Not supported	Supported
concurrently	Supported	Not supported	Not supported
condition	Supported	Not supported	Not supported
configuration	Supported	Not supported	Not supported
conflict	Not supported	Supported	Not supported
connection	Supported	Not supported	Not supported
constant	Supported	Not supported	Not supported
containing	Supported	Supported	Not supported
content	Supported	Not supported	Supported
conversion	Supported	Not supported	Not supported
convert	Not supported	Supported	Not supported
coordinator	Supported	Not supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
coordinators	Supported	Not supported	Not supported
copy_custom_id	Supported	Supported	Not supported
cost	Supported	Not supported	Not supported
crossbucket	Supported	Supported	Not supported
csn	Supported	Supported	Not supported
csv	Supported	Not supported	Not supported
cube	Supported	Not supported	Not supported
current	Supported	Not supported	Not supported
current_schema	Supported	Not supported	Not supported
cume_dist	Not supported	Not supported	Supported
cycle	Supported	Not supported	Not supported
data	Supported	Not supported	Not supported
datatype_cl	Supported	Not supported	Not supported
datafiles	Supported	Supported	Not supported
date_add	Not supported	Supported	Not supported
date_format	Supported	Not supported	Not supported
date_sub	Not supported	Supported	Not supported
datetime	Supported	Supported	Not supported
day	Supported	Not supported	Not supported
day_hour	Supported	Supported	Not supported
day_microsecond	Supported	Supported	Not supported
day_minute	Supported	Supported	Not supported
day_second	Supported	Supported	Not supported
days	Supported	Supported	Not supported
db4aishot	Supported	Supported	Not supported
dbcompatibility	Supported	Not supported	Not supported
dbtimezone	Supported	Supported	Not supported
deallocate	Supported	Not supported	Not supported
decode	Supported	Not supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
deferred	Supported	Not supported	Not supported
definer	Supported	Not supported	Not supported
definition	Not supported	Not supported	Supported
delete_all	Supported	Supported	Not supported
delimiter	Supported	Not supported	Not supported
delimiters	Supported	Not supported	Not supported
delta	Supported	Not supported	Not supported
deltamerge	Supported	Not supported	Not supported
dense_rank	Supported	Supported	Supported
description	Not supported	Not supported	Supported
deterministic	Supported	Not supported	Not supported
dictionary	Supported	Not supported	Not supported
direct	Supported	Not supported	Not supported
directory	Supported	Not supported	Not supported
disable_all	Supported	Supported	Not supported
discard	Supported	Not supported	Not supported
discard_path	Supported	Supported	Not supported
disconnect	Supported	Not supported	Not supported
distribute	Supported	Not supported	Not supported
distributed	Supported	Supported	Not supported
distribution	Supported	Not supported	Not supported
document	Not supported	Not supported	Supported
dumpfile	Supported	Supported	Not supported
duplicate	Supported	Not supported	Not supported
each	Supported	Not supported	Not supported
elastic	Supported	Not supported	Not supported
empty	Supported	Supported	Supported
enable_all	Supported	Supported	Not supported
enclosed	Supported	Not supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
encoding	Supported	Not supported	Supported
encrypted	Supported	Not supported	Not supported
encrypted_value	Supported	Not supported	Not supported
encryption	Supported	Not supported	Not supported
encryption_type	Supported	Not supported	Not supported
ends	Supported	Supported	Not supported
enforced	Supported	Not supported	Supported
engine	Supported	Supported	Not supported
engine_attribute	Not supported	Not supported	Supported
entityescaping	Supported	Supported	Not supported
eol	Supported	Not supported	Not supported
errors	Supported	Not supported	Not supported
escaped	Supported	Supported	Not supported
escaping	Supported	Not supported	Not supported
evalname	Supported	Supported	Not supported
event	Supported	Supported	Not supported
events	Supported	Supported	Not supported
every	Supported	Not supported	Not supported
exchange	Supported	Not supported	Not supported
except	Not supported	Not supported	Supported
exclude	Supported	Not supported	Supported
excluding	Supported	Not supported	Not supported
execute	Supported	Not supported	Not supported
exclusive	Not supported	Not supported	Supported
expdp	Supported	Supported	Not supported
expire	Supported	Supported	Not supported
expired	Supported	Not supported	Not supported
extend	Supported	Supported	Not supported
external	Supported	Not supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
family	Supported	Not supported	Not supported
fast	Supported	Not supported	Not supported
features	Supported	Not supported	Not supported
factor	Not supported	Not supported	Supported
failed_login_attempts	Not supported	Not supported	Supported
fenced	Not supported	Not supported	Supported
fields	Supported	Not supported	Not supported
fileheader	Supported	Not supported	Not supported
fill_missing_fields	Supported	Not supported	Not supported
filler	Supported	Not supported	Not supported
filter	Supported	Not supported	Not supported
finish	Supported	Supported	Supported
first	Supported	Not supported	Not supported
first_value	Not supported	Not supported	Supported
fixed	Supported	Not supported	Not supported
following	Supported	Not supported	Supported
force	Not supported	Not supported	Supported
formatter	Supported	Not supported	Not supported
forward	Supported	Not supported	Not supported
freeze	Supported	Not supported	Not supported
functions	Supported	Not supported	Not supported
generated	Supported	Not supported	Not supported
generate	Not supported	Not supported	Supported
geomcollection	Not supported	Not supported	Supported
get_format	Not supported	Supported	Not supported
get_master_public_key	Not supported	Not supported	Supported
get_source_public_key	Not supported	Not supported	Supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
granted	Supported	Not supported	Not supported
greatest	Supported	Not supported	Not supported
grouping	Supported	Not supported	Supported
groups	Not supported	Not supported	Supported
gsidead	Supported	Supported	Not supported
gsigetxid	Supported	Supported	Not supported
gsimerge	Supported	Supported	Not supported
gsiready	Supported	Supported	Not supported
gsiswitch	Supported	Supported	Not supported
gsiusable	Supported	Supported	Not supported
gsivalid	Supported	Supported	Not supported
gsiwaitall	Supported	Supported	Not supported
gtid_only	Not supported	Not supported	Supported
hdfsdirectory	Supported	Not supported	Not supported
header	Supported	Not supported	Not supported
high	Supported	Supported	Not supported
hold	Supported	Not supported	Not supported
histogram	Not supported	Not supported	Supported
history	Not supported	Not supported	Supported
hour	Supported	Not supported	Not supported
hour_microsecond	Supported	Supported	Not supported
hour_minute	Supported	Supported	Not supported
hour_second	Supported	Supported	Not supported
hours	Supported	Supported	Not supported
identity	Supported	Not supported	Not supported
ifnull	Not supported	Supported	Not supported
ignore	Supported	Supported	Not supported
ignore_extra_data	Supported	Not supported	Not supported
ilm	Supported	Supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
ilm_pidx_list	Supported	Supported	Not supported
imcvtable	Supported	Supported	Not supported
immediate	Supported	Not supported	Not supported
immutable	Supported	Not supported	Not supported
impdp	Supported	Supported	Not supported
implicit	Supported	Not supported	Not supported
inactive	Not supported	Not supported	Supported
include	Supported	Not supported	Not supported
including	Supported	Not supported	Not supported
increment	Supported	Not supported	Not supported
incremental	Supported	Not supported	Not supported
indexes	Supported	Not supported	Not supported
infile	Supported	Not supported	Not supported
inherit	Supported	Not supported	Not supported
inherits	Supported	Not supported	Not supported
initial	Supported	Not supported	Supported
initrans	Supported	Not supported	Not supported
initiate	Not supported	Not supported	Supported
inline	Supported	Not supported	Not supported
inner	Supported	Not supported	Not supported
inout	Supported	Not supported	Not supported
input	Supported	Not supported	Not supported
insensitive	Supported	Not supported	Not supported
instead	Supported	Not supported	Not supported
intersect	Not supported	Not supported	Supported
internal	Supported	Not supported	Not supported
interval	Supported	Supported	Not supported
invisible	Supported	Supported	Supported
invoker	Supported	Not supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
isnull	Supported	Not supported	Not supported
json_object	Not supported	Supported	Not supported
json_table	Not supported	Not supported	Supported
json_value	Not supported	Not supported	Supported
keep	Supported	Supported	Not supported
key_path	Supported	Not supported	Not supported
key_store	Supported	Not supported	Not supported
kill	Supported	Not supported	Not supported
keyring	Not supported	Not supported	Supported
label	Supported	Not supported	Not supported
large	Supported	Not supported	Not supported
lag	Not supported	Not supported	Supported
last	Supported	Not supported	Not supported
last_day	Not supported	Supported	Not supported
last_value	Not supported	Not supported	Supported
lateral	Not supported	Not supported	Supported
lc_collate	Not supported	Not supported	Supported
lead	Not supported	Not supported	Supported
leakproof	Supported	Not supported	Supported
least	Supported	Not supported	Not supported
left	Supported	Not supported	Not supported
lines	Supported	Supported	Not supported
link	Supported	Supported	Not supported
listen	Supported	Not supported	Not supported
lnnvl	Supported	Supported	Not supported
load_bad	Supported	Not supported	Supported
load_discard	Supported	Supported	Supported
load_integer_len	Supported	Supported	Not supported
load_session_id	Supported	Supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
load_smallint_len	Supported	Supported	Not supported
load_unfixed_len	Supported	Supported	Not supported
load_unfixed_start_pos	Supported	Supported	Not supported
local	Supported	Not supported	Not supported
location	Supported	Not supported	Supported
locked	Supported	Supported	Supported
logging	Supported	Not supported	Not supported
login_any	Supported	Not supported	Not supported
login_failure	Supported	Not supported	Not supported
login_success	Supported	Not supported	Not supported
logout	Supported	Not supported	Not supported
loop	Supported	Not supported	Not supported
low	Supported	Supported	Not supported
manual	Supported	Supported	Not supported
mapping	Supported	Not supported	Not supported
mark	Supported	Supported	Not supported
masking	Supported	Not supported	Not supported
master	Supported	Not supported	Not supported
master_compression_algorithms	Not supported	Not supported	Supported
master_public_key_path	Not supported	Not supported	Supported
master_tls_ciphersuites	Not supported	Not supported	Supported
master_zstd_compression_level	Not supported	Not supported	Supported
match	Supported	Not supported	Supported
matched	Supported	Not supported	Not supported
materialized	Supported	Not supported	Not supported
maxextents	Supported	Not supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
maxsize	Supported	Not supported	Not supported
maxtrans	Supported	Not supported	Not supported
medium	Supported	Supported	Not supported
mediumint	Supported	Supported	Not supported
member	Not supported	Not supported	Supported
microsecond	Supported	Supported	Not supported
minextents	Supported	Not supported	Not supported
minute	Supported	Not supported	Not supported
minute_microsecond	Supported	Supported	Not supported
minute_second	Supported	Supported	Not supported
minvalue	Supported	Not supported	Supported
model	Supported	Not supported	Not supported
modification	Supported	Supported	Not supported
month	Supported	Not supported	Not supported
months	Supported	Supported	Not supported
move	Supported	Not supported	Supported
movement	Supported	Not supported	Not supported
names	Supported	Not supported	Not supported
national	Supported	Not supported	Not supported
nested	Not supported	Not supported	Supported
next	Supported	Not supported	Not supported
nocache	Supported	Supported	Not supported
nocolview	Supported	Supported	Not supported
nocycle	Not supported	Not supported	Supported
node	Supported	Not supported	Supported
noentityescaping	Supported	Supported	Not supported
noextend	Supported	Supported	Not supported
nologging	Supported	Not supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
nomaxvalue	Supported	Not supported	Supported
nominvalue	Supported	Not supported	Supported
none	Supported	Not supported	Not supported
noscale	Supported	Supported	Not supported
nothing	Supported	Not supported	Not supported
notify	Supported	Not supported	Not supported
notnull	Supported	Not supported	Not supported
now	Supported	Supported	Not supported
nowait	Not supported	Not supported	Supported
nth_value	Not supported	Not supported	Supported
ntile	Not supported	Not supported	Supported
nullcols	Supported	Not supported	Not supported
nullif	Supported	Not supported	Not supported
nulls	Supported	Not supported	Supported
numstr	Supported	Not supported	Not supported
nvl	Supported	Not supported	Not supported
nvl2	Supported	Supported	Not supported
object	Supported	Not supported	Not supported
of	Not supported	Not supported	Supported
off	Not supported	Not supported	Supported
offline	Supported	Supported	Not supported
oids	Supported	Not supported	Supported
oj	Not supported	Not supported	Supported
old	Not supported	Not supported	Supported
online	Supported	Supported	Not supported
open	Supported	Supported	Not supported
operator	Supported	Not supported	Supported
optional	Not supported	Not supported	Supported
ordinality	Supported	Supported	Supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
organization	Supported	Not supported	Supported
optionally	Supported	Not supported	Not supported
options	Supported	Not supported	Not supported
others	Not supported	Not supported	Supported
out	Supported	Not supported	Not supported
outfile	Supported	Supported	Not supported
over	Supported	Not supported	Supported
overlaps	Supported	Not supported	Not supported
overlay	Supported	Not supported	Not supported
owned	Supported	Not supported	Supported
owner	Supported	Not supported	Not supported
ordinality	Supported	Not supported	Not supported
package	Supported	Not supported	Not supported
packages	Supported	Not supported	Not supported
parser	Supported	Not supported	Not supported
partition	Supported	Not supported	Not supported
partitioning	Supported	Supported	Not supported
partitions	Supported	Not supported	Not supported
passing	Supported	Not supported	Not supported
password_lock_time	Not supported	Not supported	Supported
path	Not supported	Not supported	Supported
per	Supported	Not supported	Not supported
percent	Supported	Not supported	Not supported
perm	Supported	Not supported	Not supported
percent_rank	Not supported	Not supported	Supported
performance	Supported	Supported	Not supported
persist	Not supported	Not supported	Supported
persist_only	Not supported	Not supported	Supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
pivot	Supported	Supported	Not supported
pluggable	Supported	Supported	Not supported
plan	Supported	Supported	Supported
plans	Supported	Not supported	Not supported
policy	Supported	Not supported	Not supported
pool	Supported	Not supported	Not supported
populate	Supported	Supported	Not supported
position	Not supported	Not supported	Supported
preceding	Supported	Not supported	Supported
predict	Supported	Not supported	Not supported
preferred	Supported	Not supported	Not supported
prefix	Supported	Not supported	Not supported
prepared	Not supported	Not supported	Supported
preserve	Supported	Not supported	Not supported
priority	Supported	Supported	Not supported
private	Supported	Not supported	Not supported
procedural	Supported	Not supported	Not supported
profile	Supported	Not supported	Not supported
publication	Supported	Not supported	Not supported
privilege_checks_user	Not supported	Not supported	Supported
process	Not supported	Not supported	Supported
public	Supported	Supported	Not supported
publish	Supported	Not supported	Not supported
purge	Supported	Not supported	Not supported
quarter	Supported	Supported	Not supported
query	Supported	Not supported	Not supported
quote	Supported	Not supported	Not supported
random	Not supported	Not supported	Supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
randomized	Supported	Not supported	Not supported
range	Supported	Not supported	Not supported
rank	Not supported	Not supported	Supported
ratio	Supported	Not supported	Not supported
reassign	Supported	Not supported	Not supported
rebuild	Supported	Not supported	Not supported
recheck	Supported	Not supported	Not supported
recover	Supported	Supported	Not supported
recursive	Supported	Not supported	Supported
recyclebin	Supported	Not supported	Supported
redisanyvalue	Supported	Not supported	Not supported
ref	Supported	Not supported	Not supported
refresh	Supported	Not supported	Not supported
reference	Not supported	Not supported	Supported
regexp	Supported	Supported	Not supported
regexp_like	Supported	Supported	Not supported
registration	Not supported	Not supported	Supported
reindex	Supported	Not supported	Supported
reject	Not supported	Not supported	Supported
relative	Supported	Not supported	Supported
reloptions	Supported	Not supported	Not supported
remote	Supported	Not supported	Not supported
remove	Supported	Not supported	Not supported
repeatable	Supported	Not supported	Not supported
replace	Supported	Not supported	Not supported
replica	Supported	Not supported	Supported
replica_type	Supported	Not supported	Not supported
replicas	Not supported	Not supported	Supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
require_row_format	Not supported	Not supported	Supported
resize	Supported	Not supported	Not supported
resource	Not supported	Not supported	Supported
respect	Supported	Supported	Supported
restart	Supported	Not supported	Supported
return	Supported	Not supported	Not supported
returns	Supported	Not supported	Not supported
retain	Not supported	Not supported	Supported
returning	Not supported	Not supported	Supported
reuse	Supported	Not supported	Supported
right	Supported	Not supported	Not supported
rlike	Supported	Supported	Not supported
role	Supported	Not supported	Supported
roles	Supported	Not supported	Not supported
rollup	Supported	Not supported	Not supported
rotation	Supported	Not supported	Not supported
row	Supported	Not supported	Not supported
rows	Supported	Not supported	Not supported
row_number	Not supported	Not supported	Supported
rowtype	Supported	Not supported	Not supported
rule	Supported	Not supported	Not supported
sample	Supported	Not supported	Not supported
scale	Supported	Supported	Not supported
schedule	Supported	Supported	Not supported
scroll	Supported	Not supported	Supported
search	Supported	Not supported	Not supported
second	Supported	Not supported	Not supported
second_microsecond	Supported	Supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
security	Supported	Not supported	Not supported
secondary	Not supported	Not supported	Supported
secondary_engine	Not supported	Not supported	Supported
secondary_engine_attribute	Not supported	Not supported	Supported
secondary_load	Not supported	Not supported	Supported
secondary_unload	Not supported	Not supported	Supported
separator	Supported	Supported	Not supported
sequence	Supported	Not supported	Supported
sequences	Supported	Not supported	Not supported
serializable	Supported	Not supported	Not supported
server	Supported	Not supported	Not supported
sessiontimezone	Supported	Supported	Not supported
setof	Supported	Not supported	Supported
sets	Supported	Not supported	Not supported
sessiontimezone	Supported	Not supported	Not supported
shippable	Supported	Not supported	Supported
show	Supported	Not supported	Not supported
shrink	Supported	Supported	Not supported
siblings	Supported	Not supported	Not supported
signed	Supported	Supported	Not supported
similar	Supported	Not supported	Not supported
simple	Supported	Not supported	Not supported
size	Not supported	Not supported	Supported
skip	Supported	Not supported	Supported
slave	Supported	Supported	Not supported
slice	Supported	Not supported	Supported
smalldatetime	Supported	Not supported	Supported
smalldatetime_format	Supported	Not supported	Supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
smartroute	Supported	Supported	Not supported
source	Supported	Not supported	Not supported
source_auto_position	Not supported	Not supported	Supported
source_bind	Not supported	Not supported	Supported
source_compression_algorithms	Not supported	Not supported	Supported
source_connect_retry	Not supported	Not supported	Supported
source_delay	Not supported	Not supported	Supported
source_heartbeat_period	Not supported	Not supported	Supported
source_host	Not supported	Not supported	Supported
source_log_file	Not supported	Not supported	Supported
source_log_pos	Not supported	Not supported	Supported
source_password	Not supported	Not supported	Supported
source_port	Not supported	Not supported	Supported
source_public_key_path	Not supported	Not supported	Supported
source_retry_count	Not supported	Not supported	Supported
source_ssl	Not supported	Not supported	Supported
source_ssl_ca	Not supported	Not supported	Supported
source_ssl_capath	Not supported	Not supported	Supported
source_ssl_cert	Not supported	Not supported	Supported
source_ssl_cipher	Not supported	Not supported	Supported
source_ssl_crl	Not supported	Not supported	Supported
source_ssl_crlpath	Not supported	Not supported	Supported
source_ssl_key	Not supported	Not supported	Supported
source_ssl_verify_server_cert	Not supported	Not supported	Supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
source_tls_ciphersuites	Not supported	Not supported	Supported
source_tls_version	Not supported	Not supported	Supported
source_user	Not supported	Not supported	Supported
source_zstd_compression_level	Not supported	Not supported	Supported
space	Supported	Not supported	Not supported
specification	Supported	Supported	Not supported
spill	Supported	Not supported	Not supported
split	Supported	Not supported	Supported
srid	Not supported	Not supported	Supported
stable	Supported	Not supported	Supported
standalone	Supported	Not supported	Not supported
starting	Supported	Supported	Not supported
starts	Supported	Supported	Not supported
statement	Supported	Not supported	Not supported
statement_id	Supported	Not supported	Not supported
statistics	Supported	Not supported	Not supported
stdin	Supported	Not supported	Supported
stdout	Supported	Not supported	Supported
storage	Supported	Not supported	Not supported
store	Supported	Not supported	Not supported
stored	Supported	Not supported	Not supported
str	Supported	Supported	Not supported
stratify	Supported	Not supported	Not supported
stream	Not supported	Not supported	Supported
strict	Supported	Not supported	Supported
strip	Supported	Not supported	Not supported
subdate	Not supported	Supported	Not supported
subpartition	Supported	Not supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
subpartitioning	Supported	Supported	Not supported
subpartitions	Supported	Supported	Not supported
subscription	Supported	Not supported	Not supported
substr	Supported	Supported	Not supported
substring	Supported	Not supported	Supported
sys_refcursor	Supported	Not supported	Not supported
sysid	Supported	Not supported	Not supported
sysdate	Not supported	Not supported	Supported
system	Not supported	Not supported	Supported
table_compress_clause	Supported	Supported	Not supported
thread_priority	Not supported	Not supported	Supported
ties	Not supported	Not supported	Supported
timestampadd	Supported	Supported	Not supported
tls	Not supported	Not supported	Supported
trim	Not supported	Not supported	Supported
turbo	Supported	Supported	Not supported
table_compress_clause	Supported	Not supported	Not supported
tables	Supported	Not supported	Not supported
tablesample	Supported	Not supported	Not supported
tablespace	Supported	Not supported	Not supported
target	Supported	Not supported	Not supported
temp	Supported	Not supported	Not supported
template	Supported	Not supported	Not supported
temporary	Supported	Not supported	Not supported
terminated	Supported	Not supported	Not supported
than	Supported	Not supported	Not supported
time_format	Supported	Not supported	Not supported
timecapsule	Supported	Not supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
timestampadd	Not supported	Not supported	Not supported
timestamp_format	Supported	Not supported	Not supported
timestampdiff	Supported	Not supported	Not supported
transform	Supported	Not supported	Not supported
treat	Supported	Not supported	Not supported
trim	Supported	Not supported	Not supported
trusted	Supported	Not supported	Not supported
tsfield	Supported	Not supported	Not supported
tstag	Supported	Not supported	Not supported
tstime	Supported	Not supported	Not supported
turbo	Supported	Not supported	Not supported
types	Supported	Not supported	Not supported
unbounded	Supported	Not supported	Supported
uncommitted	Supported	Not supported	Not supported
unencrypted	Supported	Not supported	Not supported
unit	Supported	Supported	Not supported
unknown	Supported	Not supported	Not supported
unlimited	Supported	Not supported	Not supported
unlisten	Supported	Not supported	Not supported
unlock	Supported	Not supported	Not supported
unlogged	Supported	Not supported	Not supported
unpivot	Supported	Supported	Not supported
unregister	Not supported	Not supported	Supported
unsigned	Supported	Supported	Not supported
until	Supported	Not supported	Not supported
unusable	Supported	Not supported	Supported
url	Not supported	Not supported	Supported
useeof	Supported	Not supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
vacuum	Not supported	Not supported	Supported
valid	Supported	Not supported	Supported
validate	Supported	Not supported	Not supported
validation	Supported	Not supported	Not supported
validator	Supported	Not supported	Not supported
varchar2	Not supported	Not supported	Supported
variables	Supported	Not supported	Not supported
varying	Supported	Not supported	Not supported
vcgroup	Supported	Not supported	Not supported
vcpu	Not supported	Not supported	Supported
verbose	Supported	Not supported	Supported
verify	Supported	Supported	Not supported
version	Supported	Not supported	Supported
visible	Supported	Supported	Supported
volatile	Supported	Not supported	Supported
wait	Supported	Not supported	Not supported
weak	Supported	Not supported	Not supported
week	Supported	Supported	Not supported
wellformed	Supported	Supported	Not supported
whitespace	Supported	Not supported	Not supported
window	Not supported	Not supported	Supported
within	Supported	Not supported	Supported
without	Supported	Not supported	Not supported
workload	Supported	Not supported	Not supported
wrapper	Supported	Not supported	Not supported
xmlattributes	Supported	Not supported	Not supported
xmlconcat	Supported	Not supported	Not supported
xmlelement	Supported	Not supported	Not supported
xmlexists	Supported	Not supported	Not supported

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Keyword	ORA Compatibility	MySQL, TD, and PG Compatibilities	M Compatibility
xmlforest	Supported	Not supported	Not supported
xmlnamespaces	Supported	Supported	Not supported
xmlparse	Supported	Not supported	Not supported
xmlpi	Supported	Not supported	Not supported
xmlroot	Supported	Not supported	Not supported
xmltable	Supported	Supported	Not supported
xmlquery	Supported	Supported	Not supported
xmlcast	Supported	Supported	Not supported
xmlserialize	Supported	Not supported	Not supported
xmltype	Supported	Supported	Not supported
year_month	Supported	Supported	Not supported
years	Supported	Supported	Not supported
zerofill	Supported	Supported	Not supported
zone	Supported	Not supported	Supported

Default value:

- In non-M-compatible database: **"datetime,regexp,rlike,zerofill"**
- In M-compatible database: **""**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is enabled, some functions used as keywords will become invalid. Exercise caution when setting this parameter. If the keyword to be shielded does not exist in the to-be-shielded keyword array of the current compatibility mode database but exists in other compatibility mode databases, WARNING is generated. If the keyword to be shielded does not exist in other compatibility mode databases, ERROR is generated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

In MySQL-compatible mode, whether **datetime**, **regexp**, **rlike**, and **zerofill** are reserved keywords is determined by the system and cannot be modified or deleted by users. Specifically, in the initial state, the system adds **datetime**, **regexp**, **rlike**, and **zerofill** to the **disable_keyword_options** parameter. In this case, **datetime**, **regexp**, **rlike**, and **zerofill** are not reserved keywords. After **b_format_version** is set to **'5.7'** and **b_format_dev_version** is set to **'s1'** (or a version later than s1), the system deletes **datetime**, **regexp**, **rlike**, and **zerofill** from **disable_keyword_options**. In this case, **datetime**, **regexp**, **rlike**, and **zerofill** can be used as reserved keywords.

disable_plsql_keyword_options

Parameter description: Specifies the database compatibility behavior, that is, whether to use an identifier as a non-keyword. Values of this parameter are separated by commas (,).

Parameter type: string

Unit: none

Value range: **PIPE**, **PIPELINED**, **RANGE**, **REPLACE**, and **SUBTYPE**.

Default value: **""**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

- If this parameter is enabled, some functions used as keywords will become invalid. Exercise caution when setting this parameter.
- To cancel the function of shielding the PL/SQL keyword, leave this parameter empty.

td_compatible_truncation

Parameter description: Specifies whether to enable features compatible with a Teradata database. You can set this parameter to **on** when connecting to a database compatible with the Teradata database, so that when you perform the INSERT operation, overlong strings are truncated based on the allowed maximum length before being inserted into char- and varchar-type columns in the target table. This ensures all data is inserted into the target table without errors reported.

NOTE

- The string truncation function cannot be used if the INSERT statement includes a foreign table.
- If multi-byte character data (such as Chinese characters) is inserted into a database with the character set byte encoding (such as SQL_ASCII and LATIN1), and the character data crosses the truncation position, the string is truncated based on its bytes instead of characters. Unexpected result will occur in tail after the truncation. To ensure correct truncation behavior, it is advisable to use character-based encoding such as UTF-8 as the database's character set.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that overlong strings are automatically truncated.
- **off** indicates that overlong strings are not automatically truncated.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

uppercase_attribute_name

Parameter description: Specifies whether to return column names in uppercase to the client. This parameter is used only in ORA-compatible mode.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Column names are returned to the client in uppercase.
- **off**: Column names are not returned to the client in uppercase.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- Do not run **gs_guc set**, **gs_guc reload**, **alter database set**, or **alter user set** or modify **gaussdb.conf** to modify this parameter globally.
- When this parameter is enabled and an alias with quotation marks is specified for a projection column, whether the column name is returned in uppercase is also controlled by the **uppercase_attribute_name_respect_quote** parameter.
- After this parameter is enabled, the column name is converted to uppercase only when the column name is in lowercase. The uppercase column name is returned to the client. When the column name is case-insensitive, the column name is not converted to uppercase.

```
gaussdb=# SET uppercase_attribute_name=off;
SET
gaussdb=# SELECT 1 AS ab, 2 AS Ab, 3 AS AB, 4 AS "ab", 5 AS "Ab", 6 AS "AB";
 ab | ab | ab | ab | Ab | AB
----+----+----+----+----+----
  1 |  2 |  3 |  4 |  5 |  6
(1 row)

gaussdb=# SET uppercase_attribute_name=on;
SET
gaussdb=# SELECT 1 AS ab, 2 AS Ab, 3 AS AB, 4 AS "ab", 5 AS "Ab", 6 AS "AB";
 AB | AB | AB | AB | Ab | AB
----+----+----+----+----+----
  1 |  2 |  3 |  4 |  5 |  6
(1 row)
```

uppercase_attribute_name_respect_quote

Parameter description: Specifies whether to retain the original case of quoted column aliases after the **uppercase_attribute_name** parameter is enabled when the column aliases are returned to the client. This parameter is used only in ORA-compatible mode. This parameter can be set at the PDB level.

Parameter type: enumerated type

Unit: none

Value range:

- **off**: The quoted column aliases that are in lowercase are converted to uppercase and returned. The quoted column aliases that are in uppercase or mixed case are returned in the original case.
- **select_alias**: In the scenario where the outermost layer is a SELECT query and the projection column specifies an alias with quotation marks, the original case is retained. In other scenarios, the behavior is the same as that of **off**.

Default value: **off**. In the PDB scenario, if this parameter is not set, the global setting is inherited.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- For an upgrade from a version that does not support this parameter to the current version, this parameter does not take effect if the upgrade is not committed.
- Do not run **gs_guc set**, **gs_guc reload**, **alter database set**, or **alter user set** or modify **gaussdb.conf** to modify this parameter globally.
- This parameter takes effect only when **uppercase_attribute_name** is enabled. If this parameter is set to **select_alias**, only SELECT statements are affected, and scenarios such as PBE, CREATE VIEW, and DECLARE CURSOR are not affected.

```
gaussdb=# SET uppercase_attribute_name=on;
SET
gaussdb=# SET uppercase_attribute_name_respect_quote=off;
SET
gaussdb=# SELECT 1 AS "a";
 A
---
 1
(1 row)

gaussdb=# SET uppercase_attribute_name_respect_quote=select_alias;
SET
gaussdb=# SELECT 1 AS "a";
 a
---
 1
(1 row)
gaussdb=# PREPARE p1 AS SELECT 1 AS "a";
PREPARE
gaussdb=# EXECUTE p1;
 A
---
 1
(1 row)

gaussdb=# CREATE VIEW v1 AS SELECT 1 AS "a";
CREATE VIEW
gaussdb=# SELECT * FROM v1;
 A
---
 1
(1 row)

gaussdb=# CREATE OR REPLACE PROCEDURE p_test_quote(rc INOUT REFCURSOR) AS
gaussdb$# BEGIN
gaussdb$#     OPEN rc FOR SELECT 1 AS "ab";
gaussdb$# END;
gaussdb$# /
CREATE PROCEDURE
gaussdb=# BEGIN;
BEGIN
gaussdb=# CALL p_test_quote('rc');
 RC
----
 rc
(1 row)

gaussdb=# FETCH ALL FROM rc;
 AB
----
  1
(1 row)

gaussdb=# COMMIT;
COMMIT
gaussdb=# DECLARE c CURSOR FOR SELECT 1 AS "ab", 2 AS "cd";
DECLARE CURSOR
gaussdb=# FETCH 1 FROM c;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 AB | CD
----+----
  1 |  2
(1 row)

gaussdb=# CLOSE c;
CLOSE CURSOR
gaussdb=# COMMIT;
COMMIT
```

enable_object_special_character

Parameter description: Determines whether the value of the **schema** parameter in the control file can contain any special characters in ["$'\] when the CREATE EXTENSION statement is executed and "@extschema@" is used in the script file.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the value can contain any special characters in ["$'\].
- **off** indicates that the value cannot contain any special characters in ["$'\].

Default value:

- **off**: default value of the newly installed database.
- **on**: This parameter is added in 505.0. If a database is upgraded from a version earlier than 505.0, the value of this parameter is **on**. If this parameter exists and is set before the upgrade, the default value remains unchanged after the upgrade.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Extensions are for internal use only. Only advanced packages or extensions in the database can be operated.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

enable_implicit_cast_str2num

Parameter description: Specifies whether a string can be converted to a NUMERIC type when a comparison or arithmetic operator is performed between an integer data type and a string data type.

NOTE

This parameter is valid only when **a_format_version** is set to **'10c'** and **a_format_dev_version** is set to **'s1'**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Comparison operators (>, >=, <, <=, =, and <>): The string data type is implicitly converted to the NUMERIC type. If the **enable_crosstype_integer_operator** option of the **behavior_compat_options** parameter is disabled, the integer data type is implicitly converted to the NUMERIC type.

Arithmetic operators (+, -, *, /,%, and ^): Both integer and string data types are implicitly converted to the NUMERIC type.

- **off**: disabled. The string data type will be implicitly converted to the integer data type.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

a_format_disable_func

Parameter description: Disables a specified system function. The value of this parameter consists of multiple system function OIDs separated by commas (,). The system function for which this parameter is set cannot be called. When a system function cannot meet user requirements and a user-defined function with the same name is required, you can use this function.

Parameter type: string

Unit: none

Value range: a string consisting of multiple system function OIDs separated by commas (,).

Default value: **""**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_convert_illegal_char

Parameter description: Specifies whether invalid characters in the command output are not verified and are displayed as placeholders.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The parameter is enabled. Invalid sequences are replaced by the symbols specified by the **convert_illegal_char_mode** parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **off**: The parameter is disabled. If the query result contains invalid characters, an error is reported after verification.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Enable this parameter only when the data contains special characters and errors should not be reported for special characters.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

1. When the database character set is utf8, zhs16gbk, gb18030, gb18030-2022, or latin1, **enable_convert_illegal_char=on** takes effect. If the character set of the database client is different from that of the database server, invalid characters are displayed as placeholders.
2. Special character range: The special characters mentioned in this document include only fully abnormal encoding and hybrid encoding. The \u0000 character is not supported. If the character code contains the \u0000 character, the character is truncated at \u0000, which affects data integrity.
3. In the PBE scenario, if the character set of the database client is the same as that of the server, the bound parameters do not require encoding conversion. However, the validity of the bound parameters still needs to be verified, and invalid characters in the bound parameters need to be processed during the verification. When the **enable_convert_illegal_char** parameter is disabled, invalid characters (excluding \u0000) in the bound parameters are replaced with placeholders (characters specified by the **convert_illegal_char_mode** parameter). When the **enable_convert_illegal_char** parameter is enabled, the original characters are retained.
4. If the character sets of the database client and server are different, when **enable_convert_illegal_char** is set to **off**, the database replaces special characters (excluding \x0000) that exceed the current character set codes with **\x3F** (displayed as **?**) and writes the special characters into the database; when **enable_convert_illegal_char** is set to **on**, only the dbe_raw.cast_to_varchar2() function can be used to insert the original invalid character into the database and display the character as a placeholder.
5. If the character sets of the database client and server are the same, when **enable_convert_illegal_char** is set to **off**, the database replaces special characters (excluding \x0000) that exceed the current character set codes with **\x3F** (displayed as **?**) and writes the special characters into the database; when **enable_convert_illegal_char** is set to **on**, only the dbe_raw.cast_to_varchar2() function can be used to insert the original invalid character into the database.
6. Only encoding processing between the client character set and the server character set is supported. The behavior of encoding verification that is not between the client character set and the server character set is the same as that when the parameter is disabled.
7. When the GUC parameter is enabled, the behaviors of the special characters in the functions and advanced package functions listed in the following table are as follows:
   - When the character sets of the client and server are the same, no error is reported during the query of special characters. The behavior is the same as that before the GUC parameter is enabled.
   - If the character sets of the client and server are different, special characters are displayed as placeholders, which use question marks (?) by default.
   - You are advised not to use functions to process character strings that contain special characters. The functions listed in the following table do not report errors when processing character strings that contain special characters, and the correctness of the results cannot be ensured.

Table 6-39 Functions and advanced package functions supporting special characters

No.	Function/Advanced Package Function
1	bit_length(string)
2	btrim(string text [, characters text])

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Function/Advanced Package Function
3	char_length(string) character_length(string)
4	chr(cvalue int|bigint) chr(integer)
5	concat(str1,str2)
6	concat_ws(sep text, str"any" [, str"any" [, ...]])
7	decode(string text, format text)
8	dump(expr[, return_fmt [, start_position [, length]]])
9	encode(data bytea, format text)
10	find_in_set(text, set)
11	format(formatstr text [, str"any" [, ...]])
12	left(str text, n int)
13	length(string)
14	lengthb(text/bpchar)
15	ltrim(string [, characters])
16	md5(string)
17	notlike(x bytea name text, y bytea text)
18	octet_length(string)
19	overlay(string placing string FROM int [for int])
20	quote_ident(string text)
21	quote_literal(string text)
22	quote_nullable(string text)
23	rawcat(raw,raw)
24	regexp_count(string text, pattern text [, position int [, flags text]])
25	regexp_instr(string text, pattern text [, position int [, occurrence int [, return_opt int [, flags text]]]])
26	regexp_like(source_string, pattern [, match_parameter]) regexp_like(text,text,text)
27	regexp_matches(string text, pattern text [, flags text])

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Function/Advanced Package Function
28	regexp_replace(string, pattern, replacement [,flags]) regexp_replace(string text, pattern text [, replacement text [, position int [, occurrence int [, flags text]]]])
29	regexp_split_to_array(string text, pattern text [, flags text])
30	regexp_split_to_table(string text, pattern text [, flags text])
31	regexp_substr(source_char, pattern) regexp_substr(string text, pattern text [, position int [, occurrence int [, flags text]]])
32	repeat(string text, number int)
33	replace(string text, from text, to text)
34	replace(string, substring)
35	reverse(str)
36	right(str text, n int)
37	rtrim(string [, characters])
38	sha(string)
39	sha1(string)
40	sha2(string, hash_length)
41	split_part(string text, delimiter text, field int)
42	substring(string [from int] [for int]) substring(string from pattern for escape) substring(string from pattern)
43	substring_inner(string [from int] [for int])
44	tconvert(key text, value text)
45	to_single_byte(char)
46	translate(string text, from text, to text)
47	trim([leading |trailing |both] [characters] from string)
48	unistr(string)
49	vsize(expr)
50	PKG_UTIL.RAW_CAST_FROM_VARCHAR2
51	PKG_UTIL.LOB_CONVERTTOCLOB
52	PKG_UTIL.LOB_RAWTOTEXT
53	PKG_UTIL.LOB_TEXTTORAW

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Function/Advanced Package Function
54	PKG_UTIL.RAW_CAST_TO_VARCHAR2
55	DBE_OUTPUT.PUT
56	DBE_OUTPUT.PUT_LINE

fix_func_selection

Parameter description: Specifies whether to optimize the function matching policy.

The catlist sequence issue occurs in this case: If a user-defined function conflicts with a system function, the function selected by the database depends on the registration sequence of the system function in the database system.

Parameter type: string

Unit: none

Value range: **""** and **"catlist"**.

- **""**: No optimization is performed. The value is the same as that in versions earlier than 505.1.0.
- **"catlist"**: The catlist sequence is optimized. System functions are always preferentially selected and executed.

Default value:

- **"catlist"**: default value of the newly installed database
- **""**: default value of the database in versions earlier than 505.1.0 after the database is upgraded.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_allowed_packet

Parameter description: Specifies the upper limit of the return value of a string function in M-compatible database. If the actual result exceeds the upper limit, an alarm is generated and **NULL** is returned.

Parameter type: integer.

Unit: bytes

Value range: 1024 to 1073741824. The value must be a multiple of 1024. Otherwise, the value is rounded down to the nearest multiple.

Default value: **4194304**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

div_precision_increment

Parameter description: Specifies the database configuration item in M-compatible mode. This parameter specifies the precision improvement value of the division result. That is, the number of decimal places in the division result is equal to the number of decimal places in the divisor plus the value of this parameter.

Parameter type: integer.

Unit: none

Value range: 0 to 30

Default value: **4**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_m_format_hook

Parameter description: Specifies the database configuration item in M-compatible mode. This parameter specifies whether the hook in M-compatible mode takes effect.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The M-compatible database allows the corresponding parser and executor hooks to be mounted.
- **off**: The M-compatible database does not mount the corresponding parser and executor hooks.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is used only for external tools that are not fully adapted and cannot be used by users.

Risks and impacts of improper settings: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

gs_format_behavior_compat_options

Parameter description: Selects configuration items of GaussDB internal system functions.

Parameter type: string

Unit: none

Value range: Currently, only compatibility configuration items listed in **Table 6-40** are supported. Compatibility configuration items are separated by commas (,).

Default value: **""**

Table 6-40 gs_format_behavior compatibility configuration items

Configuration Item	Behavior
sqrt_karatsuba	• If this parameter is set, the Karatsuba square root algorithm is used when the sqrt function is called. The Karatsuba algorithm has higher performance but its precision may be different from that of the Newton iteration algorithm in rare cases. • If this parameter is not set, the default Newton iteration algorithm is used to calculate the square root for the sqrt algorithm.

allow_textconcat_null	• If this parameter is set, the value of the corresponding character string is returned when a character string is combined with the **null** value in PG-compatible mode.
	`-- Run the following command in PG-compatible mode:`
	`gaussdb=# set gs_format_behavior_compat_options='allow_textconcat_null';`
	`SET`
	`gaussdb=# select 'a' || null || 'b';`
	` ?column?`
	`----------`
	` ab`
	`(1 row)`
	• If this parameter is not set, **NULL** is returned when a character string is combined with the **null** value in PG-compatible mode.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Determines whether some compatibility features are available. If you want to modify this parameter, make sure you understand its meaning and modify it with caution to avoid risks caused by misoperations.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

support_zero_character

Parameter description: Specifies whether the database cancels the verification of the zero character and correctly processes it without truncating the character when the data contains the character whose Unicode code is \u0000 (referred to as the zero character).

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled. The database cancels the verification of the zero character and can correctly insert and process the zero character so that the data is not truncated during processing.
- **off**: disabled. The database retains the verification of the zero character. When some functions process the zero character, an error is reported or data is truncated.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Enable this parameter only when the data contains the zero character and the zero character need to be correctly processed.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- This parameter takes effect only when the ORA-compatible database is used and **support_zero_character** is set to **on** for all nodes.
- If you want to write zero characters to the database, ensure that **support_zero_character** is set to **on** and the setting takes effect so that the zero characters can be properly inserted and stored.
- When **support_zero_character** is set to **on** and the data type is CHAR(n), CHARACTER(n), NCHAR(n), VARCHAR(n), CHARACTER VARYING(n), VARCHAR2(n), NVARCHAR2(n), TEXT, or CLOB, the database can correctly insert and process zero characters. If the string data contains zero characters and is explicitly converted to JSON/JSONB, XML, XMLTYPE, numeric, currency, time, geometric, network address, bit string, text (for search), HLL, range, object identifier, ledger database, aclitem, or vector type, the string will be truncated at the zero character. If the string is explicitly converted to the UUID type, an error is reported at the zero character.
- The database supports only the following methods of writing the zero characters:
  - When the database is connected through JDBC, the zero character can be written to the database in text mode.
  - Use the dbe_raw.cast_to_varchar2 function to directly write the zero character code to the database.
  - Use the copy tool to write the zero character code to the database.
- When the GUC parameter is enabled and different character sets are converted, the zero character is directly returned and no error is reported.
- The gsql client version must match the database version. When the GUC parameter is enabled, if the data received by gsql contains zero characters, gsql converts the zero characters to spaces and then prints the data. If the gsql tool of a version earlier than 505.2.0 is used to connect to the database, data containing zero characters is still truncated.
- When the GUC parameter is enabled, the zero characters can be properly processed by the functions listed in **Table 6-41** and will not be truncated.
- The input parameters of some advanced package APIs are schema names or object names, which cannot contain zero characters theoretically. Therefore, zero character verification is added to the functions in **Table 6-42**. When the input parameters contain zero characters, an error is reported. (In earlier versions, data after the zero characters is truncated.)
- When **support_zero_character** is set to **on**, if the processed data does not contain zero characters, the function performance does not deteriorate significantly. If the processed data contains zero characters, the performance deterioration of the initcap, upper, lower, nls_upper, and nls_lower functions is closely related to the number of zero characters in the data. That is, the more zero characters there are, the more severe the performance deterioration.
- When **support_zero_character** is set to **on**, if the logical decoding tool reads zero characters in the data, zero characters are converted to spaces and then printed. (In earlier versions, data after the zero characters is truncated.)

Table 6-41 List of system functions and advanced package functions supported by zero characters

No.	Function/Advanced Package Function
1	btrim(string text [, characters text])
2	char_length(string) or character_length(string)
3	chr(cvalue int|bigint)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Function/Advanced Package Function
4	chr(integer)
5	coalesce(expr1, expr2, ..., exprn)
6	concat(str1,str2)
7	concat_ws(sep text, str"any" [, str"any" [, ...]])
8	convert(string bytea, src_encoding name, dest_encoding name)
9	convert_from(string bytea, src_encoding name)
10	convert_to(string text, dest_encoding name)
11	decode(base_expr, compare1, value1, Compare2,value2, ... default)
12	dump(expr[, return_fmt [, start_position [, length]]])
13	encode(data bytea, format text)
14	format(formatstr text [, str"any" [, ...]])
15	greatest(expr1 [, ...])
16	group_concat([DISTINCT | ALL] expression [,expression ...] [ORDER BY { expression [[ASC | DESC | USING operator] | nlssort_expression_clause] [NULLS { FIRST | LAST }] } [,...]] [SEPARATOR str_val])
17	initcap(string)
18	instr(string,substring[,position,occurrence]) instr(text,text,int,int)
19	instrb(text,text,int,int)
20	least(expr1 [, ...])
21	left(str text, n int)
22	length(string) length(string bytea, encoding name)
23	lengthb(text/bpchar)
24	listagg(expression [, delimiter]) WITHIN GROUP(ORDER BY order-list)
25	lower(string)
26	lpad(string text, length int [, fill text])
27	ltrim(string [, characters])
28	nls_lower(string [, nlsparam])

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Function/Advanced Package Function
29	nls_upper(string [, nlsparam])
30	nullif(expr1, expr2)
31	nvl(expr1 , expr2)
32	nvl2(expr1, expr2, expr3)
33	overlay(string placing string FROM int [for int])
34	position(substring in string)
35	quote_nullable(string text)
36	regexp_count(string text, pattern text [, position int [, flags text]])
37	regexp_instr(string text, pattern text [, position int [, occurrence int [, return_opt int [, flags text]]]])
38	regexp_like(text,text,text) regexp_like(source_string, pattern [, match_parameter])
39	regexp_matches(string text, pattern text [, flags text])
40	regexp_replace(string, pattern, replacement [,flags])
41	regexp_split_to_array(string text, pattern text [, flags text])
42	regexp_split_to_table(string text, pattern text [, flags text])
43	regexp_substr(source_char, pattern) regexp_substr(string text, pattern text [, position int [, occurrence int [, flags text]]])
44	repeat(string text, number int)
45	replace(string text, from text, to text) replace(string, substring)
46	reverse(str)
47	right(str text, n int)
48	rpad(string text, length int [, fill text])
49	rtrim(string [, characters])
50	similar_escape(pat text, esc text)
51	split_part(string text, delimiter text, field int)
52	string [NOT] LIKE pattern [ESCAPE escape-character]
53	string || string, string || non-string, or non-string || string

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Function/Advanced Package Function
54	string_agg(expression, delimiter)
55	strpos(string, substring)
56	substr(string,from) substr(bytea,from,count) substr(string,from,count)
57	substrb(text,int) substrb(text,int,int)
58	substring(string [from int] [for int]) substring(string from pattern for escape) substring(string from pattern) substring_inner(string [from int] [for int])
59	translate(string text, from text, to text)
60	trim([leading |trailing |both] [characters] from string)
61	unistr(string)
62	upper(string)
63	PKG_UTIL.LOB_GET_LENGTH
64	PKG_UTIL.LOB_READ
65	PKG_UTIL.LOB_WRITE
66	PKG_UTIL.LOB_APPEND
67	PKG_UTIL.LOB_COMPARE
68	PKG_UTIL.LOB_MATCH
69	PKG_UTIL.LOB_READ_HUGE
70	PKG_UTIL.LOB_WRITEAPPEND_HUGE
71	PKG_UTIL.LOB_APPEND_HUGE
72	PKG_UTIL.LOB_COPY_HUGE
73	PKG_UTIL.LOB_CONVERTTOBLOB_HUGE
74	PKG_UTIL.IO_PRINT
75	PKG_UTIL.RAW_CAST_FROM_VARCHAR2
76	PKG_UTIL.FILE_READ
77	PKG_UTIL.FILE_READLINE
78	PKG_UTIL.APP_READ_CLIENT_INFO

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

No.	Function/Advanced Package Function
79	PKG_UTIL.APP_SET_CLIENT_INFO
80	PKG_UTIL.LOB_CONVERTTOBLOB
81	PKG_UTIL.LOB_CONVERTTOCLOB
82	PKG_UTIL.MATCH_EDIT_DISTANCE_SIMILARITY
83	PKG_UTIL.RAW_CAST_TO_VARCHAR2
84	PKG_UTIL.APP_SET_MODULE
85	PKG_UTIL.APP_READ_MODULE
86	PKG_UTIL.APP_SET_ACTION
87	PKG_UTIL.APP_READ_ACTION
88	DBE_OUTPUT.PRINT_LINE
89	DBE_OUTPUT.PRINT
90	DBE_OUTPUT.GET_LINE
91	DBE_OUTPUT.GET_LINES
92	DBE_OUTPUT.PUT
93	DBE_OUTPUT.PUT_LINE
94	DBE_UTILITY.CANONICALIZE
95	DBE_UTILITY.COMMA_TO_TABLE
96	DBE_UTILITY.NAME_TOKENIZE
97	DBE_UTILITY.TABLE_TO_COMMA
98	DBE_UTILITY.CANONICALIZE_RET
99	DBE_UTILITY.COMMA_TO_TABLE_FUNC
100	DBE_UTILITY.NAME_SEPARATE
101	DBE_UTILITY.NAME_TOKENIZE_FUNC
102	DBE_UTILITY.NAME_TOKENIZE_LOWER
103	DBE_UTILITY.NAME_TOKENIZE_LOWER_FUNC
104	DBE_UTILITY.TABLE_TO_COMMA_FUNC

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-42 Functions for reporting an error when the input parameter contains zero characters

No.	Advanced Package Function
1	PKG_UTIL.SESSION_SET_CONTEXT
2	PKG_UTIL.UTILITY_COMPILE_SCHEMA
3	PKG_UTIL.GS_COMPILE_SCHEMA
4	DBE_UTILITY.NAME_RESOLVE
5	DBE_UTILITY.COMPILE_SCHEMA
6	DBE_UTILITY.SEARCH_CLASS_WITH_NSPOID_ONAME_TYPE
7	DBE_UTILITY.SEARCH_OBJECTS
8	DBE_UTILITY.SEARCH_OBJECTS_SYNONYM_FILL_SCHEMA
9	DBE_UTILITY.SEARCH_PROCEDURE_WITH_NSPOID_ONAME
10	DBE_UTILITY.SEARCH_SYNONYM_WITH_NSPOID_ONAME
11	PG_CATALOG.GS_COMPILE_SCHEMA

enable_case_intervaltonumeric

Parameter description: Specifies whether INTERVAL can be converted to NUMERIC in CASE statements. This parameter takes effect only in ORA-compatible mode.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: INTERVAL can be converted to NUMERIC in the CASE statement.
- **off**: INTERVAL cannot be converted to NUMERIC in the CASE statement.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **on** and then modified to **off**, an error may be reported for the existing CASE statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_ecpg_cursor_duplicate_operation

Parameter description: Specifies whether to allow ECPG to control a cursor can be opened or closed repeatedly in ORA-compatible mode. This parameter is valid only when ECPG is connected to ORA-compatible databases.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: A cursor can be opened or closed repeatedly when ECPG connects to an ORA-compatible database.
- **off**: A cursor cannot be opened or closed repeatedly when ECPG connects to an ORA-compatible database.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **off**, an error will be reported when a cursor is opened or closed repeatedly.

NOTE

If the current database version is directly installed, the default value of this parameter is **on**. For a database upgraded from an earlier version, if the source version does not support this parameter, the default value of this parameter is **off** after the upgrade. If the source version supports this parameter, the current setting of this parameter is retained after the upgrade.

format_pg_to_timestamp

Parameter description: Controls the behavior of the to_timestamp(string [, fmt]) function in PG-compatible mode. After this parameter is enabled, unsupported formats are automatically skipped. The current version does not support formats such as FF, FF7, FF8, FF9, RR, RRRR, SYYYY, x and X. Dates earlier than 294277-01-01 00:00:00 and negative years (converted to BC) are supported. FF1 to FF6 round off the milliseconds that exceed the precision. When the same items in the time and date conflict (for example, the days specified by DDD and DD are different), a time in the correct format is returned and no error is reported. For details about the conflict behavior, see **Table 6-43**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The preceding functions are enabled.
- **off**: The preceding functions are disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is enabled, a format that is not supported by GaussDB will become invalid.

Table 6-43 to_timestamp fmt conflicts

Type	Description
SSSSS conflicts with time.	Retain minutes and hours of time and the seconds part of SSSSS.
W and J conflict with time and date.	Priority: Overwrite J, W, and mmdd in sequence. W: indicates Wednesday by default. Default behavior: Overwrite mmdd with W.
SSSSS conflicts with SS.	Retain SS and discard SSSSS.
DDD conflicts with a date.	Discard DDD.

synonym_priv_strict_check

Parameter description: Specifies whether to check access to objects associated with a synonym when you search for an object using the synonym.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The access permission on the objects associated with the synonyms is checked when searching for objects by synonyms.
- **off**: The access permission on the objects associated with the synonyms is not checked when searching for objects by synonyms.

Default value:

- If a new version is directly installed, the default value is **on**.
- For a database upgraded from an earlier version, if the source version does not support the GUC parameter, the default value is **off**. If the source version supports the GUC parameter, the default value is **on**.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the parameter is set to **off**, the access permission on the objects associated with the synonyms is not checked when searching for objects by synonyms, which may cause unauthorized operations.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

forbid_public_funcname_same_with_sysfunc

Parameter description: Specifies whether to forbid PUBLIC functions to have the same name as system functions in pg_catalog when creating functions and RENAME functions.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: When creating a function or rename a function, the name of the PUBLIC function cannot be the same as that of a system function in pg_catalog.
- **off**: When creating a function or rename a function, the name of the PUBLIC function can be the same as that of a system function in pg_catalog.

Default value:

- If a new version is directly installed, the default value is **on**.
- For a database upgraded from an earlier version, if the source version does not support the GUC parameter, the default value is **off**. If the source version supports the GUC parameter, the default value is **on**.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **off**, the PUBLIC function can have the same name as a system function in pg_catalog.

a_format_func_col_name

Parameter description: Specifies the system function compatibility configuration item in the ORA-compatible database.

Parameter type: string

Unit: none

Value range: Currently, only information in **Table 6-44** are supported. Compatibility configuration items are separated by commas (,).

Default value: **""** (an empty string)

Table 6-44 System function compatibility configuration items

Configuration Item	Behavior
user	After this parameter is enabled, the name of the projection column returned by **SELECT user;** is **user**.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Adjust the parameter value based on the database-compatible objects.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_cursor_with_hold

Parameter description: Specifies whether a cursor has the WITH HOLD function in ORA-compatible mode. After **enable_cursor_with_hold** is enabled, if the WITHOUT HOLD keyword is not specified and the SELECT clause does not contain the FOR UPDATE or FOR SHARE clause when a cursor is created, the WITH HOLD function is used by default.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: If the WITHOUT HOLD keyword is not specified and the SELECT clause does not contain the FOR UPDATE or FOR SHARE clause when a cursor is created, the WITH HOLD function is used by default. In this case, **CURSOR cursor_name FOR query;** is equivalent to **CURSOR cursor_name WITH HOLD FOR query;**.
- **off**: If WITH HOLD or WITHOUT HOLD is not specified when a cursor is created, WITHOUT HOLD is used by default.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. This parameter can be set only in a session.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is **on**, all cursors are not released, which may cause memory bloat in sessions.

modify_function_property

Parameter description: Specifies the attributes of related system functions in non-M-compatible mode.

Parameter type: string

Unit: none

Value range: Currently, only the configuration items listed in **Table 6-45** are supported. Compatibility configuration items are separated by commas (,).

Default value: **"3257,5562,4164,4073"** for a newly installed database. If the GUC parameter does not exist in the basic version to be upgraded, the default value is **""**. If the GUC parameter exists in the basic version to be upgraded, the original setting is retained.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: The attributes of system functions are affected.

Table 6-45 System function configuration items

OID Corresponding to the System Function	Description
3257	Modifies the variability of the pg_catalog.current_timestamp(numeric) function. When the configuration item exists in the GUC parameter, the variability of the function is **'stable'**. When the configuration item does not exist, the variability of the function is **'immutable'**.
4073	Modifies the variability of the pg_catalog.text_timestamp(text) function. When the configuration item exists in the GUC parameter, the variability of the function is **'stable'**. When the configuration item does not exist, the variability of the function is **'immutable'**.
4164	Modifies the variability of the pg_catalog.text_date(text) function. When the configuration item exists in the GUC parameter, the variability of the function is **'stable'**. When the configuration item does not exist, the variability of the function is **'immutable'**.
5562	Modifies the variability of the pg_catalog.DBTimezone() function. When the configuration item exists in the GUC parameter, the variability of the function is **'stable'**. When the configuration item does not exist, the variability of the function is **'immutable'**.

enable_check_insertall_target

Parameter description: Specifies whether to check the validity of the target table attributes of the INSERT ALL syntax in ORA-compatible mode.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The validity of the target table attributes of the INSERT ALL syntax is checked. The target table cannot be a replication table or a table with a global secondary index (GSI). The distribution modes of all target tables must be the same.
- **off**: The validity of the target table attributes of the INSERT ALL syntax is not checked.

Default value: **on** for a newly installed database. If the GUC parameter does not exist in the basic version to be upgraded, the default value is **off**. If the GUC

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

parameter exists in the basic version to be upgraded, the original setting is retained.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **off**, the data inserted into the target table may not meet the expectation.

forbid_hold_cursor_with_forupdate

Parameter description: Specifies whether to report an error when the cursor cyclic or for loop query statement with the FOR UPDATE/FOR SHARE feature in a stored procedure accesses the cursor in the next cycle after the transaction is committed or rolled back.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The error reporting behavior described in the parameter description is reported.
- **off**: The error reporting behavior described in the parameter description is not reported.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **off**, an error may be reported when the stored procedure is executed during transaction commit/rollback, or the FOR UPDATE/FOR SHARE feature may be lost when the cursor is accessed cyclically next time after the commit/rollback.

NOTE

If the current database version is directly installed, the default value of this parameter is **on**. For a database upgraded from an earlier version, if the source version does not support this parameter, the default value of this parameter is **off** after the upgrade. If the source version supports this parameter, the current setting of this parameter is retained after the upgrade.

adjust_systemview_priority

Parameter description: Specifies whether to preferentially access the user-defined view with the same name in Oracle-compatible mode.

Parameter type: Boolean.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: enabled. The user-defined views with the same name are preferentially accessed.
- **off**: disabled. The system views with the same name are preferentially accessed.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **on**, the user-defined views with the same name are preferentially accessed.

using_danger_feature

Parameter description: Specifies whether to enable dangerous features.

Parameter type: string.

Unit: none

Value range: Only the information in **Table 6-46** is supported. Use commas (,) to separate multiple values.

Default value: **"none"**, indicating that dangerous features are disabled.

Table 6-46 Dangerous feature configuration items

Dangerous Feature Configuration Item	Behavior Control
operator_class	If this option is set, system administrators are allowed to create operator classes. When an operator class is created, the input parameter type of the specified function must be the same as the index type or the type can be properly converted. Otherwise, unpredictable consequences may occur.
alter_function	• If this option is set, you can modify the names, owners, and schemas of functions in m_schema, information_schema, sqladvisor, sys, pg_toast, dbe_sql_util, and the schemas to which advanced packages belong. • If this option is not set, the preceding modification is not allowed.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after you fully understand the parameter meaning and test the parameter to avoid unexpected results.

NOTE

- If the current database version is directly installed, the default value of this parameter is **"none"**. For an upgrade from an earlier version to the current version, if the source version does not support this parameter, the default value of this parameter is **"operator_class, alter_function"**. If the source version supports this parameter, the current setting of this parameter is retained after the upgrade.
- When **"none"** is set together with other configuration items, **"none"** does not take effect.

disable_operator

Parameter description: Specifies the operators to be disabled. The operators specified by this parameter cannot be called. This functionality can be used when a system operator cannot meet user requirements and a user-defined operator is required.

Parameter type: string

Unit: none

Value range: a string consisting of multiple operator OIDs separated by commas (,). You can obtain the operator OID by querying the **PG_OPERATOR** system catalog.

Default value: **""**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after you fully understand the parameter meaning and test the parameter to avoid unexpected results.

enable_xmltype_ignore_format

Parameter description: Specifies whether to use the content format for XMLType data verification.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The parameter is enabled and XMLType data can be verified in content format.
- **off**: The parameter is disabled and XMLType data can be verified in document format.

Default value: **off**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is reserved for emergency. Do not set it unless necessary.

Risks and impacts of improper settings: Change the parameter value after you fully understand the parameter meaning and test the parameter to avoid unexpected results.

NOTE

When the **enable_xmltype_ignore_format** parameter is enabled, the content format can be used to verify input data only in the following scenarios:

- Convert the unknown and unspecified types to the XMLType type. For details about the conversion principle, see **Type Conversion**.
- Call the XMLType function, for example, xmltype('123').

enable_force_create_obj

Parameter description: Specifies whether to enable the one-off import function for views. After this parameter is enabled, a view can still be successfully created even if the objects it depends on are absent or the required query permissions are missing.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The one-off import function is enabled for views.
- **off**: The one-off import function is disabled for views.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the actual scenario. You are advised to enable the corresponding function in normal cases.

Risks and impacts of improper settings: Change the parameter value after you fully understand the parameter meaning and test the parameter to avoid unexpected results.

distribute_default_compatibility

Parameter description: Specifies the default compatibility mode when the **DBCOMPATIBILITY** and **TEMPLATE** keywords are not specified in the CREATE DATABASE operation.

Parameter type: enumerated type

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **ora**: The default mode for creating a database is ORA-compatible.
- **mysql**: The default mode for creating a database is MySQL-compatible.

Default value:

- For a newly installed database: **ora**.
- For a database upgraded from a version earlier than 507.0.0: **mysql**.
- For a database upgraded from 507.0.0 or later: original parameter value.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements. This parameter can be specified during database installation. If the specified value is **NULL**, an error will be reported. If the specified value is not within the value range, the default value **ora** will be used. You can set this parameter during installation as follows:

```
gs_install -X /data/software/XmlConfig/Dist_3C3D3REP.xml --init-password --gsinit-parameter="--distribute_default_compatibility ora"
```

Risks and impacts of improper settings: Change the parameter value after you fully understand the parameter meaning and test the parameter to avoid unexpected results.

load_data_behavior_compat_options

Parameter description: Specifies the behavior of importing data using the LOAD DATA syntax. This database configuration item is used when **sql_compatibility** is set to **'MYSQL'** or in the M-compatible database.

Parameter type: enumerated type

Unit: none

Value range: See **Table 6-47**.

Table 6-47 Configuration items of load_data_behavior_compat_options

Configuration Item	Behavior
none	No compatibility configuration item is used. When **none** is used together with other compatibility configuration items, **none** does not take effect. For example, **SET load_data_behavior_compat_options='none,new_escape_string'** is equivalent to **SET load_data_behavior_compat_options='new_escape_string'**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
new_escape_string	Specifies the escape sequences supported when LOAD DATA is used to import data. The default escape character is a backslash (\). ● If the **new_escape_string** option is not set, the following escape sequences are supported: – **\a[b[c]]**: **a**, **b**, and **c** in the escape sequence are octal digits (0 to 7), indicating a value consisting of octal digits. The system parses the value into the corresponding byte value. For example, the octal value of **\141** is 141, which corresponds to the decimal value 97, indicating the ASCII character (the ASCII code of character **a** is 97). – **\xhh**: In the escape sequence, **h** indicates a hexadecimal digit (0 to 9, a to f, or A to F). **hh** indicates a value consisting of two hexadecimal digits. The system parses the value into the corresponding byte value. For example, the hexadecimal value of **\x6a** is **6A**, which corresponds to the decimal value **106**, indicating the ASCII character **'j'** (the ASCII code of character **'j'** is 106). – **\r**: carriage return character. – **\n**: newline character. – **\b**: backspace character. – **\v**: vertical tab character. – **\t**: horizontal tab character. – **\f**: form feed character. – **\0**: zero character. ● If the **new_escape_string** option is set, the following escape sequences are supported: – **\r**: carriage return character. – **\n**: newline character. – **\b**: backspace character. – **\t**: horizontal tab character. – **\0**: zero character. – **\Z**: ASCII 26(Control Z). Specifies whether to escape the specified line terminator when LOAD DATA is used to import data. ● If **new_escape_string** is not set and the line terminator is specified as follows, it will be escaped to the corresponding control character: – **0x0A** or **\n**: newline character. – **0x0D0A** or **\r\n**: carriage return character and newline character. – **0x0D** or **\r**: carriage return character.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
	– **0x09** or **\t**: horizontal tab character. – **0x0B** or **\v**: vertical tab character. – **0x0C** or **\f**: form feed character. • If the **new_escape_string** option is set, the specified line terminator is not escaped.

Configuration Item	Behavior
fixed_row_column_division	When LOAD DATA is used to import data, you can use this option to adjust the row and column splitting behavior of the source data file. The processing methods in the following scenarios are controlled by this option: ● When the end of a line in the source data file does not contain a line separator (**lines_terminated_by**) and the data in the last column of the last line starts with an enclosure character (**enclosed_by**): – If this option is not set, one byte will be lost during the import. – If this option is set, no data will be lost during the import. ● When the escape character is followed by a column delimiter (**fields_terminated_by**): – If this option is not set, the column delimiter will not be escaped to a common character. – If this option is set, the column delimiter will be escaped to a common character. ● When the escape character is followed by an enclosure: – If this option is not set, the enclosure will not be escaped to a common character. – If this option is set, the enclosure will be escaped to a common character. ● When the escape character is followed by an unspecified line separator: – If this option is not set, the line separator will not be escaped to a common character. – If this option is set, the line separator will be escaped to a common character. ● When no line separator is specified: – If this option is not set, the line separator is the first matched **'\r\n'**, **'\n'**, or **'\r'** in the file. – If this option is set, the line separator is **'\n'**. ● When the escape character and the enclosure are the same character: – If this option is not set, escape sequences such as **'\t'** and **'\n'** will be escaped. – If this option is set, escape sequences such as **'\t'** and **'\n'** will not be escaped. ● In M-compatible mode, when the **LOCAL** option is specified for remote transmission: – If this option is not set, the specified line starting character (**lines_starting_by**) will not take effect. If the source data

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior
	contains a whole line of the '\.' string, '\.' and the subsequent data will not be imported. – If this option is set, the specified line starting character will take effect. If the source data contains a whole line of the '\.' string, '\.' and the subsequent data will be imported.

Default value:

- For a newly installed database, the value is **"new_escape_string, fixed_row_column_division"**.
- For a database upgraded from an earlier version, if the earlier version does not support this GUC parameter, the default value is **"none"** after the upgrade. If the earlier version supports this GUC parameter, the default value is retained after the upgrade.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after you fully understand the parameter meaning and test the parameter to avoid unexpected results.

modify_function_property_json

Parameter description: Specifies the properties of a system function in JSON format.

- **oid**: OID of the target system function. The value ranges from 0 to 10000.
- **compatibility**: database compatibility mode. The value can be **A**, **M**, or **OTHER**.
- **property_list**: function property list. Multiple properties can be configured.
  - **property_name**: property name. Currently, **proshippable** and **provolatile** are supported.
  - **property_value**: property value. The value must be within the value range of the corresponding property. The value range of **proshippable** is **t** and **f**, and the value range of **provolatile** is **i**, **s**, and **v**.

Parameter type: string

Unit: none

Value range: 0 to 50,000 characters for configuration file modification; 0 to 4,099 characters for gs_guc setting.

Default value:

- For a newly installed database:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

'[{"oid": 4162,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 4163,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 4186,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 4187,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 4905,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 4906,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 4907,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 4908,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 4909,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 4910,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 4911,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 4912,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 4913,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 4914,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 6286,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 6287,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 6288,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 6289,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "s"}]},
{"oid": 6307,"compatibility":"A", "property_list": [{"property_name": "provolatile", "property_value": "v"}]},
{"oid": 4079,"compatibility":"M", "property_list": [{"property_name": "provolatile", "property_value": "s"}]}]'.

- If the GUC parameter does not exist in the source version, the default value is "".
- If the GUC parameter exists in the source version, retain the original setting.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: The attributes of system functions are affected.

NOTE

If the values of **modify_function_property** and **modify_function_property_json** conflict, the value of **modify_function_property** prevails.

illegal_character_strict_check

Parameter description: Specifies whether to enable the check for invalid characters.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Invalid characters are checked. If the byte sequence of a character string cannot be completely parsed into valid characters based on the encoding rules (that is, the actual byte length is insufficient to form a complete character), the character string is considered invalid, and an error is reported.
- **off**: Invalid characters are not checked.

Default value:

- For a newly installed database: **on**.
- For a database upgraded from an earlier version that does not support this GUC parameter, the default value is **off** after the upgrade. If the earlier version supports this GUC parameter, the default value is retained after the upgrade.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Do not set this parameter unless necessary.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

disable_unsupported_copy_load_options

Parameter description: Specifies the behavior of handling operations that may cause exceptions in the non-recommended syntax COPY/LOAD DATA during data import and export.

Parameter type: Boolean.

Value range:

- **on**: Operations that may cause exceptions in the non-recommended syntax are disabled.
- **off**: Operations that may cause exceptions in the non-recommended syntax are not disabled.

Default value:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- For a newly installed database: **on**.
- For a database upgraded from an earlier version that does not support this GUC parameter, the default value is **off** after the upgrade. If the earlier version supports this GUC parameter, the default value is retained after the upgrade.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.16.3 Product Version of the Cloud Service

This section describes the version parameters of the cloud service.

product_version

Parameter description: Specifies the version of a cloud service product.

Parameter type: string.

Unit: none

Value range: none

Default value: ""

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: The following rules must be met to avoid errors.

- The value can contain a maximum of 50 characters.
- The value cannot contain vertical bars (|), semicolons (;), ampersands (&), dollar signs ($), ampersands (>), ampersands (<), ampersands (`), backslashes (\), exclamation marks (!), or newline characters.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

hotpatch_version

Parameter description: Specifies the hot patch version of a cloud service product.

Parameter type: string.

Unit: none

Value range: none

Default value: ""

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: The following rules must be met to avoid errors.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- The value can contain a maximum of 1500 characters.
- The value cannot contain vertical bars (|), semicolons (;), ampersands (&), dollar signs ($), ampersands (>), ampersands (<), ampersands (`), backslashes (\), exclamation marks (!), or newline characters.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.17 Fault Tolerance

This section describes parameters used for controlling how the server processes an error occurring in the database system.

exit_on_error

Parameter description: If this function is enabled, errors of the ERROR level will be escalated to PANIC errors, and core stacks will be generated. It is mainly used to locate problems and test services.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Errors of the ERROR level will be escalated to PANIC errors.
- **off**: Errors of the ERROR level will not be escalated.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

restart_after_crash

Parameter description: If this parameter is enabled and a backend thread crashes, GaussDB automatically reinitializes the backend thread.

Parameter type: Boolean

Unit: none

Value range:

- **on**: GaussDB can automatically reinitialize a backend thread when the backend thread breaks down. In some circumstances (for example, when a management tool, such as xCAT, is used to manage GaussDB), setting this parameter to **on** maximizes the availability of the database.
- **off**: A management tool is enabled to obtain control permission and take proper measures when a backend thread crashes.

Default value: **on**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

omit_encoding_error

Parameter description: If this parameter is enabled and the client character set of the database is encoded in UTF-8 format, character encoding conversion errors will be recorded in logs. Additionally, converted characters that have conversion errors will be ignored and replaced by question marks (?).

Parameter type: Boolean

Unit: none

Value range:

- **on**: Characters that have conversion errors will be ignored and replaced by question marks (?), and error information will be recorded in logs.
- **off**: Characters that have conversion errors cannot be converted and error information will be directly displayed.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

If this parameter is modified by running the **gs_guc reload** command and the connection of a session on the current node is not from the client but from another node in the cluster to which the node belongs, this parameter does not take effect immediately on the session after the **gs_guc reload** command is executed. The setting takes effect only after the connection node is disconnected and then reconnected.

cn_send_buffer_size

Parameter description: Specifies the size of the data buffer used for data transmission on CNs.

Parameter type: integer.

Unit: KB

Value range: 8–128

Default value: **8**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **100** without a unit,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

it indicates 100 KB. If this parameter is set to **128kB**, it indicates 128 KB. The unit must be KB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_cn_temp_file_size

Parameter description: Specifies the maximum size of temporary files used by the CN when the SQL statement automatic retry function is enabled. The value **0** indicates that temporary files are not used.

Parameter type: integer.

Unit: KB

Value range: 0 to 10485760

Default value: 5242880 (that is, 5 GB)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **100** without a unit, it indicates 100 KB. If this parameter is set to **16MB**, it indicates 16 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

data_sync_retry

Parameter description: Specifies whether to keep running the database when updated data fails to be written into disks by using the fsync function. In some OSs, no error is reported even if **fsync** fails after the second attempt. As a result, data is lost.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The retry mechanism is used to continue running the database after fsync fails to synchronize data to the disk.
- **off**: A PANIC-level error is reported and the database is stopped after fsync fails.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

remote_read_mode

Parameter description: Specifies whether to enable the remote read function. This function allows pages on the standby node to be read when reading pages on the primary node fails.

Parameter type: enumerated type

Unit: none

Value range: **off**, **non_authentication**, and **authentication**

- **off**: The remote read function is disabled.
- **non_authentication**: The remote read function is enabled but certificate authentication is not required.
- **authentication**: The remote read function is enabled and certificate authentication is required.

Default value: **authentication**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.18 Connection Pool Parameters

When a connection pool is used to access the database, database connections are established and then stored in the memory as objects during system running. When you need to access the database, no new connection is established. Instead, an existing idle connection is selected from the connection pool. After you finish accessing the database, the database does not disable the connection but puts it back into the connection pool. The connection can be used for the next access request.

pooler_port

Parameter description: Specifies the O&M management port of internal tools, such as cm_agent and cm_ctl. This port is used by the initial user or system administrator to connect to the database through the client.

Parameter type: integer.

Unit: none

Value range: 1 to 65535.

Default value: **5433** (The actual value is the specified value of the **port** GUC parameter plus 1.)

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

pooler_maximum_idle_time

Parameter description: Specifies the maximum amount of time that the connections can remain idle in a pool before being removed. After that, the automatic connection clearing mechanism is triggered to reduce the number of connections on each node to the value of **minimum_pool_size**.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647

Default value: **600** (that is, 10 minutes)

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **100** without a unit, it indicates 100s. If this parameter is set to **20min**, it indicates 20 minutes. If the unit is required, the value must be **s**, **min**, **h**, or **d**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the frequency of clearing idle connections in the pooler connection pool decreases and the recycling rate decreases. If the value is too small, the frequency of clearing idle connections in the pooler connection pool increases. As a result, services frequently establish connections, affecting performance.

minimum_pool_size

Parameter description: Specifies the minimum number of remaining connections on the node corresponding to each pooler connection pool after automatic pooler connection cleanup. This parameter is used by the automatic pooler connection cleanup function.

Parameter type: integer.

Unit: none

Value range: 1 to 65535

Default value: **50**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the number of clearing idle connections in the pooler connection pool decreases and the recycling rate decreases. If the value is too small, the number of clearing idle connections in the pooler connection pool increases. As a result, services frequently establish connections, affecting performance.

max_pool_size

Parameter description: Specifies the maximum number of connections between the connection pool and another CN/DN. You may need to adjust this parameter when the cluster scale changes, for example, when nodes are added or removed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer

Unit: none

Value range: 0 to 65535. This parameter does not take effect on DNs and can be set to **0** only on DNs.

Default value:

- Independent deployment:

  CN: **32768** (128-core CPU/1024 GB memory, 60-core CPU/480 GB memory); **16384** (32-core CPU/256 GB memory); **8192** (16-core CPU/128 GB memory); **4096** (8-core CPU/64 GB memory); **2048** (4-core CPU/32 GB memory); **1000** (4-core CPU/16 GB memory)

  DN: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the default value of different instance specifications. The value of this parameter must be greater than the value of **max_connections**. During the adjustment, the connections consumed by internal threads must be reserved.

Risks and impacts of improper settings: When the number of concurrent services is large, the connections from the CN to other CNs/DNs in the connection pool are consumed. If this parameter is set to a small value, an error is reported when the number of connections reaches the upper limit. As a result, services fail. When a CN is started, it allocates memory in advance based on the parameter value. Therefore, if the parameter value increases, the system consumes more memory resources. However, in general, the impact on the CN memory is small.

persistent_datanode_connections

Parameter description: Specifies whether to release the connection for the current session.

Parameter type: Boolean

Unit: none

Value range:

- **on** indicates that the connection for the current session will not be released.
- **off** indicates that the connection for the current session will be released.

Default value: **off**

Setting method: This is a BACKEND parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this parameter is enabled, a session may hold a connection but does not run a query. As a result, other queries cannot apply for a connection and an error is reported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

max_coordinators

Parameter description: Specifies the maximum number of CNs in a cluster. This parameter may need to be modified when the cluster scale changes, for example, nodes are added or deleted. During scale-out, ensure that the value of this parameter is greater than the number of CNs in the target cluster. Otherwise, the scale-out will fail. If this parameter is set to a large value during scale-in, more memory resources are consumed when the CN is started. However, in general, the impact on the CN memory is small.

Parameter type: integer.

Unit: none

Value range: 2 to 1024

Default value: **128**

Setting method: This is a POSTMASTER parameter. You are advised not to modify it. If you need to modify it, set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the actual cluster specifications.

Risks and impacts of improper settings: If this parameter is set to a value less than the number of CNs in the cluster, node creation will fail. When a CN is started, it allocates memory in advance based on the parameter value. Therefore, if the parameter value increases, the system consumes more memory resources. However, in general, the impact on the CN memory is small.

max_datanodes

Parameter description: Specifies the maximum number of DNs in a cluster. This parameter may need to be modified when the cluster scale changes, for example, nodes are added or deleted. During scale-out, ensure that the value of this parameter is greater than the total number of DN shards in the target cluster. Otherwise, the scale-out will fail. If this parameter is set to a large value during scale-in, more memory resources are consumed when the CN is started. However, in general, the impact on the CN memory is small.

Parameter type: integer.

Unit: none

Value range: 2 to 65535

Default value: **256**

Setting method: This is a POSTMASTER parameter. You are advised not to modify it. If you need to modify it, set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the actual cluster specifications.

Risks and impacts of improper settings: If this parameter is set to a value less than the number of DNs in the cluster, node creation will fail. When a CN is started, it allocates memory in advance based on the parameter value. Therefore, if the parameter value increases, the system consumes more memory resources. However, in general, the impact on the CN memory is small.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

cache_connection

Parameter description: Specifies whether to recycle the connections of a connection pool.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The connections of a connection pool will be recycled.
- **off**: The connections of a connection pool will not be recycled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the connections held by the session are cleared after the session execution statement ends. As a result, the service frequently establishes connections, affecting the performance. After this parameter is enabled in the multi-database scenario, if the memory size is small and the value of **max_connections** is large, the memory may be insufficient and DNs cannot be started.

enable_force_reuse_connections

Parameter description: Specifies whether a session forcibly reuses a new connection.

Parameter type: Boolean

Unit: none

Value range:

- **on** indicates that the new connection is forcibly used.
- **off** indicates that the current connection is used.

Default value: **off**

Setting method: This is a BACKEND parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this parameter is enabled, the **reset** parameter is forcibly sent to existing connections each time before a statement is executed, which affects the performance.

pooler_connect_max_loops

Parameter description: Specifies whether to enable the connection retries to enhance stability of setting up connections in switchover scenarios. If a service fails to connect to the primary node, it will retry by attempting to connect to the standby node. If the standby node is successfully promoted to primary, the retry

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

attempt will succeed. This parameter is used by the pooler connection retry function. This parameter specifies the total number of retry attempts. If this parameter is set to **0**, retry is disabled. The service only establishes a connection to the primary node.

Parameter type: integer.

Unit: none

Value range: 0 to 20

Default value: **1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: This parameter is an HA parameter. If this parameter is set to **0**, the retry function is disabled, reducing the HA capability. If this parameter is set to a value greater than 1, the number of retries increases.

pooler_connect_interval_time

Parameter description: Specifies the interval between retries when **pooler_connect_max_loops** is set to a value greater than 1. This parameter is used by the pooler connection retry function. You are advised to set this parameter to a value slightly greater than the time required for switchover in the current cluster.

Parameter type: integer.

Unit: seconds

Value range: 0 to 7200

Default value: **15**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **100** without a unit, it indicates 100s. If this parameter is set to **20min**, it indicates 20 minutes. If the unit is required, the value must be **s**, **min**, **h**, or **d**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: This parameter is an HA parameter. If this parameter is set to a large value, the retry period is prolonged and the HA capability is reduced.

pooler_timeout

Parameter description: Specifies the timeout interval of communication between each connection in a CN's connection pool and another CN/DN.

Parameter type: integer.

Unit: seconds

Value range: 0 to 7200

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **600** (that is, 10 minutes)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **100** without a unit, it indicates 100s. If this parameter is set to **20min**, it indicates 20 minutes. If the unit is required, the value must be **s**, **min**, **h**, or **d**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, an error is reported during data transmission and reception when the network fluctuates. If the value is too large, an error is reported during data transmission and reception when the network is disconnected. As a result, the HA capability is reduced.

pooler_connect_timeout

Parameter description: Specifies the timeout interval of connecting a CN's connection pool to another CN/DN in the same cluster.

Parameter type: integer.

Unit: seconds

Value range: 0 to 7200

Default value: **60** (1 minute)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **100** without a unit, it indicates 100s. If this parameter is set to **20min**, it indicates 20 minutes. If the unit is required, the value must be **s**, **min**, **h**, or **d**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, an error is reported during connection setup when the network fluctuates. If the value is too large, an error is reported during connection setup when the network is disconnected. As a result, the HA capability is reduced.

pooler_cancel_timeout

Parameter description: Specifies the timeout interval of canceling a connection by a CN's connection pool during error processing. If similar timeout occurs when an exception of the subtransaction or stored procedure is captured, the transaction containing the subtransaction or the stored procedure rolls back. If the source data from the COPY FROM operation is not consistent with that of the table definition in the target table, and the parameter value is not **0**, an error is reported.

Parameter type: integer.

Unit: seconds

Value range: 0 to 7200. **0** (not recommended) indicates that this function is disabled and no timeout limit is set.

Default value: **15**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **100** without a unit, it

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

indicates 100s. If this parameter is set to **20min**, it indicates 20 minutes. If the unit is required, the value must be **s**, **min**, **h**, or **d**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, an error is reported for cancel actions when the network fluctuates. If the value is too large, an error is reported for cancel actions when the network is disconnected. As a result, the HA capability is reduced.

6.3.19 Cluster Transaction Parameters

This section describes the settings and value ranges of transaction parameters for the cluster.

transaction_isolation

Parameter description: Specifies the isolation level of the current transaction. For a new transaction that is started, the isolation level is fixed to the value of **default_transaction_isolation**.

Parameter type: string.

Unit: none

Value range: The value is a string of case-sensitive and space-sensitive characters. Only the following values are supported:

- **"serializable"**: Currently, GaussDB does not support this isolation level. Setting this isolation level has the same effect as **"repeatable read"**.
- **"read committed"**: Only data of committed transactions is read. Uncommitted data cannot be read.
- **"read uncommitted"**: The performance is the same as that of **"read committed"**.
- **"repeatable read"**: Only the data committed before a transaction starts is read. Uncommitted data or data committed in other concurrent transactions cannot be read.
- **"default"**: The value is the same as that of **default_transaction_isolation**.

Default value: **"default"**. If the default value of **default_transaction_isolation** is **"read committed"**, the output of the following command is **"read committed"**:

```
show transaction_isolation;
```

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

transaction_read_only

Parameter description: Specifies whether the current transaction is a read-only transaction.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The current transaction is a read-only transaction.
- **off**: The current transaction is a read/write transaction.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

autocommit

Parameter description: Specifies whether to enable the automatic commit mode. If this parameter is set to **on**, sessions will be committed automatically and each individual statement is implicitly committed. If this parameter is set to **off**, you must run COMMIT to commit a transaction or run ROLLBACK to cancel a transaction. If **autocommit** is changed from **off** to **on**, all unfinished transactions are automatically committed.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The automatic commit mode is enabled.
- **off**: The automatic commit mode is disabled.

Default value: **on**

Setting method: This parameter is a USERSET parameter. You can set it based on instructions provided in **Table 6-1**. The gs_guc tool cannot be used to set this parameter.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **off**, DDL or DCL statements executed in a transaction will not be automatically committed.

NOTE

- When **dbcompatibility** is set to **'M'**, this parameter can be set to **off**. Otherwise, this parameter is set to **on**.
- In GaussDB, this parameter cannot be queried in the pg_settings view. You can run the **SHOW autocommit;** command to view the value of this parameter.
- If the **autocommit** parameter is used, the internal automatic commit mode must be enabled. You can run **\echo :AUTOCOMMIT** to check whether the internal automatic commit mode is enabled. If the automatic commit mode is not used, run **\set AUTOCOMMIT on** to enable the automatic commit mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

xc_maintenance_mode

Parameter description: Specifies whether the system is in maintenance mode.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The maintenance mode is used.
- **off**: The maintenance mode is not used.

NOTICE

Exercise caution when enabling this function. You need to avoid data inconsistencies in the cluster.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on method 3 provided in **Table 6-2**.

Setting suggestion: This parameter is used by maintenance personnel to locate faults. You are advised not to enable this parameter.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

allow_concurrent_tuple_update

Parameter description: Specifies whether to allow concurrent update.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Concurrent updates are allowed.
- **off**: Concurrent updates are not allowed.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_host

Parameter description: Specifies the IP address of the primary GTM process. This parameter is visible only to the SYSADMIN user. The IP address can be an IPv4 or IPv6 address.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: string.

Unit: none

Value range: a string

Default value: IP address of the primary GTM.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_port

Parameter description: Specifies the listening port of the primary GTM process. This parameter is visible only to the SYSADMIN user.

NOTE

This parameter is specified in the configuration file during installation. Do not modify this parameter unless absolutely necessary. Otherwise, database communication will be affected.

Parameter type: integer.

Unit: none

Value range: 1 to 65535

Default value: specified during installation.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the port number is changed, the database communication will be affected.

gtm_host1

Parameter description: Specifies the IP address of the standby GTM process. This parameter is visible only to the SYSADMIN user. The IP address can be an IPv4 or IPv6 address.

Parameter type: string.

Unit: none

Value range: a string

Default value: IP address of the standby GTM.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_port1

Parameter description: Specifies the listening port of the standby GTM process. This parameter is visible only to the SYSADMIN user.

NOTE

This parameter is specified in the configuration file during installation. Do not modify this parameter unless absolutely necessary. Otherwise, database communication will be affected.

Parameter type: integer.

Unit: none

Value range: 1 to 65535

Default value: The value is specified during installation if the standby node 1 is deployed. Otherwise, the default value is **6665**.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the port number is changed, the database communication will be affected.

gtm_backup_barrier

Parameter description: Specifies whether to create a restoration point for the GTM starting point.

Parameter type: Boolean

Unit: none

Value range:

- **on**: A restoration point is created.
- **off**: No restoration point is created.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_conn_check_interval

Parameter description: Sets the intervals between two consecutive performed checks on the connections between local threads and the primary GTM for the CN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483

Default value: **10**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_show_any_tuples

Parameter description: This parameter is available only in a read-only transaction and is used for analysis. When this parameter is set to **on**, all versions of tuples in the table are displayed.

Parameter type: Boolean

Unit: none

Value range:

- **on**: All versions of tuples in the table are displayed.
- **off**: No versions of tuples in the table are displayed.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

In the TOAST scenario, if DML operations (mainly INSERT+UPDATE or INSERT+DELETE) are performed before and after the REINDEX operation, and the read-only transaction is started and the GUC parameter is enabled after the REINDEX operation, historical data in the TOAST table or TOAST index table can be queried separately in the released version. However, when historical data in the TOAST column is queried in the main table, the error message "missing chunk number xxx" is displayed.

gtm_connect_timeout

Parameter description: Specifies the GTM connection timeout. If the connection time of the GTM exceeds its value, the connection times out and exits.

Parameter type: integer.

Unit: seconds

Value range: 1 to 2147483647

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **2**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_connect_retries

Parameter description: Specifies the number of GTM reconnection attempts.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483647

Default value: **30**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_rw_timeout

Parameter description: Specifies the GTM response timeout. If the time spent waiting for GTM responses exceeds its value, the operation times out and exits.

Parameter type: integer.

Unit: seconds

Value range: 1 to 2147483647

Default value: **60** (1 minute)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_redistribute

Parameter description: Specifies whether unmatched nodes are redistributed.

Parameter type: Boolean

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on** indicates that unmatched nodes are redistributed.
- **off** indicates that unmatched nodes are not redistributed.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

replication_type

Parameter description: Specifies whether the current database is deployed in standalone or one-primary-multiple-standby mode.

Parameter type: integer.

Unit: none

Value range: 0 to 3

- **0**: reserved parameter.
- **1**: one-primary-multiple-standby mode, covering all scenarios. This mode is recommended.
- **2**: single primary mode. In this mode, the standby node cannot be expanded.
- **3**: reserved parameter.

Default value: **1**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. This parameter is used for CM deployment and cannot be set.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

pgxc_node_name

Parameter description: Specifies the name of a node.

When a standby node requests to replicate logs on the primary node, if the **application_name** parameter is not set, the **pgxc_node_name** parameter is used as the name of the streaming replication slot of the standby node on the primary node. The streaming replication slot is named in the following format: *Value of this parameter_IP address of the standby node_Port number of the standby node*. The IP address and port number of the standby node are obtained from the IP address and port number of the standby node specified by the **replconninfo** parameter. The maximum length of a streaming replication slot name is 61 characters. If the length of the concatenated string exceeds 61 characters, the truncated **pgxc_node_name** will be used for concatenation to ensure that the length of the streaming replication slot name is less than or equal to 61 characters.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: string.

Unit: none

Value range: a string

Default value: current node name.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After the modification, the cluster will fail to be connected.

enable_twophase_commit

Parameter description: Currently, the cloud database uses the GTM-free mode to solve the SDS replacement problem. To prevent unreliability caused by service abuse, the GUC parameter **enable_twophase_commit** is used to enable or disable distributed two-phase write transactions.

Parameter type: Boolean

Unit: none

Value range:

- **on** indicates that distributed two-phase commit is allowed in the GTM-free mode.
- **off** indicates that distributed two-phase commit is not allowed in the GTM-free mode.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

application_type

Parameter description: valid only when **enable_gtm_free** is set to **on**. This parameter specifies the service type of a user.

Parameter type: enumerated type.

Unit: none

Value range:

- **not_perfect_sharding_type** indicates a service across nodes. If this value is used, statements across nodes can be executed.
- **perfect_sharding_type** indicates a service on a single node. If an SQL statement requires multiple nodes, an error is reported and the corresponding

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

SQL statement is recorded in the system log. If this value is used, you can run the **/*+ multinode */ hint** command to allow SQL statements to be executed on multiple nodes. The multinode hint can be added after the select, insert, update, delete, and merge keywords.

Default value: **perfect_sharding_type**

Setting method: This parameter is a USERSET parameter and cannot be set using gs_guc. It can be set using either of the following methods:

- Use the **gsql** client to perform session-level configuration.
- When JDBC is used to connect to the database, set the **ApplicationType** parameter for the connection string.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_host2

Parameter description: Specifies the IP address of the GTM 2 if the standby GTM 2 is deployed. This parameter is visible only to the SYSADMIN user. The IP address can be an IPv4 or IPv6 address.

Parameter type: string.

Unit: none

Value range: a string

Default value: IP address of the GTM 2 if the GTM 2 is deployed. Otherwise, the value is **""**.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_host3

Parameter description: Specifies the IP address of the GTM 3 if the standby GTM 3 is deployed. This parameter is visible only to the SYSADMIN user. The IP address can be an IPv4 or IPv6 address.

Parameter type: string.

Unit: none

Value range: a string

Default value: IP address of the GTM 3 if the GTM 3 is deployed. Otherwise, the value is **""**.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_host4

Parameter description: Specifies the IP address of the GTM 4 if the standby GTM 4 is deployed. This parameter is visible only to the SYSADMIN user. The IP address can be an IPv4 or IPv6 address.

Parameter type: string.

Unit: none

Value range: a string

Default value: IP address of the GTM 4 if the GTM 4 is deployed. Otherwise, the value is **""**.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_host5

Parameter description: Specifies the IP address of the GTM 5 if the standby GTM 5 is deployed. This parameter is visible only to the SYSADMIN user. The IP address can be an IPv4 or IPv6 address.

Parameter type: string.

Unit: none

Value range: a string

Default value: IP address of the GTM 5 if the GTM 5 is deployed. Otherwise, the value is **""**.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_host6

Parameter description: Specifies the IP address of the GTM 6 if the standby GTM 6 is deployed. This parameter is visible only to the SYSADMIN user. The IP address can be an IPv4 or IPv6 address.

Parameter type: string.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: a string

Default value: IP address of the GTM 6 if the GTM 6 is deployed. Otherwise, the value is **""**.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_host7

Parameter description: Specifies the IP address of the GTM 7 if the standby GTM 7 is deployed. This parameter is visible only to the SYSADMIN user. The IP address can be an IPv4 or IPv6 address.

Parameter type: string.

Unit: none

Value range: a string

Default value: IP address of the GTM 7 if the GTM 7 is deployed. Otherwise, the value is **""**.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_port2

Parameter description: Specifies the listening port of the standby GTM 2 if the standby GTM 2 is deployed. This parameter is visible only to the SYSADMIN user.

Parameter type: integer.

Unit: none

Value range: 1 to 65535

Default value: The value is specified during installation if the GTM 2 is deployed. Otherwise, the value is **6666**.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the port number is changed, the database communication will be affected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

gtm_port3

Parameter description: Specifies the listening port of the standby GTM 3 if the standby GTM 3 is deployed. This parameter is visible only to the SYSADMIN user.

Parameter type: integer.

Unit: none

Value range: 1 to 65535

Default value: The value is specified during installation if the GTM 3 is deployed. Otherwise, the value is **6666**.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the port number is changed, the database communication will be affected.

gtm_port4

Parameter description: Specifies the listening port of the standby GTM 4 if the standby GTM 4 is deployed. This parameter is visible only to the SYSADMIN user.

Parameter type: integer.

Unit: none

Value range: 1 to 65535

Default value: The value is specified during installation if the GTM 4 is deployed. Otherwise, the value is **6666**.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the port number is changed, the database communication will be affected.

gtm_port5

Parameter description: Specifies the listening port of the standby GTM 5 if the standby GTM 5 is deployed. This parameter is visible only to the SYSADMIN user.

Parameter type: integer.

Unit: none

Value range: 1 to 65535

Default value: The value is specified during installation if the GTM 5 is deployed. Otherwise, the value is **6666**.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the port number is changed, the database communication will be affected.

gtm_port6

Parameter description: Specifies the listening port of the standby GTM 6 if the standby GTM 6 is deployed. This parameter is visible only to the SYSADMIN user.

Parameter type: integer.

Unit: none

Value range: 1 to 65535

Default value: The value is specified during installation if the GTM 6 is deployed. Otherwise, the value is **6666**.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the port number is changed, the database communication will be affected.

gtm_port7

Parameter description: Specifies the listening port of the standby GTM 7 if the standby GTM 7 is deployed. This parameter is visible only to the SYSADMIN user.

Parameter type: integer.

Unit: none

Value range: 1 to 65535

Default value: The value is specified during installation if the GTM 7 is deployed. Otherwise, the value is **6666**.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the port number is changed, the database communication will be affected.

enable_defer_calculate_snapshot

Parameter description: Delays the calculation of **xmin** and **oldestxmin** of a snapshot. The calculation is triggered only after 1000 transactions are executed or the calculation interval is 1s.

Parameter type: Boolean

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on** indicates that snapshots **xmin** and **oldestxmin** are calculated with a delay.
- **off** indicates that snapshots **xmin** and **oldestxmin** are calculated in real time.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **on**, the snapshot calculation overhead can be reduced in high load scenarios. However, **oldestxmin** is updated slowly, affecting dead tuple collection. If this parameter is set to **off**, **xmin** and **oldestxmin** can be updated in real time, but the overhead for calculating snapshots increases.

seqscan_csn_cache_num

Parameter description: Specifies the size of the CSN cache. The cache is used only when the Seq Scan scans the heap table page through the MVCC snapshot to determine the visibility.

Parameter type: integer.

Unit: none

Value range: 0 to 1000. If this parameter is set to **0**, the CSN cache mechanism is not used.

Default value: **100**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

reject_gtm_address

Parameter description: Specifies the IP address of the GTM that is rejected for connection.

Parameter type: string.

Unit: none

Value range: The substrings are separated by commas (,). The format of each substring is *IP[:port]*.

NOTE

[:port] is optional.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: This parameter is automatically set by the CM component in a specific process. Manual modification is not recommended.

Risks and impacts of improper settings: Improper settings may cause communication failures between the cluster and GTM, affecting cluster transactions.

verify_clog_csnlog_mode

Parameter description: Specifies the log level and alarm output for data correctness verification errors on the Clog and CSNlog pages.

Parameter type: enumerated type.

Unit: none

Value range:

- **off**: The error reporting and alarm output are disabled.
- **warning**: The warning-level error reporting and alarm output are enabled.
- **error**: The error-level error reporting and alarm output are enabled.
- **panic**: The panic-level error reporting and alarm output are enabled.

Default value:

- When a version that supports this function is installed, the default value is **warning**.
- When a version that does not support this parameter is upgraded to a version that supports this parameter, the default value is **off**. After the upgrade is complete, you need to manually set this parameter.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the actual requirements. If this parameter is set to **error** or **panic**, services or even database running will be affected when the page is faulty.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.20 Dual-Cluster Replication Parameters

enable_slot_log

Parameter description: Specifies whether to enable primary/standby synchronization for replication slots. Only logical replication slots, archive slots, and backup slots are involved.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Primary/standby synchronization is enabled for replication slots.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **off**: Primary/standby synchronization is disabled for replication slots.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Enable the function of creating archive/backup slots on the primary node and disable the function of creating archive/backup slots on the standby node.

Risks and impacts of improper settings: Functions (except replication and synchronization between the primary and standby nodes) such as backup, archiving, and building, will be affected.

RepOriginId

Parameter description: This is a session-level GUC parameter. In bidirectional logical replication, set it to a non-zero value to avoid infinite data replication.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on method 3 provided in **Table 6-2**.

Setting suggestion: In two-way logical replication scenarios, set this parameter to a non-zero value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

auto_csn_barrier

Parameter description: Specifies whether the barrier logging function is enabled for the primary cluster and GTM-lite cluster for streaming DR.

Parameter type: Boolean

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This function is automatically enabled during the DR process. You are advised not to manually disable this function. To enable the read function of the standby node in the GTM-lite cluster, set this parameter to **on**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this function is disabled, the data read consistency between the standby cluster for the DR and the standby node of the GTM-lite cluster will be affected.

stream_cluster_run_mode

Parameter description: Specifies whether a CN or DN belongs to the primary or standby cluster in a dual-cluster streaming DR scenario.

Parameter type: enumerated type

Unit: none

Value range:

- **cluster_primary** indicates that the node is in the primary cluster.
- **cluster_standby** indicates that the node is in the standby cluster.

Default value: **cluster_primary**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter specifies the status of a standby cluster for DR. It is used in internal processes and can be read by users. You are advised not to modify this parameter.

Risks and impacts of improper settings: Manual modification will cause standby clusters to be abnormal.

hadr_process_type

Parameter description: Specifies a process ID of a streaming replication-based remote DR solution or intra-city dual-DC HA solution.

Parameter type: enumerated type

Unit: none

Value range:

- **none**: There is no process.
- **failover** indicates that a standby cluster for DR is promoted to primary.
- **switchover_promote** indicates that a standby cluster is promoted to primary during the cluster switchover.
- **switchover_demote** indicates that a primary cluster is demoted to a standby cluster during the cluster switchover.
- **dorado_failover** indicates that an OceanStor Dorado standby cluster is promoted to primary.
- **dorado_switchover_demote** indicates that a primary cluster is demoted to a standby cluster during the OceanStor Dorado cluster switchover.
- **dorado_failover_abnormal** indicates that a standby cluster is promoted to primary when a shared disk of the OceanStor Dorado primary cluster is faulty.
- **dorado_failover_in_standby_stream** indicates that an OceanStor Dorado standby cluster is upgraded to an OceanStor Dorado primary cluster (also the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

remote DR-standby cluster) when the remote cluster in the streaming replication-based remote DR solution is the primary cluster.

- **dorado_failover_abnormal_in_standby_stream** indicates that the intra-city dual-DC HA solution supports streaming replication-based remote DR. In this solution, the remote cluster is the primary cluster for remote DR and the shared disk of the OceanStor Dorado primary cluster (also the standby cluster for remote DR) is faulty, the OceanStor Dorado standby cluster is switched to the OceanStor Dorado primary cluster (also the remote DR standby cluster).

Default value: **none**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter specifies the status of a standby cluster for DR. It is used in internal processes and can be read by users. You are advised not to modify this parameter.

Risks and impacts of improper settings: Manual modification will cause standby clusters to be abnormal.

cluster_name

Parameter description: Specifies the name of the current cluster.

NOTICE

cluster_name cannot be set to **"invalid_cluster_uuid"**.

Parameter type: string.

Unit: none

Value range: valid cluster name.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is used in the DR process. The standby cluster for DR identifies the cluster based on this parameter. You are advised not to modify this parameter.

Risks and impacts of improper settings: The manual modification may cause DR function exceptions.

hadr_primary_cluster_name

Parameter description: Specifies the name of the cluster at the current physical log sender in the current DR.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

If **hadr_primary_cluster_name** of the main standby cluster at the log receiver is set to **invalid_cluster_uuid**, log synchronization with the cluster at the log sender is not started, and the cluster at the log receiver remains disconnected.

Parameter type: string.

Unit: none

Value range: valid cluster name.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter identifies a cluster of the log receiver. It is used in DR processes and can be read by users. You are advised not to modify this parameter.

Risks and impacts of improper settings: The manual modification may cause DR function exceptions.

hadr_standby_cluster_num

Parameter description: Specifies the number of standby clusters for streaming DR.

NOTICE

- The **hadr_standby_cluster_num** parameter is invalid in the streaming replication-based remote DR solution (one primary cluster + one standby cluster).
- This parameter is invalid when **hadr_replconninfo_config_mode** is set to **cluster_manager**, that is, the DR link between the log sender cluster and the log receiver cluster is configured through cluster management.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter specifies the status of a standby cluster for DR. It is used in DR processes and can be read by users. You are advised not to modify this parameter.

Risks and impacts of improper settings: The manual modification may cause DR function exceptions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

switchover_abort_timeout

Parameter description: Specifies the timeout threshold. If the time for the primary node to exit the thread exceeds the timeout threshold after the message interaction between the primary and standby nodes is complete, the process is terminated unexpectedly.

Parameter type: integer.

Unit: milliseconds

Value range: 0 to 2147483647. **0** indicates that the timeout abort function is not used.

Default value: **10000**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the primary node may exit unexpectedly during the switchover.

6.3.21 Developer Options

allow_system_table_mods

Parameter description: Specifies whether the structure of a system catalog and the name of a system schema can be modified.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The structure of the system catalog or the name of the system schema can be modified.
- **off**: The structure of the system catalog or the name of the system schema cannot be modified.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised not to modify this parameter.

Risks and impacts of improper settings: If this parameter is enabled, system catalogs may be damaged or even the database cannot be started.

allow_create_sysobject

Parameter description: Specifies whether objects such as functions, stored procedures, and synonyms can be created or modified in the system schema. The system schema refers to the schema provided by the database after initialization, excluding the public schema. The OID of the system schema is usually smaller than 16384.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The initial user and system administrators can create or modify objects such as functions, stored procedures, and synonyms in the system schema. For details about whether other users are allowed to create these objects, see the permission requirements of the corresponding schema.
- **off**: All users are not allowed to create or modify objects such as functions, stored procedures, and synonyms in the system schema.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter to **off** to improve system security. If you do not need to create or modify objects in the system schema, you are advised to set this parameter to **off**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

If the database is newly installed or the GUC parameter does not exist in the basic version to be upgraded, the default value is **off**. If the GUC parameter exists in the basic version to be upgraded, the original setting is retained.

debug_assertions

Parameter description: Specifies whether to enable various assertion checks. To use this parameter, the macro USE_ASSERT_CHECKING must be defined (through the configure option **--enable-cassert**) during the GaussDB compilation.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Various assertion checks are enabled.
- **off**: Various assertion checks are disabled.

Default value: **off**

NOTE

If you compile GaussDB with the assertion check enabled, **debug_assertions** is set to **on** by default.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This function helps debugging. When an exception or breakdown occurs, enable this function to locate program errors.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ignore_checksum_failure

Parameter description: Specifies whether to ignore data check errors.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Data check errors are ignored.
- **off**: Data check errors are reported.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised not to modify this parameter.

Risks and impacts of improper settings: If this parameter is enabled, the system ignores the failure (but still generates an alarm). Continuing execution may result in breakdown, damaged data being transferred or stored, failure of data recovery from remote nodes, or other serious problems.

ignore_system_indexes

Parameter description: Specifies whether to ignore system indexes when reading system catalog (but still update the indexes when modifying the tables).

NOTE

This parameter is useful for recovering data from tables whose system indexes are damaged.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: System indexes are ignored.
- **off**: System indexes are not ignored.

Default value: **off**

Setting method: This is a BACKEND parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

ignore_invalid_pages

Parameter description: Specifies whether to trigger a PANIC error when an invalid page is detected at the end of replay.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Invalid pages are ignored at the end of replay. When an invalid page is detected, no PANIC error is triggered and only WARNING logs are recorded.
- **off**: Invalid pages are not ignored at the end of replay. When an invalid page is detected, a PANIC error is triggered.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If a PANIC error occurs during page verification in the database and services need to be quickly restored, you can set this parameter to **on** to restore the services. However, this behavior may cause abnormal termination, data loss, and incorrect data synchronization to other standby nodes.

post_auth_delay

Parameter description: Specifies the delay in the connection to the server after a successful authentication. Developers can attach a debugger to the server startup process.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147

Default value: **0**, indicating no delay.

Setting method: This is a BACKEND parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **60** without a unit, **post_auth_delay** indicates 60s. If the value is **1min**, **post_auth_delay** indicates 1 minute. The unit must be **s**, **min**, **h**, or **d** if required.

Setting suggestion: This parameter is used only for commissioning and fault locating. To prevent impact on service running, ensure that the default value **0** is used in the production environment.

Risks and impacts of improper settings: If this parameter is set to a non-zero value, the cluster status may be abnormal due to a long authentication delay.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

pre_auth_delay

Parameter description: Specifies the period of delaying authentication after the connection to the server is started. Developers can attach a debugger to the authentication procedure.

Parameter type: integer.

Unit: seconds

Value Range: 0 to 60

Default value: **0**, indicating no delay.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **60** without a unit, **pre_auth_delay** indicates 60s. If the value is **1min**, **pre_auth_delay** indicates 1 minute. The unit must be **s**, **min**, **h**, or **d** if required.

Setting suggestion: This parameter is used only for commissioning and fault locating. To prevent impact on service running, ensure that the default value **0** is used in the production environment.

Risks and impacts of improper settings: If this parameter is set to a non-zero value, the cluster status may be abnormal due to a long authentication delay.

trace_notify

Parameter description: Specifies whether to enable the function of generating debugging output for the **LISTEN** and **NOTIFY** commands. The level of **client_min_messages** or **log_min_messages** must be **DEBUG1** or lower so that such output can be recorded in the logs on the client or server separately.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The output function is enabled.
- **off**: The output function is disabled.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

trace_recovery_messages

Parameter description: Specifies whether to enable logging of recovery-related debugging output. If it is not enabled, no log is recorded This parameter allows users to overwrite the normal setting of **log_min_messages**, but only for specific messages. This is intended for the use in debugging the standby node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: enumerated type.

Unit: none

Value range: **debug5**, **debug4**, **debug3**, **debug2**, **debug1**, and **log**. For details about the parameters, see **Table 6-23**.

Default value: **log**

NOTE

- **log** indicates that recovery-related debugging information will not be logged.
- Except the default value **log**, each of the other values indicates that recovery-related debugging information at the specified level will also be logged. Common settings of **log_min_messages** enable logs to be unconditionally recorded into server logs.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised to change the value of this parameter only when commissioning the standby node. In other cases, you are advised to use the default value.

Risks and impacts of improper settings: If this parameter is set to a value other than **log**, a large number of logs will be printed, occupying certain I/O resources.

trace_sort

Parameter description: Specifies whether to print information about resource usage during sorting operations. This option can be used only when the **TRACE_SORT** macro is defined during GaussDB compilation. Currently, **TRACE_SORT** can only be set to the default value and cannot be adjusted.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Information about resource usage in sorting operations is printed.
- **off**: Information about resource usage in sorting operations is not printed.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

zero_damaged_pages

Parameter description: Specifies whether to terminate the current transaction after a damaged page header that causes GaussDB to report an error is detected.

Parameter type: Boolean.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **on**: The system generates a warning instead of throwing an error, clears the damaged page, and continues processing. This behavior will destroy data, including all the rows on the damaged page. However, it allows you to bypass the error and retrieve rows from any undamaged pages that may be present in the table. Therefore, it is useful for restoring data if corruption has occurred due to a hardware or software error.
- **off**: The system does not fill zeros in damaged pages.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: In most cases, you are advised not to set it to **on** if you want to restore data from damaged pages.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

string_hash_compatible

Parameter description: Specifies whether to use the same method to calculate char-type hash values and varchar- or text-type hash values. Based on the setting of this parameter, you can determine whether a redistribution is required when a distribution key is converted from a char-type data distribution into a varchar- or text-type data distribution.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The same calculation method is used and a redistribution is not required.
- **off**: Different calculation methods are used and a redistribution is required.

NOTE

Calculation methods differ in the length of input strings used for calculating hash values. (For a char-type hash value, spaces following a string are not counted as the length. For a text- or varchar-type hash value, the spaces are counted.) The hash value affects the calculation result of queries. To avoid query errors, do not modify this parameter during database running once it is set.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

remotetype

Parameter description: Specifies the remote connection type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: enumerated type.

Unit: none

Value range: **application**, **coordinator**, **datanode**, **gtm**, **gtmproxy**, **internaltool**, and **gtmtool**

Default value: **application**

Setting method: This is a BACKEND parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_user_defined_exception

Parameter description: Specifies the maximum number of exceptions. The default value cannot be changed.

Parameter type: integer.

Unit: none

Value range: Currently, only the fixed value **1000** is supported.

Default value: **1000**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

enable_fast_numeric

Parameter description: Specifies whether to enable optimization for numeric data calculation. Calculation of numeric data is time-consuming. Numeric data is converted into int64- or int128-type data to improve numeric data calculation performance.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Optimization for numeric data calculation is enabled.
- **off**: Optimization for numeric data calculation is disabled.

Default value: **on**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this parameter is disabled, the performance of query statements of the numeric type may deteriorate.

enable_numeric_optimization

Parameter description: Specifies whether to enable optimization for numeric data calculation.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enables NUMERIC data type optimization in the vectorized engine.
- **off**: disables NUMERIC data type optimization in the vectorized engine.

Default value: **off**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

NOTE

The current version does not support this optimization. Currently, only the default value of **enable_numeric_optimization** can be viewed.

enable_fast_vecop

Parameter description: Specifies whether to enable vectorized calculation optimization. The calculation of the Timestamp, int32, int64, and float8 types in the vectorization engine is optimized, the calling cost is reduced, and automatic vectorization calculation is supported.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The vectorized operation optimization is enabled.
- **off**: The vectorized operation optimization is disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the calculation performance of the Timestamp, int32, int64, and float8 types in the vectorization engine will be affected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_compress_spill

Parameter description: Specifies whether to enable the compression function of writing data to disk.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The compression function is enabled.
- **off**: The compression function is disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

resource_track_log

Parameter description: Specifies the log level of self-diagnosis. Currently, this parameter takes effect only on multi-column statistics.

Parameter type: enumerated type.

Unit: none

Value range:

- **summary**: Brief diagnosis information is displayed.
- **detail**: Detailed diagnosis information is displayed.

Currently, the two parameter values differ only when there is an alarm about multi-column statistics not collected. If the parameter is set to **summary**, such an alarm will not be displayed. If it is set to **detail**, such an alarm will be displayed.

Default value: **summary**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_parallel_ddl

Parameter description: Specifies whether multiple CNs can concurrently perform DDL operations on the same database object.

Parameter type: Boolean.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **on**: DDL operations can be concurrently performed without distributed deadlocks.
- **off**: DDL operations cannot be concurrently performed as distributed deadlocks may occur.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

support_batch_bind

Parameter description: Specifies whether to batch bind and execute PBE statements through APIs such as JDBC, ODBC, and libpq.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Batch binding and execution are used.
- **off**: Batch binding and execution are not used.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

numa_distribute_mode

Parameter description: Specifies the distribution of some shared data and threads among NUMA nodes. This parameter is used to optimize the performance of large-scale Arm servers with multiple NUMA nodes. Generally, you do not need to set this parameter.

Parameter type: string.

Unit: none

Value range: **"none"** and **"all"**.

- **"none"**: This feature is disabled.
- **"all"**: Some shared data and threads are distributed to different NUMA nodes to reduce the number of remote access times and improve performance. Currently, this function applies only to Arm servers with multiple NUMA

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

nodes. All NUMA nodes must be available for database processes. You cannot select only some NUMA nodes.

NOTE

- In the current version, **numa_distribute_mode** cannot be set to **"all"** on the x86 platform.
- If this parameter is enabled and CPU scaling is performed, you need to run the **gs_cgroup -u -T Gaussdb -f New CPU core range** command to update the Cgroup core binding information. Ensure that the range of available cores in the Cgroup is the same as the range of available CPU cores after scaling. Otherwise, the cluster fails to be started.

Default value: **"none"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_pagewriter

Parameter description: Specifies whether to display the page refresh information of threads and details about incremental checkpoints after incremental checkpoints are enabled.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Detailed information is displayed.
- **off**: Detailed information is not displayed.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised not to set this parameter to **on**.

Risks and impacts of improper settings: If this parameter is set to **on**, more information will be displayed, affecting performance.

advance_xlog_file_num

Parameter description: Specifies the number of Xlog files that are periodically initialized in advance in the backend.

Parameter type: integer.

Unit: none

Value range: 0 to 1000000. The value **0** indicates that initialization is not performed in advance. For example, the value **10** indicates that the background

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

thread periodically initializes 10 Xlog files in advance based on the write location of the current Xlog.

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Use this parameter to prevent the Xlog file initialization from affecting the performance during transaction commit. However, such a fault may occur only when the system is overloaded. Therefore, you do not need to set this parameter.

Risks and impacts of improper settings: If the value is too large, too much disk space is occupied. As a result, the disk usage is abnormal. Therefore, you need to consider the actual disk size when adjusting this parameter. The disk space occupied by this parameter can be calculated using the following formula: **advance_xlog_file_num** x **wal_segment_size** (16 MB). It is recommended that the pre-allocated space be less than or equal to 10% of the available space. If the value is too large, the cluster may be unavailable and cannot be restored.

comm_sender_buffer_size

Parameter description: Specifies the size of the buffer for each interaction between CNs and DNs and between DNs in the stream plan. In some cases, different values affect the stream performance. After the value is set, the cluster needs to be restarted for the reset to take effect.

Parameter type: integer.

Unit: KB

Value range: 1 to 1024

Default value: **8**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, data is frequently transmitted and received, the delay is short, but the throughput is low. If the value is too large, the throughput increases but the delay increases.

ustore_attr

Parameter description: This parameter is used to control the information statistics of Ustore tables, rollback type, UB-tree index type, and data verification during the running of key modules (including data, indexes, rollback segments, and replay). This parameter helps R&D engineers locate faults.

Parameter type: string.

Unit: none

Value range: This parameter is set in key-value mode. The mapping between keys and values is as follows: If multiple key-value pairs are used, use semicolons (;) to

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

separate them. For example, **ustore_attr='ustore_verify_level=FAST;ustore_verify_module=UPAGE:UBTREE:UNDO:REDO'**.

- **ustore_verify_level**: verification level.

  Value range: a string of case-insensitive characters. For details, see the following table.

  Table 6-48 Parameter value meaning of ustore_verify_level

Parameter Value	Description
NONE	**NONE** indicates that the verification function is disabled. You are advised to enable this function to test performance.
FAST	**FAST** indicates fast verification. The verification content is small and the impact on performance is minimized.
COMPLETE	**COMPLETE** indicates complete verification. The verification content is the largest and the performance is greatly affected.
NORMAL	This level has the same function as the **FAST** level and is reserved only for compatibility with earlier versions. The **FAST** level is recommended.
SLOW	This level has the same function as the **COMPLETE** level and is reserved only for compatibility with earlier versions. The **COMPLETE** level is recommended.

  Default value: **FAST**

- **ustore_verify_module**: module that controls verification.

  Value range: The value can be one or more of **UPAGE**, **UBTREE**, **UNDO**, **REDO**, and **ROACH**, or it can be **ALL** or **NULL** (case-insensitive). When multiple values of **UPAGE**, **UBTREE**, **UNDO**, **REDO**, and **ROACH** are used, separate them with colons (:). For example, **ustore_verify_module=UPAGE:UBTREE:UNDO:REDO**.

  Table 6-49 Parameter value meaning of ustore_verify_module

Parameter Value	Description
UPAGE	Indicates that data page verification is enabled.
UBTREE	Indicates that UB-tree index verification is enabled.
UNDO	Indicates that rollback segment data verification is enabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter Value	Description
REDO	Indicates that data page verification for the REDO process is enabled.
ROACH	This parameter is invalid in the current version. The verification has been removed from the underlying logic. Although **ustore_verify_module** can be set to **roach**, the setting does not take effect.
ALL	Indicates that data verification is enabled for the UPAGE, UBTREE, UNDO, REDO, and ROACH modules.
NULL	Indicates that data verification for the UPAGE, UBTREE, UNDO, REDO, and ROACH modules is disabled.

Default value: **UPAGE:UBTREE:UNDO**

- **index_trace_level**: specifies whether to enable index tracing and controls the printing level. After this function is enabled, information about index tuples that meet the conditions is printed based on the printing level during index scan.

  Value range: The values are described in the following table.

  Default value: **NO**

  Table 6-50 Parameter value meaning of index_trace_level

Parameter Value	Description
NO	No additional information is printed.
NORMAL	Information about visible index tuples is printed, including: ● ID and offset of the index page where the current index tuple is located ● Current tuple status ● TID and partOid corresponding to the current tuple ● xmin and xmax information corresponding to the current tuple ● Current tuple content (if **enable_log_tuple** is set to **on**).
VISIBILITY	On the basis of **NORMAL**, the information about the index tuples that do not pass the visibility check is printed and whether the index tuples are visible is marked.
SHOWHIKEY	On the basis of **VISIBILITY**, the system tries to print the information about the HIKEY tuple on the page.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter Value	Description
ALL	Information about all tuples on the scanned index page is printed.

- **enable_log_tuple**: specifies whether to print the contents of related tuples when printing log-level prompts for troubleshooting and locating.

  Value range: **on** or **off** (case-insensitive)

  Default value: **off**
- **enable_ustore_sync_rollback**: specifies whether to enable synchronous rollback for Ustore tables.

  Value range: Boolean

  Default value: **true**
- **enable_ustore_async_rollback**: specifies whether to enable asynchronous rollback for Ustore tables.

  Value range: Boolean

  Default value: **true**
- **enable_ustore_page_rollback**: specifies whether to enable page rollback for Ustore tables.

  Value range: Boolean

  Default value: **true**
- **enable_ustore_partial_seqscan**: specifies whether to enable partial scan for Ustore tables.

  Value range: Boolean

  Default value: **false**
- **enable_candidate_buf_usage_count**: specifies whether to enable buffer usage statistics.

  Value range: Boolean

  Default value: **false**
- **ustats_tracker_naptime**: specifies the interval for collecting statistics on Ustore tables.

  Value range: 1 to 1073741

  Default value: **20**, in seconds.
- **umax_search_length_for_prune**: specifies the maximum search depth of the prune operation on the Ustore table.

  Value range: 1 to 1073741

  Default value: **10**
- **ustore_td_reservation_timeout**: specifies the maximum timeout interval for a DML statement to wait for TD on the Ustore table page.

  Value range: 0 to 120000

  Default value: **60000**, in milliseconds.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When setting **ustore_attr**, do not leave spaces or other characters before and after the equal sign (=) between key and value, for example, **ustore_attr='ustore_verify_level = FAST;'**. Otherwise, the parameter is invalid during kernel code verification and the parameter setting fails.
- The online verification function is enabled by default. To disable the online verification function, set **ustore_attr** to **""**.

Default value:
"ustore_verify_level=FAST;ustore_verify_module=UPAGE:UBTREE:UNDO"

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Exercise caution when setting the **ustore_attr** parameter. You are advised to modify it with the assistance of engineers.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

index_txntype

Parameter description: Determines whether the index type of UB-tree is PCR or RCR. PCR supports flashback query based on indexes, but RCR does not. An empty string is equivalent to RCR in most scenarios. In the reindex scenario, the new index keeps the original index type, which is not affected by this parameter.

Parameter type: string.

Unit: none

Value range: **""**, **"PCR"**, or **"RCR"**

Default value: **""**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If the PCR index is required, set this parameter to **"PCR"**. Otherwise, you are advised not to change the value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

default_index_kind

Parameter description: Specifies the default behavior of creating indexes for partitioned tables.

Parameter type: integer.

Unit: none

Value range: 0 to 2

- **0**: The global partitioned index function is disabled for distributed deployment.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **1**: A local index is created by default.
- **2**: A global index is created by default.

Default value: **2**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **0**, global partitioned indexes cannot be created.

enable_auto_segment_remain_cleanup

Parameter description: Specifies whether to enable automatic clearing of residual segment-page data.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Residual segment-page data is automatically cleared.
- **off**: Residual segment-page data is not automatically cleared.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Do not switch the clearing mode randomly. Otherwise, residual segment-page data may fail to be cleared or other exceptions may occur.

enable_stream_noblock_memcopy

Parameter description: Specifies whether to enable memory copy optimization for the stream operator in the vectorized engine. If this parameter is enabled, data of the local stream operator in the vectorized engine is sent in non-waiting mode, improving the execution efficiency of the local stream operator.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Memory copy optimization is enabled for the stream operator.
- **off**: Memory copy optimization is disabled for the stream operator.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this parameter is set to **off**, the execution efficiency of the local stream operator may decrease.

enable_remove_redundant_update_col

Parameter description: Specifies whether to optimize the pseudo-update of distribution keys. For details about this parameter, see **Pseudo-update of Distribution Keys**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The pseudo-update of distribution keys is optimized.
- **off**: The pseudo-update of distribution keys is not optimized.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements.

Risks and impacts of improper settings: If this parameter is set to **on**, the pseudo-update of distribution keys is optimized. As a result, the performance deteriorates slightly in real update scenarios. Set this parameter to **off** if it is not used.

convert_illegal_char_mode

Parameter description: Specifies the placeholders of invalid characters that can be displayed on the client when the **enable_convert_illegal_char** parameter is enabled.

Parameter type: string.

Unit: none

Value range: 95 characters whose decimal codes range from 32 to 126 in the ASCII coding table.

Default value: **"?"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

default_segment

Parameter description: Specifies whether to create a segment-page table by default.

Parameter type: Boolean.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range:

- **on**: A segment-page table is created by default when the segment field is not specified.
- **off**: A page table is created by default when the segment field is not specified.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

pldebugger_timeout

Parameter description: Specifies the timeout interval for the pldebugger server to wait for a response from the debug end.

Parameter type: integer.

Unit: seconds

Value range: 1 to 86400

Default value: **900** (that is, 15 minutes)

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **3600** without a unit, **pldebugger_timeout** indicates 3600s. If the value is **1h**, **pldebugger_timeout** indicates 1 hour. The unit must be **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.22 Auditing

6.3.22.1 Audit Switch

audit_enabled

Parameter description: Specifies whether to enable the traditional audit.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the auditing function is enabled.
- **off** indicates that the auditing function is disabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Enabling the audit function affects the database performance. The impact depends on the audit recording frequency. If this parameter is set to **on**, set other audit items based on service and tracing requirements. If this parameter is set to **off**, historical operations cannot be traced using audit logs. To ensure system stability, do not disable the audit function during frequent audit operations.

audit_directory

Parameter description: Specifies the storage directory of audit files. The path can be relative to the **data** directory or an absolute path. Only the SYSADMIN user can access this parameter.

Parameter type: string.

Unit: none

Value range: valid target path.

Default value: **"pg_audit"**. If **om** is used for cluster deployment, audit logs are stored in *$GAUSSLOG*/**pg_audit/***Instance name*.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is an invalid path or a path whose space is full, the audit function cannot be used. Different directories for storing audit files must be set for different CNs or DNs. Otherwise, audit logs will be abnormal.

NOTE

- Valid path: Users have read and write permissions on the path.
- Invalid path: Users do not have read or write permission on the path.

audit_data_format

Parameter description: Audits the format of log files. Currently, only the binary or TXT format is supported. Only the SYSADMIN user can access this parameter.

Parameter type: string.

Unit: none

Value range:

- **BIN**: binary format.
- **text**: TEXT format.

Default value: **"binary"**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Currently, only the binary or TEXT format is supported. If the format is invalid, the default value is used.

audit_rotation_interval

Parameter description: Specifies the interval of creating an audit log file. When the difference between the current time and the time when the last audit log is created exceeds the value of this parameter, the server generates a new audit log file. A random delay is introduced to the performance interval.

Parameter type: integer.

Unit: minute

Value range: 1 to 35791394 (*INT_MAX*/60)

Default value: 1d (that is, 1440 minutes)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1440** without a unit, **audit_rotation_interval** indicates 1440 minutes. If the value is **2d**, **audit_rotation_interval** indicates 2 days, that is, 2880 minutes. The unit must be **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, there will be too many files in the audit directory. If the value is too large, a single audit file will be too large. Do not adjust this parameter unless necessary. Otherwise, **audit_resource_policy** may not take effect. To control the storage space and time of audit logs, use parameters such as **audit_resource_policy**, **audit_space_limit**, and **audit_file_remain_time**.

audit_rotation_size

Parameter description: Specifies the maximum size of an audit log file (the actual file size may exceed the upper limit by 1 MB to 10 MB). If the total number of messages in an audit log exceeds the value of this parameter, the server will generate a new audit log file.

Parameter type: integer.

Unit: KB

Value range: 1024 to 1048576

Default value: **10MB** (that is, 10240 KB)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **131072** without a unit, **audit_rotation_size** indicates 131072 KB. If the value is **128MB**, **audit_rotation_size** indicates 128 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If the value is too small, there will be too many files in the audit directory. If the value is too large, a single audit file will be too large. Do not adjust this parameter unless necessary. Otherwise, **audit_resource_policy** may not take effect. To control the storage space and time of audit logs, use parameters such as **audit_resource_policy**, **audit_space_limit**, and **audit_file_remain_time**. If the space occupied by a single record in an audit log file exceeds the value of this parameter, the log file is regarded as an invalid log file.

audit_resource_policy

Parameter description: Specifies the policy for determining whether audit logs are preferentially stored by space or time.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The space-first policy is used, with the maximum storage space set by **audit_space_limit**.
- **off**: The time-first policy is used, with the minimum storage duration set by **audit_file_remain_time**.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: When the space-first policy is selected, do not adjust the GUC parameters related to the time policy: **audit_rotation_interval** and **audit_file_remain_time**. When the time-first policy is selected, do not adjust the GUC parameters related to the space policy: **audit_rotation_size**, **audit_space_limit**, and **audit_file_remain_threshold**.

Risks and impacts of improper settings: This parameter must be used together with other GUC parameters related to the corresponding policy. If this parameter is set to an improper value, the audit log storage policy does not meet expectations.

audit_file_remain_time

Parameter description: Specifies the minimum duration required for recording audit logs. This parameter is valid only when **audit_resource_policy** is set to **off**.

Parameter type: integer.

Unit: days (d)

Value range: 0 to 730. The value **0** indicates no time limit.

Default value: **90**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Set this parameter according to the actual situation. If **audit_resource_policy** is set to **on**, do not change the value of this parameter.

Risks and impacts of improper settings: This parameter takes effect only when **audit_resource_policy** is set to **off**. If the value is too small, only a small number of audit logs are retained and the audit tracing duration is shortened. If the value is too large, too many audit logs are stored in the audit directory, affecting the audit query and deletion performance and occupying disk space.

audit_space_limit

Parameter description: Specifies the total disk space occupied by audit files.

Parameter type: integer.

Unit: KB

Value range: 1024 to 1073741824 (1024 GB)

Default value: **1GB** (that is, 1048576 KB)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **131072** without a unit, **audit_space_limit** indicates 131072 KB. If the value is **128MB**, **audit_space_limit** indicates 128 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Set this parameter based on the actual available space of the instance and the number of audit logs to be saved.

Risks and impacts of improper settings: This parameter takes effect only when **audit_resource_policy** is set to **on**. If the value is too small, only a small number of audit logs are retained and the audit tracing duration is shortened. If the value is too large, too many audit logs are stored in the audit directory, affecting the audit query and deletion performance and occupying disk space.

NOTE

- This parameter takes effect only for a single process instance folder in the **pg_audit** directory. By default, the total disk space occupied by audit files on each CN or DN is 1 GB.
- In the multi-audit thread scenario, the minimum disk space occupied by audit files is the product of values of **audit_thread_num** and **audit_rotation_size**. Ensure that the value of **audit_space_limit** is greater than the product of values of **audit_thread_num** and **audit_rotation_size**.

audit_file_remain_threshold

Parameter description: Specifies the maximum number of audit files in the audit directory.

Parameter type: integer.

Unit: none

Value range: 100 to 1048576

Default value: **1048576**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value and do not modify this parameter unless necessary. To control the storage space and time of audit logs, use the **audit_resource_policy**, **audit_space_limit**, and **audit_file_remain_time** parameters.

Risks and impacts of improper settings: If this parameter is set improperly, **audit_resource_policy** may not take effect.

audit_thread_num

Parameter description: Specifies the number of threads used for auditing.

Parameter type: integer.

Unit: none

Value range: 1 to 48

Default value: **1**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: When **audit_dml_state** is enabled and high performance is required, you are advised to set this parameter to a larger value to ensure that audit messages can be processed and recorded in a timely manner. In other cases, you are advised to retain the default value.

Risks and impacts of improper settings: A larger number of threads indicates a greater impact on system performance and more CPU and I/O resources are occupied.

6.3.22.2 User and Permission Audit

audit_login_logout

Parameter description: Specifies whether to audit a user's login (including login success and failure) and logout.

Parameter type: integer.

Unit: none

Value range: 0 to 7

- **0**: The function of auditing users' logins and logouts is disabled.
- **1**: Only successful user logins are audited.
- **2**: Only failed user logins are audited.
- **3**: Only successful and failed user logins are audited.
- **4**: Only user logouts are audited.
- **5**: Only user logouts and successful logins are audited.
- **6**: Only user logouts and failed logins are audited.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **7**: Successful user logins, failed user logins, and logouts are audited.

Default value: **7**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: A smaller value indicates fewer types of user login and logout behaviors to be audited and higher risks of untraceable user login and logout behaviors. A larger value indicates more types of user login and logout behaviors to be recorded. When a large number of user login and logout behaviors exist, the database performance is affected.

audit_database_process

Parameter description: Specifies whether to audit the database startup, stop, switchover, and recovery.

Parameter type: integer.

Unit: none

Value range: **0** and **1**

- **0**: The function of auditing database startup, stop, switchover, and recovery is disabled.
- **1**: The function of auditing database startup, stop, switchover, and recovery is enabled.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **0**, database startup, stop, switchover, and recovery may fail to be traced.

audit_user_locked

Parameter description: Specifies whether to audit the users' locking and unlocking.

Parameter type: integer.

Unit: none

Value range: **0** and **1**

- **0**: The function of auditing user's locking and unlocking is disabled.
- **1**: The function of auditing user's locking and unlocking is enabled.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **0**, user locking and unlocking behaviors may not be traced.

audit_user_violation

Parameter description: Specifies whether to audit the access violation operations of a user.

Parameter type: integer.

Unit: none

Value range: **0** and **1**

- **0**: The function of auditing the access violation operations of a user is disabled.
- **1**: The function of auditing the access violation operations of a user is enabled.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter to **1** if unauthorized operations need to be audited.

Risks and impacts of improper settings: If this parameter is set to **0**, unauthorized audit operations may fail to be traced. If this parameter is set to **1**, a large number of unauthorized operations may affect database performance.

audit_grant_revoke

Parameter description: Specifies whether to audit the granting and revoking of user permissions.

Parameter type: integer.

Unit: none

Value range: **0** and **1**

- **0**: The function of auditing user permission granting and revoking operations is disabled.
- **1**: The function of auditing user permission granting and revoking operations is enabled.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **0**, user permission granting and revoking behaviors may not be traced. If this parameter is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

set to **1**, the database performance will be affected when a large number of permissions are granted and revoked.

audit_security_label

Parameter description: Specifies whether to audit the creation, deletion, and application of user security labels.

Parameter type: integer.

Unit: none

Value range: **0** or **1**

- **0**: The creation, deletion, and application of user security labels are not audited.
- **1**: The creation, deletion, and application of user security labels are audited.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **0**, the creation, deletion, and application of user security labels cannot be traced. If this parameter is set to **1**, the database performance will be affected when a large number of user security labels are created, deleted, and applied.

audit_internal_event

Parameter description: Specifies whether to audit the connections and operations of internal tools cm_agent, gs_clean, and WDRXdb, and whether to audit the connections from CNs on DNs for the initial user.

Parameter type: Boolean

Unit: none

Value range:

- **on**: indicates that the logins, logouts, and operations of internal tools CM Agent, gs_clean, and WDRXdb are audited, and logins and logouts from CNs are audited on DNs for the initial user.
- **off**: indicates that the logins, logouts, and operations of internal tools CM Agent, gs_clean, and WDRXdb are not audited, and logins and logouts from CNs are not audited on DNs for the initial user.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value to save space occupied by audit logs and improve audit log query performance.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this parameter is enabled, login and logout operations between internal tools or nodes on the database server will be recorded, causing performance deterioration.

full_audit_users

Parameter description: Specifies the full audit user list. Audit logs are recorded for all auditable operations performed by users in the list.

Parameter type: string.

Unit: none

Value range: a string. Use commas (,) to separate multiple usernames.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter when all behaviors of a specific user need to be recorded and audited.

Risks and impacts of improper settings: If there are too many users in the list, all operations performed by these users are recorded in audit logs, causing database performance deterioration.

no_audit_client

Parameter description: Specifies the names and IP addresses of clients that do not need to be audited. The IP addresses can be IPv6 addresses. The parameter format is *client name*@*IP address*, which is the same as that of the **client_conninfo** column in the gs_query_audit function, for example, **cm_agent@127.0.0.1 or gs_clean@127.0.0.1**.

Parameter type: string.

Unit: none

Value range: a character string. Use commas (,) to separate multiple configuration items.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Audit logs are generated for communication among tools or nodes in the database server. To save space occupied by audit logs and improve the query performance of audit logs, you can configure the **no_audit_client** parameter not to audit the low-risk scenarios.

Risks and impacts of improper settings: If the settings are incorrect, operations performed on a client may fail to be audited. As a result, risky operations cannot be traced.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

If the executed SQL statement meets the configuration requirements of **full_audit_users** and **no_audit_client**, the **no_audit_client** is preferentially configured.

6.3.22.3 Operation Auditing

audit_system_object

Parameter description: Specifies whether to audit the operations such as CREATE, DROP, and ALTER on database objects. Database objects include databases, users, schemas, and tables. You can change the value of this parameter to audit only the operations on required database objects.

Parameter type: integer.

Unit: none

Value range: 0 to 1073741823

- **0** indicates that the operations such as CREATE, DROP, and ALTER are not audited.
- A non-zero value indicates that the operations such as CREATE, DROP, and ALTER on a certain or some database objects are audited.

Value description:

The value of this parameter is calculated by 30 binary bits. The 30 binary bits represent 30 types of database objects. If the corresponding binary bit is set to **0**, the operations such as CREATE, DROP, and ALTER on corresponding database objects are not audited. If it is set to **1**, the operations such as CREATE, DROP, and ALTER are audited. For details about the audit contents represented by these 30 binary bits, see **Table 6-51**.

When SQL patches are audited and **audit_dml_state_select** is enabled, an SQL patch operation will be audited twice and recorded as DML and DDL operations in the audit log, respectively. If a remote API is called, the DDL logs are generated on the node corresponding to the input parameter, instead of on the node where the statement is issued.

Default value: **67121159** (decimal), corresponding to 00 0100 0000 0000 0011 0000 0000 0111 in binary, indicating that DDL operations on DATABASE, SCHEMA, USER, NODE GROUP, and SQLPatch are audited.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set the type of database objects to be audited based on service requirements. In the scenario where the primary node is forcibly selected from the primary and standby nodes, you are advised to set **audit_system_object** to the maximum value to audit all objects in table 1.

Risks and impacts of improper settings: A larger number of object types to audit indicates a greater impact on system performance, and more CPU and I/O resources are occupied.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-51 Meaning of each value for the **audit_system_object** parameter

Binary Bit	Description	Value Range
Bit 0	Specifies whether to audit the CREATE, DROP, and ALTER operations on databases.	• **0** indicates that the CREATE, DROP, and ALTER operations on these objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on these objects are audited.
Bit 1	Specifies whether to audit the CREATE, DROP, and ALTER operations on schemas.	• **0** indicates that the CREATE, DROP, and ALTER operations on these objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on these objects are audited.
Bit 2	Specifies whether to audit the CREATE, DROP, and ALTER operations on users and user mappings.	• **0** indicates that the CREATE, DROP, and ALTER operations on these objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on these objects are audited.
Bit 3	Specifies whether to audit the CREATE, DROP, ALTER, TRUNCATE, VACUUM, and ANALYZE operations on tables.	• **0** indicates that the CREATE, DROP, ALTER, TRUNCATE, VACUUM, and ANALYZE operations on these objects are not audited. • **1** indicates that the CREATE, DROP, ALTER, TRUNCATE, VACUUM, and ANALYZE operations on these objects are audited.
Bit 4	Specifies whether to audit the CREATE, DROP, and ALTER operations on indexes.	• **0** indicates that the CREATE, DROP, and ALTER operations on these objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on these objects are audited.
Bit 5	• Specifies whether to audit the CREATE, DROP, and ALTER operations on views. • Specifies whether to audit the CREATE and DROP operations on MATVIEW objects.	• **0** indicates that the CREATE, DROP, and ALTER operations on views and the CREATE and DROP operations on MATVIEW objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on views and the CREATE and DROP operations on MATVIEW objects are audited.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Binary Bit	Description	Value Range
Bit 6	Specifies whether to audit the CREATE, DROP, and ALTER operations on triggers.	• **0** indicates that the CREATE, DROP, and ALTER operations on these objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on these objects are audited.
Bit 7	Specifies whether to audit the CREATE, DROP, and ALTER operations on procedures/functions.	• **0** indicates that the CREATE, DROP, and ALTER operations on these objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on these objects are audited.
Bit 8	Specifies whether to audit the CREATE, DROP, and ALTER operations on tablespaces.	• **0** indicates that the CREATE, DROP, and ALTER operations on these objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on these objects are audited.
Bit 9	Specifies whether to audit the CREATE, DROP, and ALTER operations on resource pools.	• **0** indicates that the CREATE, DROP, and ALTER operations on these objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on these objects are audited.
Bit 10	Specifies whether to audit the CREATE, DROP, and ALTER operations on workloads.	• **0** indicates that the CREATE, DROP, and ALTER operations on these objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on these objects are audited.
Bit 11	Specifies whether to audit the CREATE, DROP, and ALTER operations on server objects.	• **0** indicates that the CREATE, DROP, and ALTER operations on these objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on these objects are audited.
Bit 12	Reserved.	-
Bit 13	Specifies whether to audit the CREATE and DROP operations on node groups.	• **0** indicates that the CREATE and DROP operations on these objects are not audited. • **1** indicates that the CREATE and DROP operations on these objects are audited.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Binary Bit	Description	Value Range
Bit 14	Specifies whether to audit the CREATE, DROP, and ALTER operations on ROW LEVEL SECURITY objects.	• **0** indicates that the CREATE, DROP, and ALTER operations on these objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on these objects are audited.
Bit 15	Specifies whether to audit the CREATE, DROP, and ALTER operations on types.	• **0** indicates that the CREATE, DROP, and ALTER operations on types are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on types are audited.
Bit 16	Specifies whether to audit the CREATE, DROP, and ALTER operations on text search objects (CONFIGURATION and DICTIONARY).	• **0** indicates that the CREATE, DROP, and ALTER operations on text search objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on text search objects are audited.
Bit 17	Specifies whether to audit the CREATE, DROP, and ALTER operations on directories.	• **0** indicates that the CREATE, DROP, and ALTER operations on directories are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on directories are audited.
Bit 18	Specifies whether to audit the CREATE, DROP, and ALTER operations on synonyms.	• **0** indicates that the CREATE, DROP, and ALTER operations on synonyms are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on synonyms are audited.
Bit 19	Specifies whether to audit the CREATE, DROP, and ALTER operations on sequences.	• **0** indicates that the CREATE, DROP, and ALTER operations on sequences are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on sequences are audited.
Bit 20	Specifies whether to audit the CREATE, ALTER, and DROP operations on CMKs and CEKs.	• **0** indicates that the CREATE, ALTER, and DROP operations on CMKs and CEKs are not audited. • **1** indicates that the CREATE, ALTER, and DROP operations on CMKs and CEKs are audited.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Binary Bit	Description	Value Range
Bit 21	Specifies whether to audit the CREATE, DROP, and ALTER operations on packages.	• **0** indicates that the CREATE, DROP, and ALTER operations on packages are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on packages are audited.
Bit 22	Reserved	-
Bit 23	Reserved	-
Bit 24	Specifies whether to audit the ALTER and DROP operations on the gs_global_config objects.	• **0** indicates that the ALTER and DROP operations on the **gs_global_config** objects are not audited. • **1** indicates that the ALTER and DROP operations on the gs_global_config objects are audited.
Bit 25	Specifies whether to audit the CREATE, DROP, and ALTER operations on FOREIGN DATA WRAPPER objects. Currently, this function is not supported.	• **0** indicates that the CREATE, DROP, and ALTER operations on FOREIGN DATA WRAPPER objects are not audited. • **1** indicates that the CREATE, DROP, and ALTER operations on FOREIGN DATA WRAPPER objects are audited.
Bit 26	Specifies whether to audit the CREATE, ENABLE, DISABLE, and DROP operations on SQL patches.	• **0** indicates that the CREATE, ENABLE, DISABLE, and DROP operations on SQL patches are not audited. • **1** indicates that the CREATE, ENABLE, DISABLE, and DROP operations on SQL patches are audited.
Bit 27	Reserved	-
Bit 28	Specifies whether to audit the CREATE, ALTER, and DROP operations on database links.	• **0** indicates that the CREATE, ALTER, and DROP operations on database links are not audited. • **1** indicates that the CREATE, ALTER, and DROP operations on database links are audited.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Binary Bit	Description	Value Range
Bit 29	Specifies whether to audit the CREATE, DROP, and ALTER operations on resource labels, masking policies, and audit policies.	• **0** indicates that the CREATE, ALTER, and DROP operations on resource labels, masking policies, and audit policies are not audited. • **1** indicates that the CREATE, ALTER, and DROP operations on resource labels, masking policies, and audit policies are audited.

audit_dml_state

Parameter description: Specifies whether to audit the INSERT, UPDATE, DELETE, and MERGE operations on all tables.

Parameter type: integer.

Unit: none

Value range: **0** and **1**

- **0**: The INSERT, UPDATE, DELETE, and MERGE operations on a specific table are not audited.
- **1**: The INSERT, UPDATE, DELETE, and MERGE operations on a specific table are audited.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **1**, audit logs are frequently recorded when INSERT, UPDATE, DELETE, and MERGE operations are frequently performed in the database. As a result, the database performance deteriorates.

audit_dml_state_select

Parameter description: Specifies whether to audit the SELECT operation.

Parameter type: integer.

Unit: none

Value range: **0** and **1**

- **0**: The SELECT auditing function is disabled.
- **1**: The SELECT auditing function is enabled.

Default value: **0**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **1**, audit logs are frequently recorded when the SELECT operation is frequently performed in the database. As a result, the database performance deteriorates.

audit_function_exec

Parameter description: Specifies whether to record the audit information during the execution of the stored procedures, anonymous blocks, or user-defined functions (excluding system functions).

Parameter type: integer.

Unit: none

Value range: **0** and **1**

- **0**: The stored procedures, anonymous blocks, or user-defined functions (excluding system functions) are not audited.
- **1**: The stored procedures, anonymous blocks, or user-defined functions (excluding system functions) are audited.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **1**, audit logs are frequently recorded when the database frequently executes stored procedures, anonymous blocks, or user-defined functions. As a result, the database performance deteriorates.

audit_system_function_exec

Parameter description: Specifies whether to record audit logs when system functions in the whitelist are executed.

Parameter type: integer.

Unit: none

Value range: **0** and **1**

- **0**: The function of auditing the execution of system functions is disabled.
- **1**: The function of auditing system function execution is enabled.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this parameter is set to **1**, audit logs are frequently recorded when the database frequently executes system functions in the whitelist. As a result, the database performance deteriorates.

The following table lists the whitelist of system functions that can be recorded and audited.

set_working_grand_version_num_manually	set_config	pg_terminate_backend	pg_cancel_backend	pg_cancel_session	pg_cancel_invalid_query
pg_reload_conf	pg_rotate_logfile	pg_terminate_session	pg_terminate_backend	pg_start_backup	pg_stop_backup
pg_create_restore_point	pg_switch_xlog	pg_cbm_get_merged_file	pg_cbm_recycle_file	pg_enable_delay_ddl_recycle	pg_disable_delay_ddl_recycle
pg_cbm_rotate_file	gs_roach_enable_delay_ddl_recycle	gs_roach_disable_delay_ddl_recycle	gs_roach_stop_backup	pg_last_xlog_receive_location	pg_xlog_replay_pause
pg_xlog_replay_resume	gs_roach_switch_xlog	gs_pitr_archive_slot_force_advance	gs_pitr_clean_history_global_barriers	gs_download_obs_file	gs_upload_obs_file
gs_set_obs_file_context	gs_set_obs_delete_location	gs_hadr_do_switchover	gs_set_obs_delete_location_with_slotname	gs_streaming_dr_in_switchover	pg_advisory_lock
pg_advisory_lock_shared	pg_advisory_unlock	pg_advisory_unlock_shared	pg_advisory_unlock_all	pg_advisory_xact_lock	pg_advisory_xact_lock_shared
pg_try_advisory_lock	pg_try_advisory_lock_shared	pg_try_advisory_xact_lock	pg_try_advisory_xact_lock_shared	gs_get_hadr_key_cn	pg_create_physical_replication_slot_extern
pg_create_logical_replication_slot	pg_drop_replication_slot	pg_logical_slot_peek_changes	pg_logical_slot_get_changes	pg_logical_slot_get_binary_changes	pg_replication_origin_drop
pg_replication_origin_session_reset	local_space_shrink	gs_space_shrink	global_space_shrink	pg_free_remain_segment	gs_fault_inject

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

sqladvisor.init	sqladvisor.set_weight_params	sqladvisor.set_cost_params	sqladvisor.assign_table_type	gs_repair_file	local_clear_bad_block_info
gs_repair_page	clean_abnormal_sql	abnormal_sql_manage_allowlist	dsl_add_rule	dsl_del_rule	dsl_set_rule
dsl_reload_rule	pgxc_fenced_udf_process	pgxc_terminate_all_fenced_udf_process	-	-	-

NOTE

In the audit record of the system function execution event, the value of the **object_name** column is the system function name and does not contain function parameters.

audit_copy_exec

Parameter description: Specifies whether to audit the COPY operation.

Parameter type: integer.

Unit: none

Value range: **0** and **1**

- **0**: The function of auditing the COPY operation is disabled.
- **1**: The function of auditing the COPY operation is enabled.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **1**, audit logs are frequently recorded when the COPY operation is frequently performed in the database. As a result, the database performance deteriorates.

audit_set_parameter

Parameter description: Specifies whether to audit the SET operation.

Parameter type: integer.

Unit: none

Value range: **0** and **1**

- **0**: The function of auditing the SET operation is disabled.
- **1**: The function of auditing the SET operation is enabled.

Default value: **0**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **1**, audit logs are frequently recorded when the SET operation is frequently performed in the database. As a result, the database performance deteriorates.

audit_xid_info

Parameter description: Specifies whether to record the transaction ID of the SQL statement in the **detail_info** column of the audit log.

Parameter type: integer.

Unit: none

Value range: **0** and **1**

- **0**: The function of recording transaction IDs in the audit log is disabled.
- **1**: The function of recording transaction IDs in the audit log is enabled.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

If this function is enabled, the **detail_info** information in the audit log starts with *xid*. For example:

```
detail_info: xid=14619 , create table t1(id int);
```

If transaction IDs do not exist, **xid=NA** is recorded.

audit_rollback

Parameter description: Specifies whether to audit the rollback operation. This parameter can be set at the PDB level.

Parameter type: integer

Unit: none

Value range: **0** and **1**

- **0**: Rollback is not audited.
- **1**: Rollback is audited.

Default value: **0**. In the PDB scenario, if this parameter is not set, the global setting is inherited.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this function is enabled, rollback operations will be recorded in the audit log.

audit_sql_error

Parameter description: Specifies whether to audit SQL statements that fail to be executed. This parameter can be set at the PDB level.

Parameter type: integer

Unit: none

Value range: **0** and **1**

- **0**: The function of auditing SQL statements that fail to be executed is disabled.
- **1**: The function of auditing SQL statements that fail to be executed is enabled.

Default value: **0**. In the PDB scenario, if this parameter is not set, the global setting is inherited.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this function is enabled, SQL statements that fail to be executed will be recorded in the audit log.

6.3.23 Transaction Monitoring

The automatic rollback transaction can be monitored and its statement problems can be located by setting the transaction timeout warning. In addition, the statements with long execution time can also be monitored.

transaction_sync_naptime

Parameter description: For data consistency, when the local transaction's status differs from that in the snapshot of GTM, other transactions will be blocked. You need to wait for a few minutes until the transaction status of the localhost is consistent with that of the GTM. The gs_clean tool is automatically triggered for cleansing when the waiting period on the CN exceeds that of **transaction_sync_naptime**. The tool will shorten the blocking time after it completes the cleansing.

Parameter type: integer.

Unit: second

Value range: 0 to 2147483

NOTE

If the value of this parameter is set to **0**, gs_clean will not be automatically called for the cleansing before the blocking arrives the duration. Instead, the gs_clean tool is called by **gs_clean_timeout**. The default value is 5 minutes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **30**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the time for the CN to trigger clearing is prolonged, and the blocking time in the case of inconsistency is longer.

transaction_sync_timeout

Parameter description: For data consistency, when the local transaction's status differs from that in the snapshot of GTM, other transactions will be blocked. You need to wait for a few minutes until the transaction status of the localhost is consistent with that of the GTM. An exception is reported when the waiting duration on the CN exceeds the value of **transaction_sync_timeout**. Roll back the transaction to avoid system blocking due to long time of thread response failures (for example, sync lock).

Parameter type: integer.

Unit: second

Value range: 0 to 2147483

NOTE

- If the value is **0**, no error is reported when the blocking times out or the transaction is rolled back.
- If the value is not 0, the value of this parameter must be greater than that of **gs_clean_timeout**. Otherwise, unnecessary transaction rollback will probably occur due to a block timeout caused by residual transactions that have not been deleted by gs_clean on the DN.

Default value: **600** (10 minutes)

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.24 GTM Parameters

GTM parameters can be set in the **gtm.conf** file or using gs_guc.

nodename

Parameter description: Specifies the name of the primary or standby GTM.

Parameter type: string.

Unit: none

Value range: a string, which complies with the identifier naming convention

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **""**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

port

Parameter description: Specifies the host port number listened by the primary or standby GTM.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647. The recommended value range is 1024 to 65535.

Default value: **6666**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_file

Parameter description: Specifies a log file name.

Parameter type: string.

Unit: none

Value range: a string, which complies with the identifier naming convention

Default value: **"gtm-***%Y-%m-%d_%H%M%S***.log"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

active_host

Parameter description: Specifies the IP address of a target GTM. For the primary GTM, it is the IP address of the standby GTM; for the standby GTM, it is the IP address of the primary GTM. The IP address can be an IPv4 or IPv6 address.

Parameter type: string.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: a string, which complies with the identifier naming convention

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

local_host

Parameter description: Specifies the HA local address. Set this parameter based on the cluster configuration file. You do not need to manually set this parameter. The IP address can be an IPv4 or IPv6 address.

Parameter type: string.

Unit: none

Value range: a string, which complies with the identifier naming convention

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

active_port

Parameter description: Specifies the port number of the target GTM server.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647. The recommended value range is 1024 to 65535.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impact of improper settings: This parameter is specified in the configuration file during installation. Do not modify this parameter unless necessary. Otherwise, database communications will be affected after the modification.

local_port

Parameter description: Specifies the local port for HA.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647. The recommended value range is 1024 to 65535.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impact of improper settings: This parameter is specified in the configuration file during installation. Do not modify this parameter unless necessary. Otherwise, database communications will be affected after the modification.

standby_connection_timeout

Parameter description: Specifies the timeout interval between the primary and standby GTMs. A larger value enhances the fault tolerance capability of the network between the primary and standby GTMs, but increases the duration for reporting disconnection between them.

Parameter type: integer.

Unit: seconds

Value range: 5 to 2147483647

Default value: **5**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impact of improper settings: If the value is too large, disconnection faults may not be detected in a timely manner.

keepalives_count

Parameter description: Specifies the maximum number of unresponsive keepalive signals that can be waited before the GTM server is disconnected from the client on an OS that supports the **TCP_KEEPCNT** socket option. This parameter takes effect only on the standby GTM.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **0**, indicating that the connection is immediately broken if no keepalive signal from the client is received by the GTM.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

keepalives_idle

Parameter description: Specifies the interval for sending keepalive signals if no data is sent after the GTM server is connected to the client on the OS that supports the **TCP_KEEPIDLE** socket option.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

keepalives_interval

Parameter description: Specifies the interval at which the GTM server sends active signals on the OS that supports the **TCP_KEEPINTVL** socket option.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

synchronous_backup

Parameter description: Specifies whether to enable synchronization for backing up data to the standby GTM.

Parameter type: enumerated type.

Unit: none

Value range:

- **on**: Synchronization is enabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **off**: Synchronization is disabled.
- **auto**: Automatic synchronization is enabled.

Default value: **auto**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

query_memory_limit

Parameter description: Specifies the limit of memory available for queries. This parameter applies only to the default resource group. For other resource groups, the memory available for queries is not limited.

Parameter type: floating-point.

Unit: none

Value range: 0.0 to 1.0

Default value: **0.25**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

wlm_max_mem

Parameter description: Specifies the maximum memory for GTM execution.

Parameter type: integer.

Unit: MB

Value range: 512 to 2147483647

Default value: **2048**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

config_file

Parameter description: Specifies the name of a GTM configuration file. Only the SYSADMIN user can access this parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: string.

Unit: none

Value range: valid file name.

Default value: **"gtm.conf"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

data_dir

Parameter description: Specifies the GTM data file directory.

Parameter type: string.

Unit: none

Value range: valid path of a directory.

Default value: **""**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

listen_addresses

Parameter description: Specifies the TCP/IP address of the client for a server to listen on. The IP address can be an IPv4 or IPv6 address.

Parameter type: string.

Unit: none

Value range:

- Host name or IP address. Multiple values are separated with commas (,). IPv4 and IPv6 addresses are supported.
- An asterisk (*), indicating all IP addresses.
- If the parameter is not specified, the server does not listen on any IP address. In this case, only UDS can be used for database connections.

Default value: **"*"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_directory

Parameter description: Specifies the directory for storing log files when **logging_collector** is set to **on**. The value can be an absolute path, or relative to the data directory.

NOTICE

- If the value of **log_directory** in the configuration file is an invalid path (that is, the user does not have the permission to read or write this path), the cluster cannot be restarted.
- If the value of **log_directory** is changed to a valid path (that is, the user has the permission to read and write this path), logs are generated in the new path. If the specified path is invalid, log files are generated in the last valid path and the database running is not affected. The invalid value is still written into the configuration file.

Parameter type: string.

Unit: none

Value range: valid path of a directory.

Default value: **gtm_log**, indicating that server logs will be generated in the **gtm_log/** directory under the data directory.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_min_messages

Parameter description: Specifies the level of messages written to GTM log files in the server log folder. When a message of the same or higher level is generated, the message is recorded in the log file. Therefore, a lower level indicates that more messages are recorded in the run log of the server.

Parameter type: enumerated type.

Unit: none

Value range: The message levels are **debug5**, **debug4**, **debug3**, **debug2** (**debug**), **debug1**, **info**, **notice**, **warning**, **error**, **log**, **fatal**, and **panic** in ascending order. For details about the parameters, see **Table 6-23**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The message level **debug** is equivalent to **debug2**.
- For **log_min_messages**, the log message level is higher than **error** and lower than **fatal**.

Default value: **warning**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

alarm_component

Parameter description: Certain alarms are suppressed during alarm reporting. That is, the same alarm will not be repeatedly reported by an instance within the period specified by **alarm_report_interval**. Its default value is **10s**. In this case, the parameter specifies the location of the alarm component that is used to process alarm information.

Parameter type: string.

Unit: none

Value range:

- If **--alarm-type** in the gs_preinstall script is set to **5**, no third-party component is connected and alarms are written into the **system_alarm** log. In this case, the value of **alarm_component** is **/opt/huawei/snas/bin/snas_cm_cmd**.
- If **--alarm-type** in the gs_preinstall script is set to **1**, a third-party component is connected. In this case, the value of **alarm_component** is the absolute path of the executable program of the third-party component.

Default value: **"/opt/huawei/snas/bin/snas_cm_cmd"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

alarm_report_interval

Parameter description: Specifies the interval at which an alarm is reported.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647

Default value: **10**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

standby_only

Parameter description: Specifies whether to forcibly synchronize information to standby nodes. In the mode with one primary and multiple standby nodes, information is only forcibly synchronized to the ETCD.

Parameter type: integer.

Unit: none

Value range:

- **0**: Information is not forcibly synchronized to standby nodes.
- **1**: Information is forcibly synchronized to standby nodes.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_max_trans

Parameter description: Specifies the maximum number of connections accepted by the GTM. You are advised not to change the value.

Parameter type: integer.

Unit: none

Value range: 256 to 200000

Default value: **8192**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If you have to, set this parameter to a value no less than the maximum number of connections plus 100.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_connect_control

Parameter description: Specifies whether the GTM connection control is enabled to check whether a connection IP address is within the cluster.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Checks whether a connection IP address is within the cluster. If it is not, the access is rejected.
- **off**: Does not check whether a connection IP address is within the cluster.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_authentication_type

Parameter description: Specifies the port authentication mode of the GTM module.

Parameter type: enumerated type.

Unit: none

Value range:

- **trust**: Port authentication is not configured.
- **gss**: Kerberos port authentication is used.

> **NOTICE**
>
> Note that you can change the value to **gss** only after the Kerberos server and client are successfully installed. Otherwise, the GTM cannot communicate properly, affecting the cluster status.

Default value: **trust**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_krb_server_keyfile

Parameter description: Specifies the location of the key file on the Kerberos server. The value must be an absolute path. The file is usually stored in the *${GAUSSHOME}*/**kerberos** directory and ends with keytab. The file name is the same as the name of the user who runs the cluster. This parameter is used

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

together with **gtm_authentication_type**. If **gtm_authentication_type** is changed to **gss**, this parameter must be configured as the correct path. Otherwise, the cluster status will be affected.

Parameter type: string.

Unit: none

Value range: valid file path.

Default value: **""**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_gtm_free

Parameter description: Specifies whether the GTM-free mode is enabled. In large concurrency scenarios, the snapshots delivered by the GTM increase in number and size. The network between the GTM and the CN becomes the performance bottleneck. The GTM-free mode is used to eliminate the bottleneck. In this mode, the CN communicates with DNs instead of the GTM. The CN sends queries to each DN, which locally generates snapshots and XIDs, ensuring external write consistency but not external read consistency.

NOTICE

When the GTM-free mode is used, you are advised to set **application_type** to **perfect_sharding_type** so that you can find SQL statements that may cause data inconsistency. Otherwise, the system does not intercept statements that may cause data inconsistency.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The GTM-free mode is enabled and the cluster ensures eventual read consistency.
- **off**: The GTM-free mode is disabled.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Data inconsistency may occur in GTM-free mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

gtm_option

Parameter description: Specifies the GTM mode. This parameter must be configured on all GTMs, CNs, and DNs. The value must be the same. There are three GTM modes: GTM, GTM-lite, and GTM-free. The GTM and GTM-lite modes take effect only when the **enable_gtm_free** parameter is set to **off**. The current version does not support switching between different GTM modes for installed clusters. For details about the three GTM modes, see "Feature Usage Guidelines > GTM > GTM Modes" in *Developer Guide*.

Parameter type: integer.

Unit: none

Value range: 0 to 2

- **0**: GTM mode.
- **1**: GTM-lite mode.
- **2**: GTM-free mode.

Default value: **1**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

csn_sync_interval

Parameter description: Specifies the interval for synchronizing CSN between the primary and standby GTMs.

Parameter type: integer.

Unit: seconds

Value range: 1 to 2147483647

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

restore_duration

Parameter description: Specifies the restoration duration of XID or CSN on the GTM, that is, the number of XIDs or CSNs for restoration.

Parameter type: integer.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 1000000 to 2147483647

Default value: **1000000**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_enable_threadpool

Parameter description: Specifies whether to enable the GTM thread pool function. The setting takes effect only after the GTM is restarted.

Parameter type: Boolean.

Unit: none

Value range:

- **true**: The GTM thread pool function is enabled.
- **false**: The GTM thread pool function is disabled.

Default value: **true**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_num_threads

Parameter description: Specifies the number of worker threads in the thread pool when the thread pool function **gtm_enable_threadpool** is enabled.

Parameter type: integer.

Unit: none

Value range: 0 to 16384. The value is related to the size of **gtm_max_trans** and cannot exceed the result of (Value of **gtm_max_trans** – 1 – Number of auxiliary threads). The number of auxiliary threads is **2** in the current version.

Default value: **1024**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

check_gtm_primary_time

Parameter description: Specifies the interval at which GTM checks the consistency between its primary node and ETCD primary node.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647. The value **0** indicates that the detection function is disabled.

Default value: **2**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

seq_cache_coefficient

Parameter description: Specifies the sequence cache coefficient on the GTM. The cache coefficient is used to calculate the sequence recovery value.

Parameter type: integer.

Unit: none

Value range: 3 to 20000

Default value: **20000**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. The minimum value cannot be less than the number of nodes.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_dcf_run_mode

Parameter description: Specifies the DCF running mode of the DCC when GTM uses DCC.

Parameter type: integer.

Unit: none

Value range: 0 to 2. The value **0** indicates that the election function is enabled, the value **1** indicates that the manual election is performed, and the value **2** indicates that the election function is disabled.

Default value: **2**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: This parameter is set by the CM. You are advised not to manually set or change the value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing. Unnecessary switchovers may occur or the primary and standby status may be abnormal.

gtm_dcc_id

Parameter description: Specifies the DCC ID of the instance when GTM uses DCC.

Parameter type: integer.

Unit: none

Value range: 1 to 127

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is set and maintained by the OM and cannot be modified.

Risks and impacts of improper settings: none

gtm_dcf_extreme_mode

Parameter description: Specifies whether to enable the maximum availability mode for DCF of DCC when GTM uses DCC.

Parameter type: integer.

Unit: none

Value range: 0 to 2. The value **0** indicates the maximum protection mode, the value **1** indicates the maximum availability mode, and the value **2** indicates the maximum performance mode.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is set by the CM. You are advised not to manually set or change the value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing. Service consistency may be affected.

gtm_dcf_static_leader_timeout

Parameter description: Specifies the timeout period for GTM to wait for the static primary DCF node or the original primary node to be promoted to primary when DCC is used. This parameter is mainly used in the cluster startup phase.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is set by the CM or OM. You are advised not to manually set or change the value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing. The cluster may fail to be started.

gtm_dcc_config

Parameter description: Specifies the DCC cluster configuration when GTM uses DCC.

Parameter type: string.

Unit: none

Value range: a JSON string that contains valid information.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is set and maintained by the OM and cannot be modified.

Risks and impacts of improper settings: none

gtm_dcc_data_dir

Parameter description: Specifies the DCC data directory when GTM uses DCC.

Parameter type: string.

Unit: none

Value range: a valid absolute path.

Default value: **""**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is set and maintained by the OM and cannot be modified.

Risks and impacts of improper settings: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

gtm_dcf_sync_nodes

Parameter description: Specifies the synchronization mode between DCCs when GTM uses DCC.

Parameter type: string.

Unit: none

Value range:

- **ANY num_sync:standby1,standby2**: specifies the policy for replicating the standby node. **ANY** *N*:**1,2,**... indicates that logs are synchronized to any *N* standby nodes after the colon (:).
- **FIRST num_sync:standby1,standby2**: specifies the policy for replicating the standby node. **FIRST** *N*:**1,2,**... indicates that logs are synchronized based on the sequence following the colon (:).

num_sync indicates the number of logs to be synchronized to the standby node, and **standby** indicates the standby node ID.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is set by the CM or OM. You are advised not to manually set or change the value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing. Unnecessary switchovers may occur or the primary and standby status may be abnormal.

gtm_dcf_candidate_nodes

Parameter description: Specifies the list of primary node candidates of DCC when GTM uses DCC.

Parameter type: string.

Unit: none

Value range: numbers separated by commas (,).

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is set by the CM or OM. You are advised not to manually set or change the value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing. Unnecessary switchovers may occur or the primary and standby status may be abnormal.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

gtm_dcc_log_level

Parameter description: Specifies the DCC log printing level when GTM uses DCC.

Parameter type: string.

Unit: none

Value range:

- **"NONE"**: The log function is disabled. It cannot be used together with the following log levels.
- **"RUN_ERR|RUN_WAR|RUN_INF|DEBUG_ERR|DEBUG_WAR|DEBUG_INF|TRACE|PROFILE|OPER"**: The log function of specified levels is enabled.

Default value: **"RUN_ERR|RUN_WAR|DEBUG_ERR|OPER|RUN_INF|PROFILE"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gtm_ddb

Parameter description: Specifies whether GTM uses ETCD or DCC.

Parameter type: string.

Unit: none

Value range: **etcd** or **dcc**.

Default value: **etcd**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is set by OM during cluster installation and cannot be manually modified after installation.

Risks and impacts of improper settings: none

6.3.25 Upgrade Parameters

IsInplaceUpgrade

Parameter description: Specifies whether an upgrade is ongoing. This parameter is an upgrade parameter and cannot be modified. Only the SYSADMIN user can access the parameter.

Parameter type: Boolean

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: An upgrade is ongoing.
- **off**: No upgrade is ongoing.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

inplace_upgrade_next_system_object_oids

Parameter description: Specifies the OID of a new system object during the in-place upgrade. This parameter is used for upgrade and cannot be modified by users.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **""**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

om_operation_mode

Parameter description: Specifies whether the cluster is in the O&M or running state. This parameter is used for internal component interaction and cannot be used by users.

Parameter type: enumerated type.

Unit: none

Value range:

- **upgrade_system_object_mode**: The upgrade is in the system object update phase.
- **running_mode**: The normal running phase of the upgrade.

Default value: **running_mode**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **upgrade_system_object_mode**, the database name cannot be renamed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

upgrade_mode

Parameter description: Specifies the upgrade mode.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

- **0**: No upgrade is ongoing.
- **1**: A local upgrade is ongoing.
- **2**: A grayscale upgrade is ongoing.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is used for upgrade. You are advised not to modify it. You are advised to use the default value.

Risks and impacts of improper settings: Do not modify this parameter during the upgrade. Otherwise, the upgrade may fail and rollback cannot be performed. In non-upgrade scenarios, modifying this parameter may cause database usage errors or other unpredictable consequences.

NOTE

Special case: When the gray upgrade is used, if the major version upgrade policy is selected, that is, the upgrade script needs to be executed and the binary package needs to be replaced, the value of **upgrade_mode** is set to **2**; if the minor version upgrade policy is selected, that is, only the binary package needs to be replaced, the value of **upgrade_mode** is not set to **2**.

6.3.26 Miscellaneous Parameters

enable_default_ustore_table

Parameter description: Specifies whether the Ustore is used by default. This parameter is valid only when **enable_ustore** is enabled. If this parameter is set to **on**, all created tables are Ustore tables by default. If this parameter is set to **off**, all created tables are Astore tables by default. In upgrade scenarios, the default value of this parameter ensures compatibility. That is, the default value after the upgrade is the original default value.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The Ustore is used by default.
- **off**: The Ustore is not used by default.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: When using Ustore tables, you must enable the **track_counts** and **track_activities** parameters. Otherwise, space bloat may occur.

enable_ustore

Parameter description: Specifies whether to enable Ustore. If this parameter is set to **on**, Ustore tables can be created.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The Ustore is enabled.
- **off**: The Ustore is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is used for Ustore version control. You are advised not to change the value.

Risks and impacts of improper settings: When using Ustore tables, you must enable the **track_counts** and **track_activities** parameters. Otherwise, space bloat may occur.

enable_segment_datafile_preallocate

Parameter description: Specifies whether to allocate disk space immediately during segment-page file name extension.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The immediate disk space allocation mode is enabled. During file expansion, fallocate is preferentially used to allocate disk space immediately. If the system does not support fallocate, the byte-by-byte zero write mode is used to apply for disk space.
- **off**: The immediate disk space allocation mode is disabled. In this mode, disk space is not allocated immediately when a file is expanded, and the file increases in hole mode.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

consistency_check_module

Parameter description: Specifies the module that requires consistency check.

Parameter type: string.

Unit: none

Value range: If the module is enabled (**on**), consistency check is performed. If the module is disabled (**off**), consistency check is not performed. You can use SHOW consistency_check_module to view the status.

```
ALL,on(),off(INDEX)
```

Currently, only the INDEX module supports the verification. The INDEX module verifies the consistency between index tuples and heap table tuples during the B-tree index scan of the Astore table.

Default value: **'off(ALL)'**. That is, the verification of all modules is disabled by default.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

1. First, you can use SHOW consistency_check_module to check which modules support verification. The output of the verification module is as follows:
   ```
   gaussdb=# SHOW consistency_check_module;
    consistency_check_module
   --------------------------
    ALL,on(),off(INDEX)
   (1 row)
   ```
2. Modules that can be verified are identified by uppercase letters, and the special ID **ALL** is used for setting all modules. You can use **on** or **off** to specify whether to perform verification. For example, to verify the consistency between indexes and heap tables, run the following commands:
   ```
   gaussdb=# SET consistency_check_module='on(INDEX)';
   SET
   gaussdb=# SHOW consistency_check_module;
    consistency_check_module
   --------------------------
    ALL,on(INDEX),off()
   (1 row)
   ```
   The command output shows that the INDEX check is enabled.
3. The **ALL** identifier can be used to quickly enable or disable the verification of all modules.
   ```
   gaussdb=# SET consistency_check_module='off(ALL)';
   SET
   gaussdb=# SHOW consistency_check_module;
    consistency_check_module
   --------------------------
    ALL,on(),off(INDEX)
   (1 row)

   gaussdb=# SET consistency_check_module='on(ALL)';
   SET
   gaussdb=# SHOW consistency_check_module;
    consistency_check_module
   --------------------------
   ```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 ALL,on(INDEX),off()
(1 row)
```

Setting suggestion: Enable the verification of the corresponding module as required.

Risks and impacts of improper settings: If this function is enabled, consistency check is performed during SQL execution, which takes extra time or affects performance. Exercise caution when enabling this function.

reserve_space_for_nullable_atts

Parameter description: Specifies whether to reserve space for the nullable attribute of an Ustore table.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Space is reserved for the nullable attribute of an Ustore table.
- **off**: Space is not reserved for the nullable attribute of an Ustore table.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

server_version

Parameter description: Specifies the server version number in the character string format.

Parameter type: string.

Unit: none

Value range: **"9.2.4"**

Default value: **"9.2.4"**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified. This parameter is used only by internal database tools. You are advised not to use it.

Setting suggestion: This parameter is not recommended. If you need to query the kernel version, you can obtain it by using the opengauss_version() function.

Risks and impacts of improper settings: none

server_version_num

Parameter description: Specifies the server version number in the integer format.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: none

Value range: **90204**

Default value: **90204**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified. This parameter is used only by internal database tools. You are advised not to use it.

Setting suggestion: This parameter is not recommended. If you need to query the kernel version, you can obtain it by using the opengauss_version() function.

Risks and impacts of improper settings: none

block_size

Parameter description: Specifies the block size of the current database.

Parameter type: integer.

Unit: none

Value range: **8192**

Default value: **8192**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

segment_size

Parameter description: Specifies the segment file size of the current database.

Parameter type: integer.

Unit: 8 KB

Value range: **131072**

Default value: **131072**, that is, 1 GB.

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

max_index_keys

Parameter description: Specifies the maximum number of index keys supported by the current database.

Parameter type: integer.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range: **32**

Default value: **32**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

integer_datetimes

Parameter description: Specifies whether the date and time are in the 64-bit integer format.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The date and time format in 64-bit integer format is supported.
- **off**: The date and time format in 64-bit integer format is not supported.

Default value: **on**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

lc_collate

Parameter description: Specifies the string collation locale of the current database.

Parameter type: string.

Unit: none

Value range: a string.

Default value: Determined by the configuration during the cluster installation and deployment.

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

lc_ctype

Parameter description: Specifies the locale settings for character types and case conversion in the current database. For example, it specifies what a letter and its upper-case equivalent are.

Parameter type: string.

Unit: none

Value range: a string.

Default value: Determined by the configuration during the cluster installation and deployment.

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

max_identifier_length

Parameter description: Specifies the maximum identifier length.

Parameter type: integer.

Unit: none

Value range: **63**

Default value: **63**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

server_encoding

Parameter description: Specifies the server-side encoding (character set) of the current database.

By default, gs_initdb will initialize the setting of this parameter based on the system environment during database creation. You can also run the **locale** command to check the current configuration environment.

Parameter type: string.

Unit: none

Value range: a string.

Default value: Determined by the system environment when the database is created.

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

datanode_heartbeat_interval

Parameter description: Specifies the interval for sending heartbeat messages between heartbeat threads.

Parameter type: integer.

Unit: milliseconds

Value range: 1000 to 60000

Default value: **1000** (that is, 1s)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1000** without a unit, **datanode_heartbeat_interval** indicates 1000 ms. If the value is **1s**, **datanode_heartbeat_interval** indicates 1s. The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Set this parameter to a value less than or equal to **wal_receiver_timeout**/2.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

dfs_partition_directory_length

Parameter description: Specifies the maximum directory name length for the partition directory of a table partitioned by VALUE in the HDFS.

Parameter type: integer.

Unit: none

Value range: 92 to 7999

Default value: **512**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_concurrent_autonomous_transactions

Parameter description: Specifies the maximum number of autonomous transaction connections, that is, the maximum number of concurrent autonomous transactions executed at the same time. If this parameter is set to **0**, autonomous transactions cannot be executed.

Parameter type: integer.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 0 to 10000. The theoretical maximum value is **10000**, and the actual maximum value is a dynamic value. The calculation formula is 262143 – **job_queue_processes** – **autovacuum_max_workers** – **max_inner_tool_connections** – **max_connections** – **AUXILIARY_BACKENDS** – **AV_LAUNCHER_PROCS**. The values of **job_queue_processes**, **autovacuum_max_workers**, **max_inner_tool_connections**, and **max_connections** depend on the settings of the corresponding GUC parameters. **AUXILIARY_BACKENDS** indicates the number of reserved auxiliary threads and is fixed at **20**. **AV_LAUNCHER_PROCS** indicates the number of launcher threads reserved for autovacuum and is fixed at **2**.

Default value:

- Independent deployment: **160** (128-core CPU/1024 GB memory); **80** (60-core CPU/480 GB memory); **40** (32-core CPU/256 GB memory); **20** (16-core CPU/128 GB memory); **10** (8-core CPU/64 GB memory, 4-core CPU/32 GB memory, 4-core CPU/16 GB memory)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Suggestion: Set this parameter based on actual service requirements and hardware configurations. It is recommended that this parameter be set to a value less than or equal to 1/10 of **max_connections**. If you only increase the value of this parameter but do not adjust the memory parameters in the same proportion, the memory may be insufficient and the error message "memory is temporarily unavailable" is displayed when the service load is heavy.

Risks and impacts of improper settings: If the value range of this parameter is changed during the upgrade and the value is changed before the commit operation, you need to change the value range to the value allowed before the upgrade if you roll back the upgrade. Otherwise, the database may fail to be started.

cluster_run_mode

Parameter description: Specifies whether a CN or DN belongs to the primary or standby cluster in a dual-cluster DR scenario. The default value is used for a single cluster.

Parameter type: enumerated type

Unit: none

Value range:

- **cluster_primary**: The node is in the primary cluster.
- **cluster_standby**: The node is in the standby cluster.

Default value: **cluster_primary**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Mark the parameter used by the standby cluster for DR. You are advised not to set the parameter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: After the modification, the standby cluster for DR may be abnormal.

max_resource_package

Parameter description: Specifies the maximum number of threads that each DN can run concurrently on an acceleration database instance.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_gpi_auto_update

Parameter description: Determines whether global indexes are updated by default in partition DDL commands.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Global indexes are updated regardless of whether the partition DDL commands contain the UPDATE GLOBAL INDEX clause.
- **off**: Global indexes are not updated by default. They are updated during the DROP/TRUNCATE PARTITION operation only when the partition-level DDL commands specify the UPDATE GLOBAL INDEX clause, or when the **enable_drop_truncate_partition_keep_gpi** parameter is enabled.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements.

Risks and impacts of improper settings: If this parameter is set to **off** and the partition DDL does not contain the UPDATE GLOBAL INDEX clause, the global index is unavailable. This parameter must be set to the same value on CNs and DNs. Otherwise, the global index function may be abnormal.

enable_drop_truncate_partition_keep_gpi

Parameter description: Specifies whether the global index remains valid upon the DROP/TRUNCATE PARTITION operation on a partitioned table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The global index remains valid. In this case, the global index is always valid, regardless of whether the DROP/TRUNCATE PARTITION statement contains the UPDATE GLOBAL INDEX clause. This priority is higher than that of the **enable_gpi_auto_update** parameter and the UPDATE GLOBAL INDEX clause explicitly specified by the user.
- **off**: The global index is not always valid. In this case, the global index is always valid only when the DROP/TRUNCATE PARTITION statement contains the UPDATE GLOBAL INDEX clause or the **enable_gpi_auto_update** parameter is enabled.

Default value:

- For a newly installed database, the default value is **on**.
- For an upgraded database, the default value is **off**.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements.

Risks and impacts of improper settings: If this parameter is set to **off**, the global index becomes invalid when the DROP/TRUNCATE PARTITION statement does not contain the UPDATE GLOBAL INDEX clause and the **enable_gpi_auto_update** parameter is disabled. This parameter must be set to the same value on CNs and DNs. Otherwise, the global index function may be abnormal.

enable_gpi_fast_prune

Parameter description: Specifies whether to vacuum dead tuples generated by previous DDL operations during DML operations on the global partitioned index (GPI) of a Ustore table.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The vacuum mechanism is enabled. Dead tuples generated by DDL operations are vacuumed not only using the autovacuum thread but also when the index page space is insufficient during DML operations.
- **off**: The vacuum mechanism is disabled. Dead tuples generated by DDL operations are vacuumed only using the autovacuum thread.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements. You are advised to enable this parameter in service scenarios where high-frequency DDL/DML operations exist and GPIs are used. You are advised to disable this

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

parameter in service scenarios where GPIs are not used or partitioned table DDL operations are not performed.

Risks and impacts of improper settings: If this parameter is set to **off**, dead tuples generated by DDL operations are only vacuumed using the autovacuum thread. As a result, the GPI space may expand due to accumulation of invalid tuples, affecting query performance.

enable_cluster_resize

Parameter description: Specifies the unique identifier of the scale-out tool in the kernel. If this parameter is enabled, the current session is connected to the scale-out tool.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The current session is connected to the scale-out tool.
- **off**: The current session is not connected to the scale-out tool.

Default value: **off**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**. Only the redistribution user and initial user can set this parameter in a session.

Setting suggestion: This parameter is used for internal O&M. Do not set it to **on** unless absolutely necessary.

Risks and impacts of improper settings: If this parameter is forcibly set to **on**, the kernel considers that the current session is a scale-out tool and can execute scale-out-related system functions or DDL APIs, which may cause misuse.

change_cluster_mode

Parameter description: Specifies whether the cluster is in the mode switchover process. Mode switchover refers to: A cluster with one primary and two standby nodes is switched to a cluster with one primary node, one standby node, and one log; a cluster with one primary node, one standby node, and one log is switched to a cluster with one primary and two standby nodes; standby DNs and log DNs in a cluster with one primary node, one standby node, and one log are switched to each other.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

- The value **0** indicates that the cluster mode is not being switched.
- The value **1** indicates that the cluster mode is being switched.
- Other values are meaningless and have the same effect as the value **0**.

Default value: **0**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter specifies whether the read function of the standby node in a distributed system is available. If this parameter is set to **1**, the read function of the standby node in a distributed system cannot be used. Exercise caution when setting this parameter.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_partrouting_optimization

Parameter description: Specifies whether to optimize the insertion of partitioned tables.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The insertion of partitioned tables is optimized. For the INSERT SELECT statement, if the SELECT statement contains a constant partition key, partition routing needs to be performed only once for the INSERT statement, improving performance.
- **off**: The insertion of partitioned tables is not optimized. Before inserting each piece of data, you need to perform partition routing to determine the partitioned table to be inserted.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the performance optimization effect brought by this parameter is not affected in some scenarios where data is inserted into partitioned tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

Constraints for optimizing the insertion of partitioned tables:

- Only the INSERT SELECT statement is supported.
- In INSERT INTO ta SELECT FROM tb, the column values in the SELECT result set must be constants for all partition key values inserted into table **a**.
  - INSERT INTO ta SELECT c,d FROM b WHERE tb.c='1' or SELECT '1' as c, d FROM tb: The result column **tb.c** is a constant.
  - INSERT INTO ta SELECT c,d FROM b WHERE tb.c=func('1');: If func() is neither a volatile function nor a stable/immutable function that contains non-constant parameters, **tb.c** can be determined as a constant.
  - If the column value is the return value of an aggregate function, for example, INSERT INTO ta SELECT count(c),d FROM b WHERE tb.c='1' group by c, d;, the count(c) cannot be determined as a constant.
- This optimization is not supported when the partition key is an auto-increment column, a generated column, or a column with a default value.
- The UPSERT clause is not supported.
- Tables that contain BEFORE and INSTEAD triggers are not supported because the data to be inserted may be changed.

enable_cross_partition_dead_row_cache

Parameter description: Specifies whether to enable cross-partition dead row vacuuming. This feature is mainly used for large Astore partitioned tables. It combines and vacuums dead tuple TIDs in different partitions to reduce the GPI access frequency and decouple the vacuum duration from the number of partitions. The vacuum duration is related only to the data volume. That is, when the data volume is the same, the time required for vacuuming a partitioned table is close to or the same as that for vacuuming an ordinary table. This feature is optimized in scenarios where there are a large number of partitions and GPIs.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Cross-partition dead row vacuuming is enabled.
- **off**: Cross-partition dead row vacuuming is disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this function is disabled, the VACUUM/AUTOVACUUM operation of the Astore table with GPI may be slow.

enable_art_tid_cache

Parameter description: Specifies whether to enable the ART cache for dead row TIDs. The Adaptive Radix Tree structure can be used to store TIDs to reduce the memory consumption of VACUUM. In this way, more TIDs can be cached with the same memory overhead, reducing the number of index scans on large tables. In

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

addition, the time complexity of searching for TIDs is reduced, improving the index vacuum efficiency.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The TID of the dead row cached in the ART is enabled.
- **off**: The TID of the dead row cached in the ART is disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this function is disabled, the VACUUM/AUTOVACUUM operation of the Astore table may be slow.

storage_debug_parameter

Parameter description: Specifies the debugging parameters of the storage module. This parameter is used only in the debug version.

Parameter type: string.

Unit: none

Value range: a string of storage configuration parameters separated by semicolons (;), for example, **"oid_threshold=100000;xid_stepsize=10"**.

- **oid_threshold**: maximum number of OIDs that can be applied for. The value ranges from 100000 to 4200000000. The default value is **4200000000**.
- **oid_stepsize**: step size for applying for OIDs. The value ranges from 1 to 4200000000. The default value is **1**.
- **xid_stepsize**: step size for applying for transaction XIDs. The value ranges from 1 to 100000. The default value is **1**.
- **toast_tuple_threshold**: tuple width required for creating a TOAST table. The value ranges from 100 to 7000. The default value is **2000**.

Default value: **""**.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_unique_checking_of_unusable_index

Parameter description: Specifies whether to block the insert and update operations when there is an unusable unique index in the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean

Unit: none

Value range:

- **on**: If a table contains an UNUSABLE unique index, an INSERT statement and an UPDATE statement that involves the index will report an error, prompting you to process the UNUSABLE unique index first.
- **off**: If a table contains an UNUSABLE unique index, INSERT and UPDATE statements that involve the index will not report errors.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on service requirements. You are advised to set it to **on**.

Risks and impacts of improper settings: If this parameter is enabled, a large number of errors may be reported to ensure the UNIQUE constraint. If this parameter is disabled, duplicate data in the unique index column is inserted into the table. As a result, the index cannot be rebuilt.

NOTE

If this parameter is enabled:

- For an Astore table, an error is reported when an index column of a non-UNUSABLE index is updated because a new tuple is generated.
- For a Ustore non-partitioned table, no error is reported when the index column of a non-UNUSABLE index is updated. If the update is not performed in place, an error is reported when the index column of a non-UNUSABLE index is updated.
- For a Ustore partitioned table, when the UNUSABLE unique index is a global index, an error is reported if the new data and old data are not in the same partition after the partition key is updated because a new tuple is generated.
- For a Ustore partitioned table, if the UNUSABLE unique index is a local index that contains the partition key, an error is reported when the partition key is updated.

enable_verify_datafiles

Parameter description: Specifies whether to enable the verification function for data file mis-deletion. This function can only be used to detect mis-deletion of data files (files in the **base**, **global**, **pg_tblspc**, and **undo** directories) when the cluster is running properly. Mis-deletion detection includes only two scenarios: files are deleted by mistake (**rm**) or renamed (**mv**). There is a time difference between the mis-deletion of data files and the detection. If the database is shut down during this period, the mis-deleted files cannot be detected.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The verification function for data file mis-deletion is enabled.
- **off**: The verification function for data file mis-deletion is disabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set the value based on service requirements. After this function is enabled, the list of properly deleted files needs to be maintained and traversed for verification, which may affect the performance. In addition, the detection capability depends on the inotify capability of the system kernel. The maximum number of monitored directories (including subdirectories at each level) is specified by **fs.inotify.max_user_watches**, and the maximum number of messages that can be concurrently processed is specified by **fs.inotify.max_queued_events**. If the number of files to be deleted at the same time exceeds the value of **max_queued_events**, the message queue may overflow and the file deletion alarm may be lost.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

disable_indirect_table_updates

Parameter description: Specifies whether to prohibit multiple DML operations on the same table within a single query, including modifications performed by the main statement, CTEs, subqueries, and triggers.

This parameter is mainly used to mitigate risks associated with modifying the same table multiple times within a single query, such as execution sequence dependencies, trigger-based recursive modifications, non-deterministic results, and data consistency issues.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Multiple DML operations cannot be performed on the same table within a single query.
- **off**: Multiple DML operations can be performed on the same table within a single query.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set the value based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

- When this parameter is enabled, DML operations cannot be performed again in a row trigger for the same table. However, if the outer DML operation is INSERT and the DML operation in the trigger for the same table is also INSERT, the DML operation is allowed in this scenario.
- In the scenario where a trigger is nested for self-update, if two triggers are called on the CN and DN respectively, the verification does not take effect.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_hashbucket

Parameter description: Specifies whether to create a hash bucket table by default when CREATE TABLE is executed.

Parameter type: Boolean

Unit: none

Value range:

- **on**: A hash bucket table is created by default.
- **off**: A hash bucket table is not created by default.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Set this parameter to **on** only when hash bucket tables are required by default.

Risks and impacts of improper settings: After this parameter is set to **on**, the CREATE TABLE statement is used to create a segment-page hash bucket table instead of an ordinary page table by default. Hash bucket tables and ordinary page tables have different specifications and restrictions in terms of data storage mode, space usage, buffer usage, and other feature support. For details, see "Feature Usage Guidelines > Bucket Distributed Tables" in *Developer Guide*. Before using hash bucket tables, make sure you understand its meaning and fully verify the change in a test environment.

NOTE

- The **enable_hashbucket** parameter takes effect only on CREATE TABLE and CREATE TABLE AS statements.
- The **enable_hashbucket** parameter does not take effect in the following scenarios:
  - A table is created in the PG_CATALOG schema.
  - The initial user of the cluster executes the related table creation statement.
  - The CREATE TABLE LIKE statement is executed and **INCLUDING RELOPTIONS** or **INCLUDING ALL** is specified. (In this case, whether the new table is a hash bucket table is the same as that of the source table.)
  - The table to be created is not a persistent table (for example, a temporary table, global temporary table, or unlogged table).
  - The table to be created is not a hash distributed table (for example, a replication table or range distributed table).
  - The WITH clause in the table creation statement specifies that **segment** is **off** or **hashbucket** is **off**.
- When you set the **enable_hashbucket** parameter using method 1 or 2 in **Table 6-1**, you are advised to set the same value for all nodes in the cluster. If the parameter values on different nodes are different, whether the parameter takes effect is determined by the parameter value on the CN that delivers the table creation statement.
- Do not set this parameter to **on** during the upgrade observation period.

vacuum_truncate

Parameter description: Specifies whether to truncate consecutive idle tail pages during table vacuuming.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Consecutive idle tail pages are truncated during table vacuuming.
- **off**: Consecutive idle tail pages are not truncated during table vacuuming.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **off**, the vacuum process does not truncate idle tail pages, which may cause space expansion of Astore tables.

NOTE

- This parameter is valid only for Astore tables and is invalid for hash bucket tables and segment tables.
- If this parameter conflicts with the table-level option **vacuum_truncate**, the table-level option takes precedence.
  - If the table-level **vacuum_truncate** is set to **on** and the global parameter is set to **off**, the VACUUM TRUNCATE function is enabled for the table.
  - If the table-level **vacuum_truncate** is set to **off** and the global parameter is set to **on**, the VACUUM TRUNCATE function is disabled for the table.
  - If the table-level parameter is not set, the global parameter determines whether to enable the VACUUM TRUNCATE function.

6.3.27 Wait Event

enable_instr_track_wait

Parameter description: Specifies whether to enable real-time collection of wait event information.

Parameter type: Boolean

Unit: none

Value range:

- **on** indicates that the function of collecting wait event information is enabled.
- **off** indicates that the function of collecting wait event information is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised to set it to **on**.

Risks and impacts of improper settings: If this parameter is disabled, wait events cannot be collected in real time, which makes it difficult to locate faults in a timely manner.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

enable_instr_track_wait is restricted by **track_activities**. If **track_activities** is disabled, wait events are not recorded.

6.3.28 Query

instr_unique_sql_count

Parameter description: Specifies the maximum number of unique SQL statements that can be collected in real time.

If the value decreases, the system clears the original data and collects statistics again. If the value increases, the system does not collect statistics again.

When the number of unique SQL records generated in the system (to view the statistics, query **dbe_perf.statement** or **dbe_perf.summary_statement**) is greater than the value of **instr_unique_sql_count**, the extra unique SQL records are not collected.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647. The value **0** indicates that the function of collecting unique SQL information is disabled.

Default value: **200000**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the upper limit of the memory occupied by the dbe_perf.statement view increases. If the value is too small, the number of statements recorded in the view may be too small. As a result, no fault locating method is available, affecting the diagnosis of slow SQL statements.

instr_unique_sql_size

Parameter description: Specifies the maximum length of query information recorded by the unique SQL statements, full SQL statements, and ASP functions. If the actual length of a statement exceeds the value of this parameter, the statement will be truncated. The byte length is calculated based on the character set of the database where the query is executed.

Parameter type: integer.

Unit: bytes

Value range: 100 to 102400

Default value: **1024**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impact of improper configuration: If the value is too large, the upper limit of the memory occupied by the dbe_perf.statement view increases. If the value is too small, the statements recorded in the view may be incomplete. As a result, no fault locating method is available, affecting the diagnosis of slow SQL statements.

instr_unique_sql_combination_options

Parameter description: Specifies the configuration items of combining unique SQL statements of the same type. The value of this parameter consists of multiple configuration items separated by commas (,).

If this feature is enabled, the IDs of unique SQL statements of the same type are normalized, and the generated unique SQL strings are normalized.

Parameter type: string.

Unit: none

Value range: See **Table 6-52**.

NOTE

- When configuring the combination function, set the character string based on **Table 6-52**. Use commas (,) to separate multiple configuration items, for example, **SET instr_unique_sql_combination_options='in_clause';**.
- If this parameter is left blank, this function is disabled, for example, **SET instr_unique_sql_combination_options='';**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-52 Configuration items of the combination function

Configuration Item	Behavior Control
in_clause	Combines only fixed parameters and prepared binding parameters in the IN clause of the SELECT IN() statement. Example 1: **select * from example_table where column in (1,2,3);** Unique SQL string after combination: **select * from example_table where column in (1…n);** Example 2: **select * from example_table where column in ($1,$2,$3);** Unique SQL string after combination: **select * from example_table where column in ($1…$n);** Example 3: **select * from example_table where column in (1,2,$1,3,$2);** Unique SQL string after combination: **select * from example_table where column in (1…n,$1…$n);** Example 4: **select * from example_table where (column1, column2) in ((1, 'a'), (2, 'b'), (3, 'c'));** Unique SQL string after combination: **select * from example_table where (column1, column2) in ((1…n));**
forbid_select_for_update	Specifies whether the FOR UPDATE clause in the SELECT statement is involved in normalization calculation. If this option is enabled, **for update**, **for share**, **for update wait n**, **for update nowait**, **for update skip locks** are displayed in the normalized view and different values of **unique_sql_id** are generated. If this option is disabled, all the preceding clauses and statements without the for update clause are normalized into the same sentence and share the same **unique_sql_id**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Configuration Item	Behavior Control
temp_table	Specifies the calculation method of **unique_sql_id** for local temporary tables. If this option is disabled, the table OID is used as the table ID when the DML statement of the local temporary table calculates **unique_sql_id**. When the same operation is performed on the temporary tables with the same name created in different sessions, different **unique_sql_id** values are generated and displayed separately in the unique SQL-related views and functions. If the number of temporary tables created by service statements is controllable or the execution status of temporary table-related operations needs to be analyzed at the session level, you are advised to disable this option. If this option is enabled, the hash value of the table name string is used as the table ID when the DML statement of the local temporary table calculates **unique_sql_id**. In this case, tables with the same name but different table structures cannot be distinguished. When the same operation is performed on temporary tables with the same name created in different sessions, the same **unique_sql_id** is generated and can be displayed in the unique SQL-related views and functions. If there are a large number of same operations on local temporary tables with the same name in service statements, you are advised to enable this option to avoid generating too many records.

Default value: **"in_clause,forbid_select_for_update,temp_table"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is modified, the Unique SQL ID of the merged query statement will be changed, affecting the SQL statements for which SQL patch has been created.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- During the upgrade, the parameter value before the upgrade is retained by default. When a new instance is delivered, the default value is **"in_clause, forbid_select_for_update,temp_table"**.
- For values of the bigint, real, float4, blob, numeric, decimal, number, dec, or integer type, if **in()** contains a single parameter or multiple parameters, two different **unique_sql_id** values are generated.
- After the **forbid_select_for_update** option is enabled, the constant *n* in the **for update wait** *n* clause is not normalized. Different constants are recorded as different statements. (The **for update wait 0** and **for update nowait** clauses have the same semantics and are normalized to the same clause. The statement columns in the dbe_perf.statement, gs_asp, and statement_history views record the statement whose execution time is earlier.)
- After **instr_unique_sql_combination_options** is modified, **unique_sql_id** generated by the statements related to the modification is different from the original **unique_sql_id**. The original SQL patch may become invalid. You need to install the SQL patch again based on the new **unique_sql_id**.

instr_unique_sql_track_type

Parameter description: Specifies how unique SQL statements are recorded.

Parameter type: enumerated type.

Unit: none

Value range:

- **top**: Only top-level SQL statements are recorded.
- **all**: All SQL statements are recorded.

Default value: **all**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

- The default value of this parameter is **all**. Compared with **top**, the value **all** additionally records statistics about statements in the stored procedure, which causes performance loss for the execution of the stored procedure and more severe impact on the performance of statements that are executed quickly, such as simple expressions in the stored procedure.
- If this parameter is set to **top**, the OPEN CURSOR statement in the stored procedure is recorded.

unique_sql_retention_time

Parameter description: Specifies the interval for cleaning the unique SQL hash table. The default value is 30 minutes.

Parameter type: integer.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: minutes

Value range: 1 to 3650

Default value: **30**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: You are advised not to modify this parameter because this parameter has no function control point in the current version.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

enable_instr_rt_percentile

Parameter description: Specifies whether to enable the function of calculating the response time of 80% and 95% SQL statements in the system.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the function is enabled.
- **off** indicates that the function is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised to enable the parameter.

Risks and impacts of improper settings: If this parameter is disabled, 80% and 95% SQL response time cannot be recorded. As a result, system performance cannot be directly monitored.

percentile

Parameter description: Specifies the percentage of SQL statements whose response time is to be calculated by the backend calculation thread.

Parameter type: string.

Unit: none

Value range: **"80,95"**

Default value: **"80,95"**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

instr_rt_percentile_interval

Parameter description: Specifies the interval at which the background calculation thread calculates the SQL response time.

Parameter type: integer

Unit: seconds

Value range: 0 to 3600

Default value: **10**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **60** without a unit, **instr_rt_percentile_interval** indicates 60s. If the value is **1min**, **instr_rt_percentile_interval** indicates 1 minute. The unit must be **s**, **min**, or **h**, if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_instr_cpu_timer

Parameter description: Specifies whether to capture the CPU time consumed during SQL statement execution.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the CPU time consumed during SQL statement execution is captured.
- **off** indicates that the CPU time consumed during SQL statement execution is not captured.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised to enable the parameter.

Risks and impacts of improper settings: If this parameter is disabled, the CPU time consumed by SQL execution in the system may not be recorded, which affects slow SQL analysis.

perf_directory

Parameter description: Specifies the directory of the output file of the performance view dotting task. Only the SYSADMIN user can access this parameter. The value can be an absolute path, or relative to the data directory.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- Valid path: Users have read and write permissions on the path.
- Invalid path: Users do not have read or write permission on the path.

Parameter type: string.

Unit: none

Value range: valid path of a directory.

Default value: specified during installation.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If **perf_directory** in the configuration file is set to an invalid path, the cluster cannot be restarted.

enable_stmt_track

Parameter description: Specifies whether to enable Full/Slow SQL capture.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Full/Slow SQL capture is enabled.
- **off**: Full /Slow SQL capture is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

track_stmt_parameter

Parameter description: After **track_stmt_parameter** is enabled, the executed statements recorded in **statement_history** are not normalized. The complete SQL statement information can be displayed to help the database administrator locate faults. For a simple query, the complete statement information is displayed. For a PBE statement, the complete statement information and information about each variable value are displayed. The format is **"query string; parameters: *$1=value1,$2=value2,...*"** This parameter is used to display full SQL information. You can set the GUC parameter to control the memory, preventing the **query** parameter from occupying too much memory when it is a long string or a large object. The memory control of full SQL information is divided into two parts. The SQL statement part is controlled by the **instr_unique_sql_size** parameter, and the SQL parameter part is controlled by the **track_stmt_mem_ctl** parameter. After this

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

parameter is enabled, the P packets in statement_history may not record query information. Changing the value of this parameter does not affect the SQL statements that are being executed. When the SQL bypass logic is executed in PBE statements, the parameters are directly delivered to DNs. Therefore, you cannot obtain complete statement information by querying statement_history on CNs. In addition, because DNs do not have the query string information, you cannot obtain complete statement information by querying statement_history on DNs.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The function of displaying complete SQL statement information is enabled.
- **off**: The function of displaying complete SQL statement information is disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If most SQL statements are long, more memory and disk space are occupied.

track_stmt_session_slot

Parameter description: Specifies the maximum number of full/slow SQL statements that can be cached in a session. If the number of full/slow SQL statements exceeds this value, new statements will not be traced until the flush thread flushes the cached statements to the disk to reserve idle space.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **1000**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The memory of each occupied slot is about 800 bits. The specific value depends on the statement complexity. In scenarios with heavy service pressure, if the value is too large, the memory usage will increase rapidly. If the value is too small, some statements to be analyzed may be missed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

track_stmt_details_size

Parameter description: Specifies the maximum size of execution events that can be collected by a single statement. The value of this parameter affects the lock information of the **details** column in the **statement_history** table.

Parameter type: integer.

Unit: byte

Value range: 0 to 100000000

Default value: **4096**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

track_stmt_retention_time

Parameter description: Specifies the retention period of full SQL statements and slow SQL statements in the **statement_history** table and the retention period of files that have been flushed to disks on the standby node. This is a combination of parameters. This parameter is read every 60 seconds and records exceeding the retention period are deleted. Only the SYSADMIN user can access this parameter.

Parameter type: string.

Unit: none

Value range:

This parameter consists of two parts in the format of **'full sql retention time, slow sql retention time'**. The meanings of the two parts are as follows:

- **full sql retention time**: The retention period of full SQL statements. The value ranges from 0 to 86400. The unit is second.
- **slow sql retention time**: The retention period of slow SQL statements. The value ranges from 0 to 604800. The unit is second.

Default value: **"3600,604800"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the retention period is too long, a large amount of space may be occupied. If the retention period is too short, the data retention period may be too short. As a result, no fault locating method is available, affecting diagnosis.

track_stmt_stat_level

Parameter description: Determines the level of statement execution tracing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: string.

Unit: none

Value range:

This parameter consists of two parts in the format of **'full sql stat level, slow sql stat level'**. The meanings of the two parts are as follows:

- The first part indicates the tracing level of full SQL statements. The value can be **OFF**, **L0**, **L1**, or **L2**.
- The second part indicates the tracing level of slow SQL statements. The value can be **OFF**, **L0**, **L1**, or **L2**.

NOTE

If the tracing level of full SQL statements is not OFF, the current SQL tracing level is the higher level (L2 > L1 >L0) of full SQL statements and slow SQL statements. (The **query_plan** column is still subject to the original trace level.) For details about the levels, see **Table 7-153**.

Default value: **"OFF,L0"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the full SQL tracing function is enabled, the performance is affected and a large amount of disk space may be occupied.

track_stmt_standby_chain_size

Parameter description: Specifies the maximum memory and disk space occupied by fast/slow SQL statement records on the standby node. This is a combination of parameters. Only the SYSADMIN user can access the database.

Parameter type: string.

Unit: none

Value range:

This parameter consists of four parts in the format of **'Full SQL memory size, Full SQL disk size, Slow SQL memory size, Slow SQL disk size'**. The meanings of the four parts are as follows:

Full SQL and slow SQL statements are stored in different locations. Therefore, four values are used for control.

- **Full SQL memory size**: The maximum memory space reserved for fast SQL statements. The value range is [16,1024], in MB.
- **Full SQL disk size**: The maximum disk space occupied by reserved fast SQL statements. The value range is [512,1048576], in MB.
- **Slow SQL memory size**: The maximum memory space reserved for slow SQL statements. The value range is [16,1024], in MB.
- **Slow SQL disk size**: The maximum disk space reserved for slow SQL statements. The value range is [512,1048576], in MB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The memory size cannot be greater than the disk size.

Default value: **"32, 1024, 16, 512"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, a large amount of memory or disk space may be occupied. If the value is too small, some statements to be analyzed may be missed.

track_stmt_flush_mode

Parameter description: Specifies the storage mode of full SQL statements.

Parameter type: string.

Unit: none

Value range:

This parameter consists of two parts in the format of **'full sql flush mode, slow sql flush mode'**. The meanings of the two parts are as follows:

- The first part indicates the full SQL tracing mode. The value can be **MEMORY** or **FILE**. If this parameter is set to **MEMORY**, full SQL statements are recorded in the memory. If this parameter is set to **FILE**, full SQL statements are recorded in disk files.
- The second part indicates the slow SQL tracing mode. In the current version, the value can only be **FILE**. If this parameter is set to **FILE**, slow SQL statements are recorded in disk files.

Default value: **"FILE,FILE"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **"MEMORY,FILE"**, the kernel supports full SQL statements, which occupies certain shared memory. The memory size is specified by the **track_stmt_shm_size** parameter.

track_stmt_shm_size

Parameter description: Specifies the size of the full SQL shared memory.

Parameter type: integer

Unit: byte

Value range: 134217728 to 1073741824

Default value: **134217728**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

track_stmt_mem_ctl

Parameter description: Specifies the memory upper limit of each O&M sub-function.

Parameter type: string.

Unit: none

Value range: This parameter consists of three parts, that is, **'MESSAGE_MEM,GSTRACE_MEM,SQL_PARAM_MEM'**. The meanings are as follows:

- **MESSAGE_MEM**: When a statement is executed on the JDBC side, each time point is collected and sent to the database. This key is used to control the maximum number of bytes that can be received in the database. The received information is stored in the **driver_info** column in the **DBE_PERF.statement_history** table. The value ranges from 256 to 4096, in bytes.
- **GSTRACE_MEM**: The function dotting information is recorded in the memory. This key is used to control the memory size of the function dotting record of each SQL statement. The information is stored in the **details** column of the **DBE_PERF.statement_history** table. The value ranges from 0 to 104857600, in bytes.
- **SQL_PARAM_MEM**: The memory size of the SQL statement parameter part is controlled. That is, this parameter is valid only after **track_stmt_parameter** is enabled. For PBE statements, the complete statement information is displayed, and the value of each variable is added in the format of "query string;parameters:$1=value1,$2=value2..." In this case, the memory occupied by the parameters part is controlled. The value can be **0** or a number ranging from 100 to 102400, in bytes. The value **0** indicates that the length of the added parameters is not controlled and all parameter content is recorded.

Default value:
"MESSAGE_MEM=256,GSTRACE_MEM=2048,SQL_PARAM_MEM=1024"

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, too much memory may be occupied. If the value is too small, the number of statements recorded in the view may be too small. As a result, no fault locating method is available, affecting the diagnosis of slow SQL statements.

driver_traceid_level

Parameter description: Specifies the generation mode of the trace ID in the **DBE_PERF.statement_history** table.

Parameter type: string.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range: "none" and **"transaction"**

- **"none"**: The user-defined trace ID of the service is enabled to ensure compatibility with earlier versions.
- **"transaction"**: Statements in the same transaction use the same trace ID.

Default value: **"none"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Adjust the value based on service requirements.

Risks and impacts of improper settings: none

track_stmt_wait_driver_threshold

Parameter description: Collects statistics about statement execution time in the JDBC. When the statistics are transmitted to the database within the time range specified by this parameter, the statistics are processed in the database and flushed to disks. The statistics are recorded in the **driver_info** column of the **DBE_PERF.statement_history** table. If timeout occurs, the information is discarded.

Parameter type: integer.

Unit: milliseconds

Value range: 0 to 60000. The value **0** indicates that the function of collecting unique SQL information is disabled.

Default value: **5000**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **60000** without a unit, **track_stmt_wait_driver_threshold** indicates 60000 ms. If the value is **1min**, **track_stmt_wait_driver_threshold** indicates 1 minute. The unit must be **ms**, **s**, or **min** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a large value, SQL statements that fail to be executed occupy memory for a long time. As a result, statistics about newly executed SQL statements cannot be collected, and a large amount of memory is occupied.

enable_gstrace

Parameter description: Specifies whether to enable full-link GSTrace to trace kernel code execution.

Parameter type: Boolean.

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: enabled.
- **off**: disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: none

concurrent_dml_mode

Parameter description: Specifies how to handle concurrent DML execution conflicts (lock conflicts) in a table where the function of updating distribution keys is enabled (that is, the table attribute **enable_update_distkey** is set to **on**).

This parameter takes effect only on DNs.

Parameter type: integer

Unit: none

Value range: 0 to 1

- **0**: loose mode. That is, after a conflict occurs, the DML statement that obtains the lock later does not report an error.
- **1**: strict mode. If a conflict occurs, the DML statement that obtains the lock later terminates and reports an error.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the value range. If the DML statement that obtains the lock later does not report an error after a conflict is expected, set this parameter to **0**. If the DML statement that obtains the lock later terminates and reports an error after a conflict is expected, set this parameter to **1**.

Risks and impacts of improper settings: If this parameter is set to **0**, the DML statement that obtains the lock later may fail to obtain the data to be updated. Unexpected operations may occur during concurrent updates. For example, the later statement is successfully executed but does not take effect, or an extra data record is inserted using the MERGE INTO syntax. Therefore, change the parameter value after you fully understand the parameter meaning and test the parameter.

sql_hash_abort_list

Parameter description: Specifies the SQL exception blocking patch created based on the GUC parameter.

Parameter type: string.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: a single SQL hash value or multiple SQL hash values separated by commas (,).

Default value: ''

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: The performance may be affected, but the performance does not decrease by more than 5%.

NOTE

- Whether to block SQL statements is checked only on CNs and does not take effect on DNs.
- The valid range of SQL hash value is [1,4294967295].
- A maximum of 1,000 SQL hash values can be configured, and only the first 1,000 SQL hash values take effect.
- There is a low probability (1/4294967295) that hash conflicts occur in the exception blocking patch configured based on SQL hash values. As a result, irrelevant statements are blocked. You are advised to evaluate the risks.
- The configured SQL hash values do not distinguish databases and schemas. The same SQL statements in different databases and schemas are blocked.
- If multiple SQL execution paths contain the same internally executed SQL statement and the internally executed SQL statement is blocked, all external SQL statements that depend on the internally executed SQL statement are blocked. Exercise caution when performing this operation.

6.3.29 System Performance Snapshot

enable_perf_trigger

Parameter description: Specifies whether to enable the flame graph function for high seconds-level CPU usage.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The flame graph function is enabled for high seconds-level CPU usage.
- **off**: The flame graph function is disabled for high seconds-level CPU usage.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

perf_trigger_limit

Parameter description: Specifies the threshold for triggering flame graph collection due to high CPU usage. When the CPU usage exceeds the value of this parameter, a flame graph of high CPU usage is generated.

Parameter type: integer

Unit: %

Value range: 0 to 100

Default value: **90**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

perf_trigger_increase

Parameter description: Specifies the threshold for triggering flame graph collection due to high seconds-level CPU usage. When the increase rate of the system CPU usage compared with that in the previous second exceeds the value of this parameter, a flame graph of high seconds-level CPU usage is generated.

Parameter type: integer

Unit: %

Value range: 0 to 100

Default value: **30**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_wdr_snapshot

Parameter description: Specifies whether to enable the database monitoring snapshot function.

Parameter type: Boolean.

Unit: none

Value range:

- **on** indicates that the database monitoring snapshot function is enabled.
- **off** indicates that the database monitoring snapshot function is disabled.

Default value: **on**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised not to enable this function in the following scenarios:

- In multi-table and multi-database scenarios, WDR snapshots are recorded in each database in serial mode, which takes a long time. Therefore, you are advised not to enable it.
- In the multi-table scenario, the query speed of the pg_stat_all_tables is slow due to the performance problems of some internal tables. In this case, the generation speed of WDR snapshot function is slow. Therefore, you are advised not to enable it.
- If WDR snapshots are generated during a large number of DDL operations, the generation may fail. Therefore, you are advised not to enable it.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_wdr_snapshot_standby

Parameter description: Specifies whether the WDR function is supported on a distributed standby node. If this parameter is enabled, WDR snapshots obtain the performance data of the standby node and store the data on the primary node. This parameter takes effect only in GTM-free mode, that is, **enable_gtm_free** is set to **on**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The distributed standby node supports the WDR function.
- **off**: The distributed standby node does not support the WDR function.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If you want to adjust the value, make sure you understand the parameter meaning and adjust the value with caution to avoid risks caused by misoperations.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

To set **enable_wdr_snapshot_standby** to **on**, the following conditions must be met:

- The value of **wal_level** is **logical** or **hot_standby**.
- The value of **hot_standby** is **on**.
- In distributed standby node read mode, **gtm_option** is set to **2** or **enable_gtm_free** is set to **on**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_show_standby_name

Parameter description: Specifies whether to display the names of the primary and standby nodes in the same shard. The names are used to distinguish the primary and standby nodes. This parameter is used to distinguish primary and standby DNs by **node_name** in the performance view that is queried. If this parameter is enabled, different names are returned when pgxc_node_str() is queried on different nodes of the same shard.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The function of distinguishing primary and standby node names is enabled.
- **off**: The function of distinguishing primary and standby node names is disabled.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_wdr_cache

Parameter description: This parameter is used to adjust the status of the GUC parameter **enable_gsstat_cache** for WDR background threads. If **enable_wdr_cache** is set to **on**, **enable_gsstat_cache** is set to **on** for WDR backup threads. Otherwise, **enable_gsstat_cache** is set to **off**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Set **enable_gsstat_cache** to **on**.
- **off**: Set **enable_gsstat_cache** to **off**.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

wdr_snapshot_retention_days

Parameter description: Specifies the number of days database monitoring snapshots are retained. If the number of snapshots generated during database

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

running exceeds the maximum number of snapshots that can be generated within the retention period (24 x 8 = 192 by default), the system deletes the snapshots with the smallest ID at an interval specified by the value of **wdr_snapshot_interval**.

NOTE

This parameter is valid only when **enable_wdr_snapshot** is set to **on**.

Parameter type: integer

Unit: day

Value range: 1 to 30

Default value: **8**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: A larger value of this parameter indicates the higher disk usage.

wdr_snapshot_query_timeout

Parameter description: Specifies the execution timeout for the SQL statements associated with database monitoring snapshot operations. If the execution of a statement is not complete within the specified time and the result is returned, the snapshot operation fails, but some snapshots that have been generated are retained.

Parameter type: integer

Unit: seconds

Value range: 0 to 2147483647

Default value: **100**

Setting method: This parameter is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value **0** indicates that there is no timeout limit. For example, if the value is **60** without a unit, **wdr_snapshot_query_timeout** indicates 60s. If the value is **1min**, **wdr_snapshot_query_timeout** indicates 1 minute. The unit must be **s**, **min**, **h**, or **d** if required.

Setting suggestion: Set this parameter based on the actual scenario. If you do not care about the data in the snapshot and want to quickly generate a snapshot, set this parameter to a smaller value. If you want to ensure that the data in the snapshot is complete, set this parameter to a larger value.

Risks and impacts of improper settings: If snapshot generation times out, a larger value of this parameter results in a longer timeout period. As a result, the snapshot generation time is prolonged, but the data in the snapshot is more complete.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wdr_snapshot_interval

Parameter description: Specifies the interval at which the background thread snapshot automatically takes snapshots of the database monitoring data.

NOTE

This parameter is valid only when **enable_wdr_snapshot** is set to **on**.

Parameter type: integer

Unit: minute

Value range: 10 to 60

Default value: **60**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **60** without a unit, **wdr_snapshot_interval** indicates 60 minutes. If the value is **1h**, **wdr_snapshot_interval** indicates 1 hour. The unit must be **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: A smaller value of this parameter indicates the larger disk usage.

wdr_snapshot_space_threshold

Parameter description: Specifies the threshold at which the space used by snapshots is controlled. When the space used by snapshots reaches 50% of the value of this parameter, the control logic of the database is enabled to stabilize the space used by the snapshots.

NOTE

- This parameter is valid only when **enable_wdr_snapshot** is set to **on**.
- If this parameter is set to a value smaller than the space used by the snapshots, the space usage does not shrink. Instead, the control logic is enabled to control the snapshot growth and stabilize the space used by the snapshots.

Parameter type: integer

Unit: KB

Value range: 0 to 107374182400

Default value: **0**, indicating that the function of controlling the space used by snapshots based on the space threshold is disabled.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1024** without a unit, **wdr_snapshot_space_threshold** indicates 1024 KB. If the value is **1MB**, **wdr_snapshot_space_threshold** indicates 1 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Set this parameter based on the actual scenario.

Risks and impacts of improper settings: If this parameter is incorrectly set, only the latest group of snapshots is available.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wdr_snapshot_full_backup_interval

Parameter description: Specifies the interval at which a full WDR snapshot is created. The interval specified by this parameter is about a number instead of time. For example, if the parameter is set to **12**, a full snapshot and then 11 incremental snapshots are generated for each group. If the parameter is set to **1**, all snapshots generated are full snapshots.

NOTE

- This parameter is valid only when **enable_wdr_snapshot** is set to **on**.
- If you change the value of this parameter after multiple snapshots have been generated, the next snapshot will be a full snapshot. For example, after five WDR incremental snapshots have been generated, if the parameter value is changed to **10**, a full snapshot will be generated before a new interval starts.
- When a snapshot is deleted, other snapshots in the same group are deleted. The number of snapshots to be deleted depends on the value of this parameter when the earliest group of snapshots is generated and is irrelevant to the current value.

Parameter type: integer

Unit: none

Value range: 1 to 24

Default value: **12**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the actual scenario.

Risks and impacts of improper settings: A smaller value of this parameter indicates more full snapshots and fewer snapshots that can be retained with the same space usage.

enable_asp

Parameter description: Specifies whether to enable the active session profile (ASP) function.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The ASP function is enabled.
- **off**: The ASP function is disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_asp_standby

Parameter description: Specifies whether the standby node is enabled for ASP. After this function is enabled, the collected data of the standby node is obtained synchronously when the ASP is sampled.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The standby node function is enabled for ASP. ASP obtains the data collected from all standby nodes.
- **off**: The standby node function is disabled for ASP.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

asp_flush_mode

Parameter description: Specifies the mode for the ASP to update data to the disk. The value can be **"file"** (default value), **"table"** (system catalog), or **"all"** (system catalog and file). Only the SYSADMIN user can access this parameter.

Parameter type: string.

Unit: none

Value range: **"table"**, **"file"**, and **"all"**

Default value: **"table"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

asp_flush_rate

Parameter description: When the number of samples reaches the value of **asp_sample_num**, the samples in the memory are updated to the disk based on a certain proportion. **asp_flush_rate** indicates the update proportion. If this parameter is set to **10**, it indicates that the update ratio is 10:1.

Parameter type: integer

Unit: none

Value range: 1 to 10

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **10**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the amount of data flushed to disks increases.

asp_flush_interval

Parameter description: Specifies the interval at which ASP flushes data.

Parameter type: integer

Unit: seconds

Value range: 300 to 3600

Default value: **300**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **60** without a unit, **asp_flush_interval** indicates 60s. If the value is **1min**, **asp_flush_interval** indicates 1 minute. The unit must be **s**, **min**, or **h**, if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

asp_log_filename

Parameter description: Generates the file name format when writing files using ASP. Only the SYSADMIN user can access this parameter.

Parameter type: string.

Unit: none

Value range: valid character string in the file name format.

Default value: **"asp-***%Y-%m-%d_%H%M%S***.log"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

asp_retention_days

Parameter description: Specifies the maximum number of days for reserving ASP samples when they are written to the system catalog.

Parameter type: integer

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: day

Value range: 1 to 7

Default value: **2**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the disk space occupied by the gs_asp system catalog increases.

asp_sample_interval

Parameter description: Specifies the sampling interval.

Parameter type: integer

Unit: seconds

Value range: 1 to 10

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **10** without a unit, **asp_sample_interval** indicates 10s. If the value is **10s**, **asp_sample_interval** indicates 10s. The unit must be **s** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

asp_sample_num

Parameter description: Specifies the maximum number of samples allowed in the LOCAL_ACTIVE_SESSION view. Only the SYSADMIN user can access this parameter.

Parameter type: integer

Unit: none

Value range: 10000 to 100000

Default value: **100000**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

asp_log_directory

Parameter description: Specifies the directory for storing ASP log files on the server when **asp_flush_mode** is set to **all** or **file**. It can be an absolute path or a

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

relative path (relative to the data directory). Only the SYSADMIN user can access the path.

NOTE

- Valid path: Users have read and write permissions on the path.
- Invalid path: Users do not have read or write permission on the path.

Parameter type: string.

Unit: none

Value range: valid path of a directory.

Default value: specified during installation.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter after thorough tests and verification.

Risks and impacts of improper settings: If **asp_log_directory** in the configuration file is set to an invalid path, the cluster cannot be restarted.

gs_perf_interval

Parameter description: Specifies the interval at which the stacks are collected automatically.

Parameter type: floating-point number

Unit: minute

Value Range: 0 to 60. **0** indicates that the function of automatically collecting stack information is disabled.

NOTE

The valid value ranges from 0.2 to 60. If you attempt to set this parameter to a non-zero value less than 0.2, the value will be automatically adjusted to the default value **5**.

Default value: **5**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **60** without a unit, **gs_perf_interval** indicates 60 minutes. If the value is **1h**, **gs_perf_interval** indicates 1 hour. The unit must be **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the collection frequency is high, which may affect the system performance. In addition, a large number of logs are generated, occupying a large amount of storage space. Before modifying this parameter, make sure you understand its meaning and fully verify the change in a test environment.

gs_perf_deploy_mode

Parameter description: Specifies the deployment status of the **gs_perf_service** collection process.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: string.

Unit: none

Value range: **"ROOT"**, **"DBUSER"**, and **"DISABLE"**

- **"ROOT"**: The collection process is deployed in the **ROOT** state and supports all flame graph collection functions.
- **"DBUSER"**: The collection process is deployed in the **DBUSER** state and supports only on-CPU flame graph collection functions.
- **"DISABLE"**: The collection process is deployed in the **DISABLE** state and does not provide the flame graph function.

Default value: **"ROOT"**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

gs_perf_freq

Parameter description: Specifies the frequency of automatically collecting stack information.

Parameter type: integer

Unit: Hz

Value range: 10 to 1000

Default value: **999**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gs_perf_retention_days

Parameter description: Specifies the retention period of flame graph files.

Parameter type: integer

Unit: day

Value range: 1 to 8

Default value: **3**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.30 Black Box Parameters

blackbox_number

Parameter description: Specifies the maximum number of black box files that can be retained in the *$GAUSSLOG*/**gs_blackbox/***{nodename}* directory.

Parameter type: integer.

Unit: none

Value range: 0 to 10. **0** indicates that the black box function is disabled.

Default value: **5** (256-core CPU/2048 GB memory, 192-core CPU/1536 GB memory, 128-core CPU/1024 GB memory, 104-core CPU/1024 GB memory, 96-core CPU/1024 GB memory, 96-core CPU/768 GB memory, 80-core CPU/640 GB memory, 64-core CPU/512 GB memory, 60-core CPU/480 GB memory, 32-core CPU/256 GB memory, 16-core CPU/128 GB memory, 8-core CPU/64 GB memory, 4-core CPU/32 GB memory); **0** (4-core CPU/16 GB memory)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **0**, the black box function is disabled, and files in the *$GAUSSLOG*/**gs_blackbox/***{nodename}* directory are not managed or controlled. The gs_blackbox tool function is not affected.

blackbox_directory

Parameter description: Specifies the directory for storing black box temporary files.

Parameter type: string.

Unit: none

Value range: a string in the format of a valid directory. The file system type of the directory must be tmpfs.

Default value: **"/dev/shm"**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing. You can run the **df -TH** *[path]* command to query the file system type. The current user must have the read and write permissions on the newly specified directory. After the parameter is set and the node is restarted, you need to manually clear the files in the original directory.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3.31 Security Configuration

enable_security_policy

Parameter description: Specifies whether the unified audit and dynamic data masking policies take effect.

NOTE

- Unified audit: The unified audit mechanism is a technology that implements efficient security audit management by customizing audit policies. After the administrator defines the audit object and audit behaviors, if the task executed by a user is associated with an audit policy, the corresponding audit behavior is generated and the audit log is recorded.
- Dynamic data masking: The dynamic data masking mechanism is a technology that protects privacy data by customizing masking policies. It can effectively prevent unauthorized users from accessing sensitive information while retaining original data.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The security policy is enabled.
- **off**: The security policy is disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **on**, system resources will be occupied and system performance will be affected.

unified_audit_location

Parameter description: Specifies the location where unified audit logs are recorded.

Parameter type: integer.

Unit: none

Value range: **0** and **1**

- **0**: Unified audit logs are recorded in the location configured for rsyslog.
- **1**: Unified audit logs are recorded in the traditional audit log file.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. Before using unified audit logs to record audit logs, ensure that the GUC parameter **audit_enabled** is enabled.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If unified audit logs are required to record audit logs, set this parameter to **1**. This setting will occupy the resources used by the traditional audit function.

use_elastic_search

Parameter description: Specifies whether to send unified audit logs to Elasticsearch. If **enable_security_policy** and this parameter are enabled, unified audit logs are sent to Elasticsearch through HTTP or HTTPS (used by default).

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Unified audit logs are sent to Elasticsearch.
- **off**: Unified audit logs are not sent to Elasticsearch.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. This parameter is enabled when unified audit logs need to be sent to the Elasticsearch system.

Risks and impacts of improper settings: If this parameter enabled, you need to ensure that Elasticsearch specified by **elastic_search_ip_addr** can be properly connected. Otherwise, the setting of **use_elastic_search** does not take effect.

elastic_search_ip_addr

Parameter description: Specifies the IP address of the Elasticsearch system. If HTTPS is used, the format is **https://***ip*:*port*:*username*. If HTTP is used, the format is **http://***ip*:*port*. In the preceding command, *ip* indicates the IP address of the Elasticsearch server. *port* indicates the listening port for Elasticsearch HTTP communication, and the value ranges from 9200 to 9299. *username* indicates the username used for registering an Elasticsearch account. The initial user is **elastic**. If the HTTPS protocol is used, you need to configure related certificates.

Parameter type: string.

Unit: none

Value range: valid IP address string or empty string.

Default value: **""**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. This parameter is set when unified audit logs need to be sent to the Elasticsearch system and **use_elastic_search** is set to **on**.

Risks and impacts of improper settings: You need to ensure that the specified Elasticsearch can be properly connected. Otherwise, the setting of **use_elastic_search** does not take effect.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

is_sysadmin

Parameter description: Specifies whether the current user is an initial user.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The user is an initial user.
- **off**: The user is not an initial user.

Default value: **off**

Setting method: This is a fixed INTERNAL parameter. It can be viewed but cannot be modified.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

enable_tde

Parameter description: Specifies whether to enable the TDE function. Before creating an encrypted table, set this parameter to **on** and set **tde_key_info** for key information.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The TDE function is enabled.
- **off**: The TDE function is disabled.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If this parameter is set to **on**, ensure that the key information in the **tde_key_info** parameter is correctly configured and the key service can be accessed properly. Otherwise, the TDE function cannot be used.

Risks and impacts of improper settings: If encrypted data exists, exercise caution when disabling this function. After this function is disabled, encrypted table data cannot be read and encrypted logs cannot be replayed. As a result, the node may be faulty (**FATAL**/**PANIC**), the database may fail to be restarted, or the database may be unavailable.

tde_key_info

Parameter description: If TDE is enabled, transparent data needs to access the external key service to implement key management. This parameter is used to configure information about the key service, such as the service address, identity authentication information, and project information.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: string.

Unit: none

Value range: Contact technical support.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the key service in use.

Risks and impacts of improper settings: If encrypted data exists, exercise caution when changing the value of this parameter. Incorrect configuration means that the key is lost. As a result, encrypted table data cannot be read and encrypted logs cannot be replayed. As a result, the node may be faulty (**FATAL**/**PANIC**), the database may fail to be restarted, or the database may be unavailable.

tde_request_kms_count

Parameter description: Specifies the number of attempts to access the key management service after the TDE function is enabled.

Parameter type: integer.

Unit: none

Value range: 1 to 10

Default value: **3**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **1**, the key management service is requested only once, which cannot defend against intermittent network disconnection or jitter.

tde_index_default_encrypt

Parameter description: After this parameter is enabled, if you create an index whose base table is an encrypted table, the database automatically sets the index as an encrypted index, copies encryption parameters such as the algorithms and keys of the base table for the index, and encrypts the data of the index before storing it.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Encryption parameters are automatically set for indexes of encrypted tables.
- **off**: Encryption parameters are not automatically set for indexes in encrypted tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If indexes of all encrypted tables need to be encrypted, enable this parameter. You do not need to manually set indexes to encrypted indexes.

Risks and impacts of improper settings: If this parameter is set to **on**, indexes created in encrypted tables are automatically encrypted, affecting index query performance.

tde_encrypt_config

Parameter description: Specifies whether TDE supports Xlogs and undo logs.

Parameter type: string.

Unit: none

Value range: The value is in the format of **"key=value, key=value, ..."**, in which valid values of **key** and **value** are as follows.

key	value	Default Value	Function
log_encrypt	**on** or **off**	on	Specifies whether to encrypt Xlogs and undo logs.
log_algorithm	**aes_128_ctr** or **sm4_ctr**	aes_128_ctr	Specifies the algorithm for encrypting Xlogs and undo logs.
table_algorithm	**aes_128_ctr**, **sm4_ctr**, **aes_128_ctr_sha256_hmac**, **sm4_ctr_sm3_hmac**, or **aes_128_gcm**	aes_128_ctr	In the syntax CREATE .. SET (enable_tde=on, encrypt_algo=aes_128_ctr), if the **encrypt_algo** parameter is not set, the parameter set in **table_algorithm** is used by default.

Default value: **""**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value or adjust the value based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

block_encryption_mode

Parameter description: Specifies the block encryption mode used by the aes_encrypt and aes_decrypt functions for encryption and decryption.

Value type: enumerated type

Unit: none

Value range: **aes-128-cbc**, **aes-192-cbc**, **aes-256-cbc**, **aes-128-cfb1**, **aes-192-cfb1**, **aes-256-cfb1**, **aes-128-cfb8**, **aes-192-cfb8**, **aes-256-cfb8**, **aes-128-cfb128**, **aes-192-cfb128**, **aes-256-cfb128**, **aes-128-ofb**, **aes-192-ofb**, and **aes-256-ofb**. **aes** indicates the encryption or decryption algorithm. **128**, **192**, and **256** indicate the key length (unit: bit). **cbc**, **cfb1**, **cfb8**, **cfb128**, and **ofb** indicate the block encryption or decryption mode.

Default value: **aes-128-cbc**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value or adjust the value based on service requirements.

Risks and impacts of improper settings: The use of the aes_encrypt and aes_decrypt functions is affected.

enable_mac_check

Parameter description: Specifies whether label-based mandatory access control takes effect.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Label-based mandatory access control takes effect.
- **off**: Label-based mandatory access control does not take effect.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value or adjust the value based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

allow_initialuser_exec_others_udf

Parameter description: Specifies whether the initial user is allowed to execute functions or stored procedures defined by other users.

Parameter type: Boolean.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **on**: The initial user is allowed to execute functions or stored procedures defined by other users.
- **off**: The initial user is not allowed to execute functions or stored procedures defined by other users.

Default value:

- If a new version is directly installed, the default value is **off**.
- For a database upgraded from an earlier version, if the source version does not support this GUC parameter, the default value is **on**. If the source version supports this parameter, the current value of this parameter is retained after the upgrade.

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter to **off** to reduce the risk of unauthorized operations.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

allow_sysadmin_exec_others_udf

Parameter description: Specifies whether the SYSADMIN user is allowed to execute functions or stored procedures defined by other users.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The SYSADMIN user is allowed to execute functions or stored procedures defined by other users.
- **off**: The SYSADMIN user is not allowed to execute functions or stored procedures defined by other users.

Default value: **on**

Setting method: This is a SUSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter to **off** to reduce the risk of unauthorized operations. The SYSADMIN user can set this parameter to **on** in sessions as required.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_rls_match_index

Parameter description: Specifies whether indexes of a base table can be scanned based on target predicate conditions in row-level security scenarios. Target scenario: The row level security (RLS) policies are set and enabled in the base

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

table, and the query predicate contains the unleakproof system function or like operator.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Base table indexes can be scanned in the target scenario.
- **off**: Base table indexes cannot be scanned in the target scenario.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If hotspot query statements belong to the target scenario and there are few row-level access policies in the base table, enabling this function significantly improves query performance.

Risks and impacts of improper settings: If this parameter is set to **on**, the bitmap scanning operator is switched to the lossy mode, and there are a large number of RLS policies, filtering predicates will be executed repeatedly, affecting performance.

NOTE

- The modification of this parameter affects the generation of the execution plan in the target scenario. You can manually invalidate the cache plan by reconnecting to the system or creating operators.
- If this parameter is enabled, the generation of the bitmap scanning operator is affected, and the RLS policy predicate is inserted into the Recheck filter condition. Therefore, when the operator is switched to the lossy mode and there are many RLS policies, the performance is affected.

restrict_nonsystem_relation_kind

Parameter description: Specifies whether the relationship object types created by users can be accessed.

Parameter type: string.

Unit: none

Value range:

- **""**: There is no restriction.
- **"view"**: The access to views created by the user is disabled.
- **"foreign-table"**: The access to foreign tables created by the user is disabled.

Default value: **""**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

When gs_dump is used to export data, access to views and foreign tables created by users must be disabled to prevent malicious attacks. The tool sets this parameter to **'view, foreign-table'** at the session level. You do not need to manually configure this parameter.

enable_tde_async_encryption

Parameter description: Specifies whether to enable the asynchronous TDE function.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The asynchronous TDE function is enabled.
- **off**: The asynchronous TDE function is disabled.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If the service performance significantly deteriorates after the TDE function is enabled and all new data tables use the same asynchronous encryption data key, you are advised to enable this parameter to improve the performance.

Risks and impacts of improper settings: none

tde_async_dek

Parameter description: Stores the information about the asynchronous encryption key of transparent data, including the master key ID and data key ciphertext.

Parameter type: string.

Unit: none

Value range: a string.

Default value: **""**

Setting method: This is a SIGHUP parameter and cannot be set.

Setting suggestion: This parameter is automatically set by the database.

Risks and impacts of improper settings: none

tde_dkcache_remain_time

Parameter description: Specifies the validity period of the TDE key cache.

Parameter type: integer.

Unit: hour

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 0 to 1440

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **0** or a large value, the TDE key cache that is not used may remain in the database memory for a long time.

NOTE

- If this parameter is set to **0**, the TDE key cache takes effect permanently and is not cleared from the memory.
- If this parameter is set to a non-zero value, the TDE key cache is cleared from the memory only when it is not used for more than **tde_dkcache_remain_time** hours. (The cache is cleared every hour.)

enable_fips

Parameter description: Specifies whether to enable the FIPS mode. To enable the FIPS mode, set this parameter to **on** and ensure that the FIPS mode has been enabled in the operating environment.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The FIPS mode is enabled.
- **off**: The FIPS mode is disabled.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If this parameter is set to **on**, ensure that the existing data does not use the algorithms that are not supported by FIPS. Otherwise, unexpected exceptions may occur.

Risks and impacts of improper settings: After this parameter is set to **on**, the existing data that uses the algorithms that are not supported by FIPS cannot be used (including but not limited to data import, export, backup, and restoration). As a result, unexpected exceptions may occur. You are advised not to enable the FIPS mode in an environment with data. If you need to enable the FIPS mode, ensure that the existing data does not use the algorithms that are not supported by FIPS.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If this parameter is set to **on**, the GUC parameter **password_encryption_type** becomes invalid when it is set to **0**, **1**, or **3**. The SHA-256 algorithm is forcibly used to encrypt user passwords. Users whose historical passwords are encrypted using MD5 or SM3 cannot log in to the system.
- If this GUC parameter is set to **on**, errors can be intercepted and reported only for encryption and decryption functions, user creation, login, and TLS communication algorithms based on FIPS standards. To enter the FIPS mode, you need to configure OpenSSL to enter the FIPS mode. Otherwise, unexpected behavior may occur.

enable_thread_permission_down

Parameter description: Specifies whether to enable the background thread permission control. If this parameter is enabled, the background thread uses permissions of the gs_role_signal_backend role. If this parameter is disabled, the background thread uses the initial user permission.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The low permission mode is enabled.
- **off**: The low permission mode is disabled.

Default value:

- **on**: default value for a newly installed database.
- **off**: default value after an upgrade from a version earlier than 507.0.0 to the current version.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter to **on** to prevent background threads from exceeding their permissions.

Risks and impacts of improper settings: none

enableSeparationOfDuty

Parameter description: Specifies whether the separation of duties is enabled.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The separation of duties is enabled.
- **off**: The separation of duties is disabled.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to **on**, the permissions of a system administrator are restricted, and some operations that can be performed before may fail.

enable_nonsysadmin_execute_direct

Parameter description: Specifies whether non-SYSADMIN and non-MONADMIN users are allowed to execute the EXECUTE DIRECT ON statement.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Any user is allowed to execute the EXECUTE DIRECT ON statement.
- **off**: Only users with the SYSADMIN or MONADMIN permission are allowed to execute the EXECUTE DIRECT ON statement.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_access_server_directory

Parameter description: Specifies whether to allow non-initial users to create, modify, and delete directories.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Non-initial users are allowed to create, modify, and delete directories.
- **off**: Non-initial users are not allowed to create, modify, or delete directories.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- To use the advanced package UTL_FILE to access files on the server, you must have the permissions on the specified directory.
- For security purposes, only the initial user can create, modify, and delete directories by default.
- If **enable_access_server_directory** is enabled, users with the SYSADMIN permission and users who inherit the permissions of the built-in role gs_role_directory_create can create directories. A user with the SYSADMIN permission, the owner of a directory, a user who is granted the DROP permission on the directory, or a user who inherits the permissions of the built-in role gs_role_directory_drop can delete the directory. A user with the SYSADMIN permission and the owner of a directory object can change the owner of the directory, and the user must be a member of the new owner.

6.3.32 HyperLogLog

hll_default_log2m

Parameter description: Specifies the number of buckets for HLL data. The number of buckets affects the precision of distinct values calculated by HLL. More buckets indicate a smaller deviation. The deviation range is as follows: $[-1.04/2^{hll_default_log2m \times 1/2}, +1.04/2^{hll_default_log2m \times 1/2}]$

Parameter type: integer.

Unit: none

Value range: 10 to 16

Default value: **14**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value or adjust the value based on service requirements.

Risks and impacts of improper settings: If the value is too small, the required precision cannot be met. If the value is too large, resources may be wasted.

hll_default_log2explicit

Parameter description: Specifies the default threshold for switching from the explicit mode to the sparse mode.

Parameter type: integer.

Unit: none

Value range: 0 to 12

- **0**: The explicit mode is skipped.
- 1 to 12: The mode is switched when the base value reaches $2^{hll_default_log2explicit}$.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **10**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value or adjust the value based on service requirements.

Risks and impacts of improper settings: If the value is too small, the required precision cannot be met. If the value is too large, resources may be wasted.

hll_default_log2sparse

Parameter description: Specifies the default threshold for switching from the **sparse** mode to the **full** mode.

Parameter type: integer.

Unit: none

Value range: 0 to 14

- **0**: The sparse mode is skipped.
- 1 to 14: The mode is switched when the base value reaches $2^{hll_default_log2sparse}$.

Default value: **12**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value or adjust the value based on service requirements.

Risks and impacts of improper settings: If the value is too small, the required precision cannot be met. If the value is too large, resources may be wasted.

hll_duplicate_check

Parameter description: Specifies whether the duplicate check is enabled by default.

Parameter type: integer.

Unit: none

Value range: 0 to 1

- **0**: The function is disabled by default.
- **1**: The function is enabled by default.

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value or adjust the value based on service requirements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this parameter is enabled, more resources may be consumed for duplicate check, but the accuracy of distinct value calculation is improved.

6.3.33 User-defined Functions

udf_memory_limit

Parameter description: Specifies the maximum physical memory that can be used when CNs and DNs execute UDFs. This parameter does not take effect in the current version.

Parameter type: integer

Unit: KB

Value range: 200 x 1024 to 2147483647

Default value: **204800** (that is, 200 MB)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **204800** without a unit, it indicates 204800 KB. If this parameter is set to **300MB**, it indicates 300 MB. The unit must be KB, MB, or GB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

FencedUDFMemoryLimit

Parameter description: Specifies the virtual memory used by each fenced udf worker process.

Parameter type: integer

Unit: KB

Value range: 0 to 2147483647

Default value: **0**, indicating that the memory is not controlled.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. The value of this parameter can be specified without a unit, for example, **200** (indicating 200 KB), or with a unit, for example, **3MB**, (indicating 3 MB). The unit must be KB, MB, or GB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

UDFWorkerMemHardLimit

Parameter description: Specifies the maximum value of **FencedUDFMemoryLimit**.

Parameter type: integer

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: KB

Value range: 0 to 2147483647

Default value: **1048576**, that is, 1 GB.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. The value of this parameter can be specified without a unit, for example, **200** (indicating 200 KB), or with a unit, for example, **3MB**, (indicating 3 MB). The unit must be KB, MB, or GB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_udf_execute_direct

Parameter description: Specifies whether the EXECUTE DIRECT ON statement is allowed to execute user-defined functions, functions in the advanced packages DBE_SQL, PKG_SERVICE, and DBE_XMLGE, as well as DBE_UTILITY.EXEC_DDL_STATEMENT.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The EXECUTE DIRECT ON statement is allowed to execute user-defined functions, functions in the advanced packages DBE_SQL, PKG_SERVICE, and DBE_XMLGE, as well as DBE_UTILITY.EXEC_DDL_STATEMENT.
- **off**: The EXECUTE DIRECT ON statement is not allowed to execute user-defined functions, functions in the advanced packages DBE_SQL, PKG_SERVICE, and DBE_XMLGE, as well as DBE_UTILITY.EXEC_DDL_STATEMENT.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after you fully understand the parameter meaning and test the parameter to avoid unexpected results.

6.3.34 Scheduled Jobs

job_queue_processes

Parameter description: Specifies the number of jobs that can be concurrently executed.

Parameter type: integer.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 0 to 1000

- Setting **job_queue_processes** to **0** indicates that the scheduled job function is disabled and that no job will be executed. (Enabling scheduled jobs may affect the system performance. At sites where this function is not required, you are advised to disable it.)
- Setting **job_queue_processes** to a value that is greater than **0** indicates that the scheduled job function is enabled and this value is the maximum number of jobs that can be concurrently processed.

After the scheduled job function is enabled, the job_scheduler thread polls the **pg_job** system catalog at a scheduled interval. The scheduled job check is performed every second by default.

Too many concurrent jobs consume many system resources, so you need to set the number of concurrent jobs to be processed. If the current number of concurrent jobs reaches the value of **job_queue_processes** and some of them expire, these jobs will be postponed to the next polling period. Therefore, you are advised to set the polling interval (the **Interval** parameter of the submit API) based on the execution duration of each job to avoid the problem that jobs in the next polling period cannot be properly processed because of overlong job execution time.

Default value: **10**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If there are a large number of concurrent jobs at the same time, a small parameter value will cause job waiting, and a large parameter value will consume more system resources. Therefore, you are advised to set this parameter to **100**. You can adjust the parameter value based on system resource usage.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_prevent_job_task_startup

Parameter description: Specifies whether to start the job thread.

Parameter type: Boolean

Unit: none

Value range:

- **on** indicates that the job thread is not started.
- **off** indicates that the job thread is started.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3.35 Thread Pool

enable_thread_pool

Parameter description: Specifies whether to enable the thread pool function.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The thread pool function is enabled.
- **off**: The thread pool function is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: In low-concurrency persistent connection performance-sensitive scenarios, you are advised to disable this parameter. In other scenarios, you are advised to enable this parameter.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_connect_thread_pool

Parameter description: Specifies whether to enable the thread pool authentication. This thread pool authentication is optimization for the thread pool connection mode and depends on the thread pool connection function. To use this function, you need to enable the thread pool parameter **enable_thread_pool**. The thread pool authentication function is enabled only when both **enable_connect_thread_pool** and **enable_thread_pool** are set to **true**. If **enable_connect_thread_pool** is set to **true** and **enable_thread_pool** is set to **false**, the thread pool authentication function is disabled.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The thread pool authentication is enabled.
- **off**: The thread pool authentication is disabled.

Default value:

- **on**: default value of the newly installed database.
- **off**: default value of the database after upgrade from versions earlier than 505.2.1.SPC0800.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Enable this parameter in common scenarios.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: After this function is enabled, the jitter caused by high-concurrency connection setup on SQL services can be alleviated, but the connection access performance is reduced. If there are high requirements on the connection performance, you should properly configure the **connect_thread_pool_attr** parameter and the client timeout interval.

thread_pool_attr

Parameter description: Specifies the detailed attributes of the thread pool function. This parameter is valid only when **enable_thread_pool** is set to **on**. Only the SYSADMIN user can access this parameter.

Parameter type: string.

Unit: none

Value range:

This parameter is in the format of **'thread_num, group_num, cpubind_info, enable_thread_pool_auto_scaling'**. The meanings of each part are as follows:

- **thread_num** is the total number of initial threads in the thread pool, which can be dynamically expanded. The value ranges from 0 to 4096. The value **0** indicates that the database automatically configures the number of threads in the thread pool based on the number of CPU cores. If the value is greater than **0**, the number of threads in the thread pool is the same as the value of **thread_num**. You are advised to set the thread pool size based on the hardware configuration. The formula is as follows: Value of **thread_num** = Number of CPU cores x 3 to 5. The maximum value of **thread_num** is **4096**.
- **group_num** indicates the number of thread groups in the thread pool. The value ranges from 0 to 64. The value **0** indicates that the database automatically configures the number of thread groups in the thread pool based on the number of NUMA groups. If the value is greater than **0**, the number of thread groups in the thread pool is the same as the value of **group_num**.
- **cpubind_info** specifies whether the thread pool is bound to a core. The available configuration modes are as follows: 1. **'(nobind)'** indicates that no core is bound to threads. 2. **'(allbind)'** indicates that all queried CPU cores in the current system are bound to threads. 3. **'(nodebind: 1, 2)'** indicates that CPU cores in NUMA groups 1 and 2 are bound to threads. 4. **'(cpubind: 0-30)'** indicates that CPU cores 0 to 30 are bound to threads. 5. **'(numabind: 0-30)'**: CPU cores 0 to 30 in the NUMA group are bound to threads. When CPU core binding is specified, ensure that the number of NUMA groups is the same as the value of **group_num**. This parameter is case-insensitive.
- **enable_thread_pool_auto_scaling**: specifies whether to enable automatic scaling of the thread pool. This parameter is optional. The value **1** indicates that the function is enabled, and the value **0** indicates that the function is disabled. By default, this parameter is not set, indicating that the automatic scaling function of the thread pool is enabled.

Default value:

- Independent deployment: **'2048,2,(nobind)'** (128-core CPU/1024 GB memory); **'1024,2,(nobind)'** (60-core CPU/480 GB memory and 32-core CPU/256 GB memory); **'512,2,(nobind)'** (16-core CPU/128 GB memory);

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

'256,2,(nobind)' (8-core CPU/64 GB memory); **'128,2,(nobind)'** (4-core CPU/32 GB memory); **'64,2,(nobind)'** (4-core CPU/16 GB memory)

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: When the memory is sufficient and the CPU performance is good, increase the value of this parameter if the service requires more connections. Before changing the value, verify the change in the test environment.

Risks and impacts of improper settings: If the value of **thread_pool_attr** is greater than or equal to that of **max_connections**, all proc resources are occupied by the thread pool, causing proc resource insufficiency. You are advised to retain the default value. Before changing the values, check the specifications of the parameters and ensure that the hardware resources are sufficient. Otherwise, the cluster may be abnormal.

NOTE

Ensure that the sum of the number of threads specified by **thread_pool_attr** and the number of threads specified by **static_thread_pool_num** does not exceed min(4096, **max_connections**), and the value of **static_thread_pool_num** must be greater than the number of DN shards multiplied by the value of **group_num**. Otherwise, the static pool cannot be created.

connect_thread_pool_attr

Parameter description: Specifies the detailed attributes of the thread pool connection function. This parameter is valid only when **enable_thread_pool** and **enable_connect_thread_pool** are both set to **on**. Only the SYSADMIN user can access this parameter.

Parameter type: string.

Unit: none

Value range: **'auto'** or a character string in the format of '*thread_num, group_num, cpubind_info, enable_adjust_connect_pool*'.

- If the value is **'auto'**, the connection thread pool parameters are automatically calculated and set by the database based on the specifications of the deployed host. In the 4-core CPU scenario, the value is **'8,1,(cpubind:** *Two any bound valid CPUs***)'**. In other scenarios, the value is **'8,1,(nobind)'**.
- If the value is a character string consisting of configuration items, the format is '*thread_num, group_num, cpubind_info, enable_adjust_connect_pool*'. The configuration items are described as follows:
  - *thread_num*: total number of initial threads in the connection setup thread pool, which can be dynamically expanded. The value range is [1,Max(8,*Number of CPU cores*)] (that is, the lower limit is **1**, and the upper limit is the larger value between 8 and the number of CPU cores of the current deployment host). For example, if the total number of CPU cores of the current deployment host is 4, the upper limit of the number of threads in the connection setup thread pool is 4, which is less than 8. In this case, the upper limit is 8, and the value range of **thread_num** is [1,8]. If the total number of CPU cores of the current deployment host is 64, the upper limit of the number of threads in the connection setup

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

thread pool is 64, which is greater than 8. In this case, the upper limit is 64, and the value range of **thread_num** is [1,64]. The maximum number of threads is 500. If the value of **thread_num** is greater than the upper limit, scale-out is automatically disabled, and the number of threads in the connection thread pool does not change. If the value of **thread_num** is less than the upper limit, the value of the **enable_adjust_connect_pool** column is used to specify whether to enable scale-out.

- **group_num**: number of thread groups in the connection thread pool. The value is fixed to **1** and cannot be changed.
- **cpubind_info**: specifies whether the connection thread pool is bound to cores. The available configuration modes are as follows:
  - **'(nobind)'**: The connection thread pool is not bound to cores.
  - **'(allbind)'**: All CPUs that can be queried in the current system are used to bind threads to cores.
  - **'(nodebind: 1, 2)'**: CPUs in NUMA groups 1 and 2 are used to bind threads to cores.
  - **'(cpubind: 0-30)'**: CPUs 0 to 30 are used to bind threads to cores.
  - **'(numabind: 0-30)'**: CPUs 0 to 30 in the NUMA group are used to bind threads to cores. This parameter is case-insensitive.
- **enable_adjust_connect_pool**: specifies whether to disable the scale-out function of the connection thread pool. This parameter can be left empty. The default value is **0**, indicating that the scale-out function of the thread pool is enabled. If this parameter is set to **1**, the scale-out function is disabled. If the connection thread pool is overloaded, the emergency scale-out during the overload is not controlled by this parameter.

Default value: **'auto'**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: When the memory is sufficient and the CPU performance is good, increase the value of this parameter if the service requires more connections.

Risks and impacts of improper settings: Change the value after you fully understand the parameter meaning and test the parameter. If the value exceeds the value range, the GaussDB process cannot be started properly.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The value of **thread_num** in this parameter affects the system as follows: If the value of **thread_num** is small, the connection thread pool occupies fewer resources. In the scenario of a large number of concurrent connections, the connection performance deteriorates and the overload resistance capability of the system is improved. If the value of **thread_num** is large, the connection thread pool occupies more system resources. In the scenario of a large number of concurrent connections, the connection performance improves, but the overload resistance capability of the system is affected. Set this parameter within the allowed range based on service requirements.
- If the connection setup thread pool cannot allocate threads to process connection setup sessions for 2 seconds, the thread pool is overloaded. In the overload state, the system preferentially responds to the latest connection setup session and forcibly closes the connection setup session that has been waiting for more than 2 seconds.

static_thread_pool_num

Parameter description: Specifies the number of threads used to create a static thread pool (static pool). This parameter takes effect only on distributed CNs after **enable_thread_pool** is enabled.

Parameter type: integer.

Unit: none

Value range: 0 to 1024

Default value: 10% x **group_num** x **thread_num**. For details about **group_num** and **thread_num**, see **thread_pool_attr**.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- A static thread pool is created to prevent DNs from exhausting threads of the CN thread pool and ensure that available threads are connected to DNs to execute services.
- Ensure that the sum of the number of threads specified by **thread_pool_attr** and the number of threads specified by **static_thread_pool_num** does not exceed min(4096, **max_connections**), and the value of **static_thread_pool_num** must be greater than the number of DN shards multiplied by the value of **group_num**. Otherwise, the static pool cannot be created.
- **static_thread_pool_num** is a virtual value and does not take effect immediately after being set. When the CN thread pool is fully loaded or exhausted, the CN creates a number of threads specified by **static_thread_pool_num**. These threads make up a static pool.
- The static thread pool is included in the entire thread pool. After the static thread pool is configured, the maximum number of available worker threads decreases accordingly. For example, if the size of the static thread pool is set to **100** and the upper limit of the thread pool is set to **500**, the upper limit of the actual available normal threads is 400.
- The static thread pool occupies the number of connections. Therefore, the number of connections may not reach the value of **max_connections** due to the static thread pool. In this case, you can disable the static thread pool to avoid this problem.
- The CN static pool is used for single-DN connection services and evenly allocates threads to each DN shard in the cluster. The calculation formula is as follows: **static_thread_pool_num**/*n* (*n* indicates the number of DN shards). If the number of threads used exceeds the calculated fixed quota, an error is reported.
- If the static pool thread is held by the multi-DN connection service, an error is reported and the threads are returned.
- Non-default thread groups cannot be created in a static pool in multi-tenant mode.
- If **static_thread_pool_num** is set to **0**, the static pool function is disabled. If you do not want the static pool to control multiple DN connections, disable this function.
- Restrictions on creating a static thread pool: The prerequisite is that the thread pool is enabled on the distributed CN. The current function resolves DN shard faults and ensures that the CN has available threads for services to ensure continuous services. If the design does not include the solution for the fully occupied thread pool due to single-CN query, querying services only on a single CN at the same time are not controlled or counted in statistics.

thread_pool_stream_attr

Parameter description: Specifies the detailed attributes of the stream thread pool function. This parameter is valid only when **enable_thread_pool** is set to **on** and only takes effect on DNs. Only the SYSADMIN user can access this parameter.

Parameter type: string.

Unit: none

Value range:

This parameter consists of the four parts **'stream_thread_num'**, **'stream_proc_ratio'**, **'group_num'**, and **'cpubind_info'**, whose meanings are as follows:

- **stream_thread_num** indicates the total number of threads in the stream thread pool. The value ranges from 0 to 4096. The value **0** indicates that the database automatically configures the number of threads in the thread pool based on the number of CPU cores. If the value is greater than **0**, the number of threads in the thread pool is the same as the value of

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

stream_thread_num. You are advised to set the thread pool size based on the hardware configuration. The formula is as follows: Value of **stream_thread_num** = Number of CPU cores x 3 to 5. The maximum value of **stream_thread_num** is **4096**.

- **stream_proc_ratio** indicates the ratio of proc resources reserved for stream threads. The value is a floating-point number. The default value is **0.2**. The reserved proc resources are calculated as follows: Value of **stream_proc_ratio** x Value of **stream_thread_num**.
- **group_num** indicates the number of thread groups in the thread pool. The value ranges from 0 to 64. The value **0** indicates that the database automatically configures the number of thread groups in the thread pool based on the number of NUMA groups. If the value is greater than **0**, the number of thread groups in the thread pool is the same as the value of **group_num**. The value of **group_num** in **thread_pool_stream_attr** must be the same as that in **thread_pool_attr**. If they are set to different values, the value of **group_num** in **thread_pool_attr** is used.
- **cpubind_info** specifies whether the thread pool is bound to a core. The available configuration modes are as follows: 1. **'(nobind)'** indicates that no core is bound to threads. 2. **'(allbind)'** indicates that all queried CPU cores in the current system are bound to threads. 3. **'(nodebind: 1, 2)'** indicates that CPU cores in NUMA groups 1 and 2 are bound to threads. 4. **'(cpubind: 0-30)'** indicates that CPU cores 0 to 30 are bound to threads. 5. **'(numabind: 0-30)'** indicates that CPU cores 0 to 30 in the NUMA group are bound to threads. This parameter is case-insensitive. The value of **cpubind_info** in **thread_pool_stream_attr** must be the same as that in **thread_pool_attr**. If they are set to different values, the value of **cpubind_info** in **thread_pool_attr** is used.

Default value:

- **stream_thread_num**: **16**
- **stream_proc_ratio**: **0.2**
- **group_num** and **cpubind_info**: For details, see **thread_pool_attr**.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: When the memory is sufficient and the CPU performance is good, increase the value of this parameter if the service requires more concurrent performance.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_stream_threads

Parameter description: When the number of active stream threads (excluding those in the **sync quit** state) exceeds the value of **max_stream_threads**, the system waits until the number of active stream threads falls within the range specified by **max_stream_threads** before reusing or starting new stream threads.

When the number of stream threads exceeds the value of **max_stream_threads**, idle stream threads exit in batches by default until the number of stream threads

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

is less than or equal to the value of **max_stream_threads** or no idle stream thread exists.

This parameter takes effect only when **enable_thread_pool** is enabled.

Parameter type: integer.

Unit: none

Value range: 0 to 4096

Default value:

Independent deployment:

4096 (192-core CPU/1536 GB memory, 128-core CPU/1024 GB memory, 104-core CPU/1024 GB memory, and 96-core CPU/1024 GB memory); **2048** (96-core CPU/768 GB memory and 80-core CPU/640 GB memory); **1024** (64-core CPU/512 GB memory, 60-core CPU/480 GB memory, and 32-core CPU/256 GB memory); **128** (16-core CPU/128 GB memory); **64** (8-core CPU/64 GB memory); **32** (4-core CPU/32 GB memory); **16** (4-core CPU/16 GB memory)

Hybrid deployment

2048 (192-core CPU/1536 GB memory, 128-core CPU/1024 GB memory, 104-core CPU/1024 GB memory, 96-core CPU/1024 GB memory); **1024** (96-core CPU/768 GB memory, 80-core CPU/640 GB memory); **512** (80-core CPU/512 GB memory, 72-core CPU/576 GB memory, 64-core CPU/512 GB memory, 60-core CPU/480 GB memory, 32-core CPU/256 GB memory); **64** (16-core CPU/128 GB memory); **32** (8-core CPU/64 GB memory); **16** (4-core CPU/32 GB memory); **16** (4-core CPU/16 GB memory)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If the memory is sufficient and the CPU performance is good, increase the value of this parameter if the service requires higher parallel execution performance. Otherwise, decrease the value.

Risks and impacts of improper settings:

- If this parameter is set to a small value, the parallel execution plan cannot be generated, or the plan is queued while waiting for available stream threads, increasing the query delay.
- If this parameter is set to a large value and the number of CPU cores is small, the stream thread may occupy a large number of CPU resources, affecting the performance of other services.

NOTE

max_stream_threads is used to control the upper limit of stream threads on a single DN. If this parameter is set to **0**, the control is disabled.

stream_resource_control_attr

Parameter description: This parameter consists of three parts: **cpu_control_threshold**, **stop_stream_create_threshold**, and **os_run_queue_threshold**. They are used to control the CPU upper limit, the default error reporting threshold for new stream services, and the threshold for

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

the total number of running threads in the OS to exceed the number of CPU cores, respectively.

This parameter takes effect only when **enable_thread_pool** is enabled.

Parameter type: string.

Unit: none

Value range:

This parameter consists of three parts: **cpu_control_threshold**, **stop_stream_create_threshold**, and **os_run_queue_threshold**. The meanings of the three parts are as follows:

- **cpu_control_threshold**: threshold for controlling the CPU upper limit. The value ranges from 0 to 100.
- **stop_stream_create_threshold**: default error reporting threshold for new stream services. The value ranges from 0 to 100.
- **os_run_queue_threshold**: threshold for the total number of threads in the running state in the OS to exceed the multiple of the number of CPU cores. The value ranges from 0 to 100.

If the values of **cpu_control_threshold**, **stop_stream_create_threshold**, and **run_queue_threshold** are all **0**, this function is disabled.

When both **cpu_control_threshold** and **stop_stream_create_threshold** take effect, **cpu_control_threshold** must be smaller than or equal to **stop_stream_create_threshold**.

Default value:

- **cpu_control_threshold**: **0**
- **stop_stream_create_threshold**: **0**
- **os_run_queue_threshold**: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

When the CPU usage exceeds the value of **stop_stream_create_threshold**, the error message "Stream worker is temporarily unavailable. Try later..." is displayed when you attempt to start a new stream service.

thread_pool_idle_time

Parameter description: Specifies the idle time of a thread in the thread pool. When the idle time of a thread in the thread pool exceeds the value of this parameter, the thread exits and releases related memory resources. This parameter is valid only when **enable_thread_pool** is enabled. Only the SYSADMIN user can access this parameter. This parameter does not apply to threads in the authentication thread pool or static thread pool.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647

Default value: **300**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter affects the number of threads in the thread pool. You are advised not to set this parameter to a small value. If this parameter is set to a small value, threads in the thread pool may be started and stopped more frequently, affecting the performance.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

resilience_threadpool_reject_cond

Parameter description: Specifies the percentage of thread pool usage for escape from overload. This parameter takes effect only when the GUC parameters **enable_thread_pool** and **use_workload_manager** are enabled.

Parameter type: string.

Unit: none

Value range:

This parameter consists of the two parts **recover_threadpool_percent** and **overload_threadpool_percent**, whose meanings are as follows:

- **recover_threadpool_percent**: specifies the thread pool usage when the thread pool recovers to the normal state. When the thread pool usage is less than the value of this parameter, the escape from overload function is disabled and new connections are allowed. The value ranges from 0 to *INT_MAX*. The value indicates a percentage.
- **overload_threadpool_percent**: specifies thread pool usage when the thread pool is overloaded. If the thread pool usage is greater than the value of this parameter, the current thread pool is overloaded. In this case, the escape from overload function is enabled to kill sessions and forbid new connections. The value ranges from 0 to *INT_MAX*. The value indicates a percentage.

Example:

```
resilience_threadpool_reject_cond = '50,90'
```

When the thread pool usage exceeds 90%, new connections are forbidden and stacked sessions are killed. When the thread pool usage decreases to 50%, session killing is stopped and new connections are allowed. The kill policy is as follows: idle sessions are killed first and then sessions are killed in sequence based on the session status (sessions with the same status are killed in ascending order of start time).

Default value: **'0,0'**, indicating that the thread pool escape function is disabled.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

- The thread pool usage can be queried in the DBE_PERF.local_threadpool_status view. The initial number of threads in the thread pool can be obtained by querying the **thread_pool_attr** parameter.
- If this parameter is set to a small value, the thread pool escape from overload process is frequently triggered. As a result, ongoing sessions are forcibly logged out, and new connections fail to be connected for a short period of time. Therefore, exercise caution when setting this parameter based on the actual thread pool usage.
- The values of **recover_threadpool_percent** and **overload_threadpool_percent** can be **0** at the same time. In addition, the value of **recover_threadpool_percent** must be smaller than that of **overload_threadpool_percent**. Otherwise, the setting does not take effect.
- If the **use_workload_manager** parameter is disabled and the **bypass_workload_manager** parameter is enabled, this parameter also takes effect. The **bypass_workload_manager** parameter is of the SIGHUP type; therefore, after the reload mode is set, you need to restart the database for the setting to take effect.
- After the parameter is dynamically set to **"0,0"**, if session killing has been started, the killing action will not be terminated. You can wait for more than 10 seconds and then connect to the database to prevent new connections from being killed.

6.3.36 Full-Text Search

ngram_gram_size

Parameter description: Specifies the tokenization length of the N-gram parser.

Parameter type: integer.

Unit: none

Value range: 1 to 4

Default value: **2**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: none

NOTE

This parameter cannot affect the parsing behaviors of pg_catalog.ngram and other text search configurations that explicitly specify **gram_size**. You need to use ALTER TEXT SEARCH CONFIGURATION to modify options of created configurations.

ngram_grapsymbol_ignore

Parameter description: Specifies whether the N-gram parser ignores graphical characters.

Parameter type: Boolean.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range:

- **on**: Ignore graphical characters.
- **off**: Do not ignore graphical characters.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: none

ngram_punctuation_ignore

Parameter description: Specifies whether the N-gram parser ignores punctuations.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Ignore punctuations.
- **off**: Do not ignore punctuations.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: none

default_text_search_config

Parameter description: Specifies a full-text search configuration. The text search configuration is used by text search functions that do not have an explicit argument specifying the configuration.

Parameter type: string.

Unit: none

Value range: **"pg_catalog.english"**, **"pg_catalog.simple"**, **"pg_catalog.ngram"**, and **"pg_catalog.pound"**.

- **pg_catalog.english** and **pg_catalog.simple**: apply to English word tokenization scenarios.
- **pg_catalog.ngram**: applies to non-semantic Chinese word tokenization scenarios.
- **pg_catalog.pound**: applies to the non-semantic scenarios where the text to be parsed is separated by fixed separators.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

When a configuration file matching the environment is determined, gs_initdb will initialize the configuration file with a setting that corresponds to the environment.

Default value: **pg_catalog.english**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: none

Risks and impacts of improper settings: If the text search configuration does not exist, an error is reported. If the specified text search configuration is deleted, set **default_text_search_config** again. Otherwise, an error will be reported, indicating incorrect configuration.

tsearch_options

Parameter description: Specifies whether to enable full-text search.

Parameter type: string.

Unit: none

Value range: **configuration**, **dictionary**, and **debug_function**. If multiple options are configured, separate them with commas (,).

- **configuration**: allows full-text search for text configuration items, including CREATE/ALTER/DROP TEXT SEARCH CONFIGURATION syntax.
- **dictionary**: allows full-text search for dictionary syntax, including CREATE/ALTER/DROP TEXT SEARCH DICTIONARY.
- **debug_function**: allows full-text search for debugging functions, including the ts_debug, ts_parse, ts_token_type, ts_lexize, and ts_stat functions.

Default value:

- The default value for a newly installed cluster is **""**.
- After the database upgrade in a version earlier than 506.0.0, the default value is **"configuration,dictionary,debug_function"**.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: none

6.3.37 Backup and Restoration

operation_mode

Parameter description: Specifies whether the system enters the backup and restoration mode.

Parameter type: Boolean.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **on**: The backup and restoration mode is entered.
- **off**: The backup and restoration mode is not entered.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

enable_cbm_tracking

Parameter description: Specifies whether to enable the CBM tracing function.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The cbm tracking is enabled.
- **off**: The cbm tracking is disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Enable this parameter to perform full or incremental backup for the cluster by using Roach. Otherwise, the backup will fail.

Risks and impacts of improper settings: If this parameter is disabled, incremental backup will fail.

max_size_for_xlog_retention

Parameter description: Specifies when to forcibly update the backup replication slot or logical replication slot to prevent the disk from being full and the cluster from being read-only because logs cannot be recycled during backup or logical decoding.

Parameter type: integer.

Unit: none

Value range: –100 to 2147483647

- The value **0** indicates that this function is disabled.
- A negative value indicates that the backup replication slot or logic replication slot is forcibly updated when the disk usage exceeds the threshold and logs are recycled due to blocked backup operation or logical decoding. For example, **-80** indicates that the disk usage exceeds 80% of the threshold.
- A positive value indicates that the backup replication slot or logic replication slot is forcibly updated when the size of stacked logs exceeds the threshold

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

and logs are recycled due to blocked backup operation or logical decoding. For example, **32** indicates that the backup replication slot lags behind the redo location of the current checkpoint by more than 32 log segments. (The size of each log segment is 16 MB.)

Default value: **-80**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter to a value slightly smaller than that of **datastorage_threshold_value_check** of the CM Server component to prevent the cluster from entering the read-only state.

Risks and impacts of improper settings: If the value is too small, backup may fail frequently.

max_cbm_retention_time

Parameter description: Specifies the interval at which CBM backup files are forcibly recycled. If CBM files cannot be recycled during backup, the disk will be full and the cluster will be read-only.

Parameter type: integer.

Unit: second

Value range: 86400 to 2147483647

Default value: **1209600** (that is, 14 days)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **86400** without a unit, it indicates 86400s. If this parameter is set to **1d**, it indicates 1 day. The unit must be **s**, **min**, **h**, or **d** if required.

Setting suggestion: Set this parameter based on the full backup interval.

Risks and impacts of improper settings: If the value is too small, backup may fail frequently.

enable_standby_move_buckets

Parameter description: Specifies whether a CN is allowed to connect to a standby node to move buckets. This parameter is used only for cluster-level restoration. When restoring the backup set generated during hash bucket redistribution, this parameter is used to establish a connection with the standby DN when the CN sends the buckets migration command to the DN. This parameter takes effect only in this connection. In other cases, the value is **off**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The standby node can receive connections.
- **off**: The standby node cannot receive connections.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **off**

Setting method: This is a BACKEND parameter and cannot be set or reloaded using gs_guc. This parameter does not take effect when written into the configuration file. It is used only as an O&M parameter for data restoration.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

hadr_max_size_for_xlog_receiver

Parameter description: Specifies the synchronization progress difference threshold between archive log download and log replay. When the progress difference exceeds the threshold, the system automatically suspends archive log download to avoid log accumulation and resource overload.

Parameter type: integer.

Unit: KB

Value range: 0 to 2147483647

Default value: **268435456**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the download of archive logs may be frequently suspended.

6.3.38 Undo

The GUC parameter of undo is valid only for the Ustore.

undo_space_limit_size

Parameter description: Specifies the threshold for forcibly recycling undo space. When the undo space usage reaches 80% of the threshold, forcible recycling starts. You are advised to set **undo_space_limit_size** to a value greater than or equal to that of **undo_limit_size_per_transaction**.

Parameter type: integer.

Unit: page (8 KB)

Value range: 102400 to 2147483647. The default unit is page.

Default value: **33554432** (that is, 256 GB)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, the value **1000000** indicates 1,000,000 pages, and the value **1000000kB** indicates 1,000,000 KB. If the value contains a unit, the value can be KB, MB, or GB, but cannot be TB.

Setting suggestion: Retain the default value. If the disk space is small and cannot meet the requirement of the default value, you can decrease the value of this

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

parameter as required. If long transactions or large transactions exist in the service, you need to increase the value of **undo_space_limit_size** based on the disk space. For details about the adjustment value, refer to the **info** column of the gs_stat_undo system function (see **Table 1-149**).

Risks and impacts of improper settings: If this parameter is set to a small value when undo records cannot be reclaimed in a timely manner, undo records may be forcibly reclaimed. As a result, there is an extremely low probability that the error message "snapshot too old! the undo record has been forcibly discard" is displayed. If this parameter is set to a large value when the disk space is insufficient and undo records cannot be reclaimed in a timely manner, a large number of undo records may be retained in the database system. As a result, the undo records occupy the physical disk space with other data, and services may be affected when the disk space is insufficient.

undo_limit_size_per_transaction

Parameter description: Specifies the undo space threshold of a single transaction. If the threshold is reached, the transaction is rolled back due to an error. You are advised to set **undo_limit_size_per_transaction** to a value less than or equal to that of **undo_space_limit_size**. If the value of **undo_limit_size_per_transaction** is greater than that of **undo_space_limit_size**, the displayed value is the same as the configured value when you run the **show undo_limit_size_per_transaction** command to query the parameter value. The only difference is that the smaller value between **undo_space_limit_size** and **undo_limit_size_per_transaction** is used as the actual undo space threshold of a single transaction.

Parameter type: integer.

Unit: page (8 KB)

Value range: 256 to 2147483647. The default unit is page. If this parameter is set to a value greater than 134217728 (that is, 1 TB), the value **134217728** takes effect.

Default value: **4194304** (32 GB)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, the value **10000** indicates 10000 pages, and the value **10000KB** indicates 10000 KB. If the value contains a unit, the value can be KB, MB, or GB, but cannot be TB.

Setting suggestion: Retain the default value. If the disk space or memory space is small and cannot meet the requirement of the default value, you can decrease the value of this parameter as required. If a long transaction or a large transaction exists in the service and the number of undo records generated in a single transaction is greater than the value of this parameter, you need to increase the value of **undo_limit_size_per_transaction** based on the disk space. For details about the adjustment value, refer to the **info** column in the gs_stat_undo system function (**Table 1-149**). If **undo_limit_size_per_transaction** is set to a value greater than 1 TB, the system performance and stability may be affected. Therefore, if the parameter is set to a value greater than 134217728 (that is, 1 TB), the configured value is displayed when you run the **show undo_limit_size_per_transaction** command to query the parameter value, but the value **134217728** takes effect.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If the parameter is set to small value, the number of undo records generated in a single transaction may exceed the threshold. As a result, the error message "undo size of the transaction exceeds the threshold" is displayed and the transaction is rolled back.

6.3.39 DCF Parameters Settings

enable_dcf

Parameter description: Specifies whether to enable the DCF mode. You are advised not to manually modify this parameter.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The current log replication mode is DCF.
- **off**: The current log replication mode is not DCF.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: To use the DCF log replication mode, run the mode switching command. Do not manually modify the **enable_dcf** parameter. This parameter takes effect only on DNs and does not take effect on CNs.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

dcf_ssl

Parameter description: This parameter is no longer used. The DCF reuses the GUC parameter **ssl**. For details, see **Security and Authentication**.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: SSL is used.
- **off**: SSL is not used.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

dcf_config

Parameter description: Specifies the DCF cluster configuration information, which is configured by the OM during installation and cannot be modified after installation.

Parameter type: string.

Unit: none

Value range: a string.

Default value: configured by the OM during installation.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

dcf_data_path

Parameter description: Specifies the DCF data path, which is configured by the OM during installation and cannot be modified after installation.

Parameter type: string.

Unit: none

Value range: valid path of a directory.

Default value: **dcf_data** in the DN data directory

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

dcf_log_path

Parameter description: Specifies the DCF log path, which is configured by the OM during installation and cannot be modified after installation.

Parameter type: string.

Unit: none

Value range: valid path of a directory.

Default value: **dcf_log** in the DN data directory

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter can only be viewed.

Risks and impacts of improper settings: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

dcf_node_id

Parameter description: Specifies the ID of the DN where the DCF is located. This parameter is defined by the user during installation and mode switching.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483647

Default value: specified by users during installation.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

dcf_max_workers

Parameter description: Specifies the largest number of DCF callback function threads. The DCF needs to apply for the shared memory and semaphore before it calls the function registered by the DN with the DCF through the callback function.

Parameter type: integer.

Unit: none

Value range: 0 to 262143

Default value: **40**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a large value, resources may be insufficient and the cluster installation may fail.

dcf_truncate_threshold

Parameter description: Specifies the threshold for a DN to truncate DCF logs.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483647

Default value: **100000**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If the threshold is too large, logs may not be truncated for a long time, occupying memory.

dcf_election_timeout

Parameter description: Specifies the timeout interval for selecting the DCF leader and follower. The election timeout interval depends on the status of the network between DNs. If the timeout interval is short and the network quality is poor, timeout occurs. After the network recovers, the election becomes normal. You are advised to set a proper timeout interval based on the current network status. Restriction on the DCF node clock: The maximum clock difference between DCF nodes is less than half of the election timeout interval. In DCF manual election mode, to ensure timely CM election, do not modify this parameter. Instead, use the default election timeout interval.

Parameter type: integer.

Unit: second

Value range: 1 to 600

Default value: **3**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **60** without a unit, **dcf_election_timeout** indicates 60s. If the value is **1min**, **dcf_election_timeout** indicates 1 minute. The unit must be second or minute if required.

Setting suggestion: Set a proper timeout interval based on the current network status.

Risks and impacts of improper settings: If the network environment is poor and the timeout period is short, timeout election may occur.

dcf_enable_auto_election_priority

Parameter description: Specifies whether the DCF priority is allowed to be automatically adjusted.

Parameter type: integer.

Unit: none

Value range:

- **0**: not allowed.
- **1**: allowed.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

dcf_election_switch_threshold

Parameter description: Specifies the DCF threshold for preventing frequent switchover to primary. It is recommended that this parameter be set based on the maximum fault duration acceptable for user services.

Parameter type: integer.

Unit: second

Value range: 0 to 2147483647

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Set this parameter based on the maximum fault duration acceptable to services.

Risks and impacts of improper settings: If this parameter is set to a large value, services may be unavailable for a long time.

dcf_run_mode

Parameter description: Specifies the DCF election mode, including the automatic election mode, manual election mode, and disabling the election mode. Currently, the election mode can be disabled only in minority restoration scenarios. If the election mode is disabled, the database instance will become unavailable.

NOTE

The working mode of a cluster can be switched only when the cluster is running properly. Otherwise, the cluster is still abnormal after the switching. The DCF working mode configured in GUC parameters must be the same as that configured by using cm_ctl. That is, both DCF working modes must be set to manual or automatic at the same time.

For example, to set the DCF manual mode, run the following command:

```
cm_ctl set --param --server -k dn_arbitrate_mode=quorum
cm_ctl reload --param --server
gs_guc reload -Z datanode -I all -N all  -c "dcf_run_mode=1"
```

To set the DCF automatic mode, run the following command:

```
cm_ctl set --param --server -k dn_arbitrate_mode=paxos
cm_ctl reload --param --server
gs_guc reload -Z datanode -I all -N all  -c "dcf_run_mode=0"
```

Parameter type: enumerated type

Unit: none

Value range: **0**, **1**, and **2**

- **0**: automatic election mode.
- **1**: manual election mode.
- **2**: disabling the election mode.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Currently, disabling the election mode applies only to minority recovery scenarios. If the election mode is disabled, the database instance will be unavailable.

dcf_log_level

Parameter description: Specifies the DCF log level.

Parameter type: string.

Unit: none

Value range:

- To disable the log function, set this parameter to **"NONE"**, which cannot be used together with the following log levels:
- To enable the log function, set this parameter to one or a combination of the following log levels: **"RUN_ERR|RUN_WAR|RUN_INF|DEBUG_ERR|DEBUG_WAR|DEBUG_INF|TRACE|PROFILE|OPER"**.

  If two or more log levels are used together, separate them with vertical bars (|). The log level cannot be set to an empty character string.

Default value: **"RUN_ERR|RUN_WAR|DEBUG_ERR|OPER|RUN_INF|PROFILE"**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the log level is too low, logs may be missing.

dcf_log_backup_file_count

Parameter description: Specifies the number of DCF run log backups.

Parameter type: integer.

Unit: none

Value range: 1 to 1000

Default value: **100**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the number of backup files is small and logs are lost.

dcf_max_log_file_size

Parameter description: Specifies the maximum size of a DCF run log file.

Parameter type: integer.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: MB

Value range: 1 to 1000.

Default value: **10**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **10** without a unit, **dcf_max_log_file_size** indicates 10 MB. If the value is **10MB**, **dcf_max_log_file_size** indicates 10 MB. The unit must be MB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, a single log file may be opened and the query efficiency may decrease.

dcf_socket_timeout

Parameter description: Specifies the timeout interval for the DCF communication module to connect to the socket. This parameter takes effect upon the system restart. In an environment where the network quality is poor, if the timeout interval is set to a small value, a connection may fail to be set up. In this case, you need to increase the value.

Parameter type: integer.

Unit: ms

Value range: 10 to 600000 (10 minutes)

Default value: **5000**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **5000** without a unit, **dcf_socket_timeout** indicates 5000 ms. If the value is **1min**, **dcf_socket_timeout** indicates 1 minute (that is, 60000 ms). The unit must be millisecond, second, or minute if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: When the network environment is poor, a small timeout period may cause a connection setup failure.

dcf_connect_timeout

Parameter description: Specifies the timeout interval for the DCF communication module to set up a connection. This parameter takes effect upon the system restart. In an environment where the network quality is poor, if the timeout interval is set to a small value, a connection may fail to be set up. In this case, you need to increase the value.

Parameter type: integer.

Unit: ms

Value range: 10 to 600000 (10 minutes)

Default value: 60000

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **5000** without a unit, **dcf_connect_timeout** indicates 5000 ms. If the value is **1min**, **dcf_connect_timeout** indicates 1 minute (that is, 60000 ms). The unit must be millisecond, second, or minute if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: When the network environment is poor, a small timeout period may cause a connection setup failure.

dcf_mec_fragment_size

Parameter description: Specifies the fragment size of the DCF communication module. This parameter takes effect upon the system restart.

Parameter type: integer.

Unit: KB

Value range: 32 to 10240

Default value: **64**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

dcf_stg_pool_max_size

Parameter description: Specifies the maximum size of the memory pool of the DCF storage module. This parameter takes effect upon the system restart.

Parameter type: integer.

Unit: MB

Value range: 32 to 2147483647

Default value: **2048**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **2048** without a unit, **dcf_stg_pool_max_size** indicates 2048 MB. If the value is **5GB**, **dcf_stg_pool_max_size** indicates 5 GB (that is, 5120 MB). The unit must be MB or GB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, memory resources may be insufficient. If the value is too small, storage module resources may be insufficient.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

dcf_stg_pool_init_size

Parameter description: Specifies the minimum size of the memory pool of the DCF storage module. This parameter takes effect upon the system restart.

Parameter type: integer.

Unit: MB

Value range: 32 to 2147483647

Default value: **32**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **2048** without a unit, **dcf_stg_pool_init_size** indicates 2048 MB. If the value is **5GB**, **dcf_stg_pool_init_size** indicates 5 GB (that is, 5120 MB). The unit must be MB or GB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, memory resources may be insufficient. If the value is too small, storage module resources may be insufficient.

dcf_mec_pool_max_size

Parameter description: Specifies the maximum size of the memory pool of the DCF communication module. This parameter takes effect upon the system restart.

Parameter type: integer.

Unit: MB

Value range: 32 to 2147483647

Default value: **200**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **2048** without a unit, **dcf_mec_pool_max_size** indicates 2048 MB. If the value is **5GB**, **dcf_mec_pool_max_size** indicates 5 GB (that is, 5120 MB). The unit must be MB or GB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, memory resources may be insufficient. If the value is too small, communications module resources may be insufficient.

dcf_flow_control_disk_rawait_threshold

Parameter description: Specifies the disk waiting threshold for DCF flow control.

Parameter type: integer.

Unit: μs

Value range: 0 to 2147483647

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **100000**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a large value, the performance may deteriorate and the memory usage may increase.

dcf_flow_control_net_queue_message_num_threshold

Parameter description: Specifies the threshold for the number of messages in a network queue for DCF flow control.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **1024**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a small value, the performance may deteriorate severely.

dcf_flow_control_cpu_threshold

Parameter description: Specifies the threshold for DCF CPU flow control.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **100**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a small value, the performance may deteriorate severely.

dcf_mec_batch_size

Parameter description: Specifies the number of batch messages for DCF communication. When the value is **0**, the DCF automatically adjusts the value based on the network and the amount of data to be written. This parameter takes effect upon the system restart.

Parameter type: integer.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range: 0 to 1024

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a small value, the communications efficiency between nodes decreases.

dcf_mem_pool_max_size

Parameter description: Specifies the maximum DCF memory. This parameter takes effect upon the system restart.

Parameter type: integer.

Unit: MB

Value range: 32 to 2147483647

Default value: **2048**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **2048** without a unit, **dcf_mem_pool_max_size** indicates 2048 MB. If the value is **5GB**, **dcf_mem_pool_max_size** indicates 5 GB (that is, 5120 MB). The unit must be MB or GB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a small value, the cluster installation will fail.

dcf_mem_pool_init_size

Parameter description: Specifies the initial size of the DCF memory. This parameter takes effect upon the system restart.

Parameter type: integer.

Unit: MB

Value range: 32 to 2147483647

Default value: **32**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **2048** without a unit, **dcf_mem_pool_init_size** indicates 2048 MB. If the value is **5GB**, **dcf_mem_pool_init_size** indicates 5 GB (that is, 5120 MB). The unit must be MB or GB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a small value, the cluster installation will fail.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

dcf_compress_algorithm

Parameter description: Specifies the compression algorithm for DCF run log transmission.

Parameter type: integer.

Unit: none

Value range:

- **0**: no compression.
- **1**: LZ4 compression algorithm.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: To enable the LZ4 compression algorithm, set **dcf_compress_algorithm** to **1**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

dcf_compress_level

Parameter description: Specifies the compression level for DCF log transmission. Before this parameter takes effect, a valid compression algorithm must be configured, that is, the **dcf_compress_algorithm** parameter is set.

Parameter type: integer.

Unit: none

Value range: 1 to 22. If compression is disabled, the configured compression level does not take effect.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

dcf_mec_channel_num

Parameter description: Specifies the number of DCF communication channels. This parameter takes effect upon the system restart.

Parameter type: integer.

Unit: none

Value range: 1 to 64

Default value: **1**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a small value, the communications efficiency between nodes may decrease.

dcf_rep_append_thread_num

Parameter description: Specifies the number of DCF log replication threads. This parameter takes effect upon the system restart.

Parameter type: integer.

Unit: none

Value range: 1 to 1000

Default value: number of DN copies in each shard.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a large value, threads may be idle and the resource usage may decrease.

dcf_mec_agent_thread_num

Parameter description: Specifies the number of DCF communication working threads. This parameter takes effect upon the system restart.

Parameter type: integer.

Unit: none

Value range: 1 to 1000

Default value: **10**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: It is recommended that the value be greater than or equal to 2 x Number of nodes x **dcf_mec_channel_num**.

Risks and impacts of improper settings: If this parameter is set to a large value, threads may be idle and the resource usage may decrease.

dcf_mec_reactor_thread_num

Parameter description: Specifies the number of reactor threads used by the DCF. This parameter takes effect upon the system restart.

Parameter type: integer.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 1 to 100

Default value: **1**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: It is recommended that the ratio of the value of **dcf_mec_reactor_thread_num** to the value of **dcf_mec_agent_thread_num** be 1:40.

Risks and impacts of improper settings: If this parameter is set to a large value, threads may be idle and the resource usage may decrease.

dcf_log_file_permission

Parameter description: Specifies the attribute of the DCF run log file. The parameter setting takes effect after the system is restarted. This parameter is configured during installation and cannot be modified. To allow other users in the same group to access logs, ensure that all parent directories can be accessed by other users in the same group. That is, if **dcf_log_path_permission** is set to **750**, **dcf_log_file_permission** can only be set to **600** or **640**. If **dcf_log_path_permission** is set to **700**, **dcf_log_file_permission** must be set to **600**.

Parameter type: enumerated type.

Unit: none

Value range: **600** and **640**

Default value: **600**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: The permission of **dcf_log_path_permission** must be higher than that of **dcf_log_file_permission**.

Risks and impacts of improper settings: Improper settings may cause user access failures.

dcf_log_path_permission

Parameter description: Specifies the attribute of the DCF run log directory. The parameter setting takes effect after the system is restarted. This parameter is configured during installation and cannot be modified. To allow other users in the same group to access the log path, set this parameter to **750**. Otherwise, set this parameter to **700**.

Parameter type: enumerated type.

Unit: none

Value range: **700** and **750**

Default value: **700**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: To allow other users in the same group to access the log path, set this parameter to **750**. Otherwise, set this parameter to **700**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

dcf_majority_groups

Parameter description: Sets the DCF policy-based majority function. For a group that requires this parameter, at least one standby node in the group receives logs. That is, there is a synchronous standby node in the group. If nodes are added to or deleted from the DCF instance or the group value of a node in the instance is changed, you need to modify the configuration accordingly. When modifying this parameter, ensure that the value of **group** exists and is valid.

Parameter type: string.

Unit: none

Value range:

- To disable the policy-based majority function, enter an empty string **""**.
- To enable the function, configure valid group values separated by commas (,). The group values must exist in **dcf_config**. For example, if the group values 1 and 2 are added to the DCF policy-based majority configuration, you can set this parameter to **"1,2"**. If the group value does not exist in **dcf_config** or other characters are configured, the DCF considers the configured group invalid.

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTICE

If all nodes in a group are faulty after the parameter is configured, you need to remove the group from the parameter list when performing node build operations (node recovery or node replacement without changing the IP address) on a node. After the node recovers, you can configure the group again.

dcf_node_id_map

Parameter description: Specifies the dictionary mapping between standby DN names and DCF node IDs. The parameter setting takes effect after the system is restarted. This parameter is configured during installation and cannot be modified. This parameter is used in DCF cluster installation, upgrade, and node replacement scenarios. The value of **standby_name** configured in the GUC parameter **synchronous_standby_names** must be included in this dictionary.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: string.

Unit: none

Value range: The setting format is **'standby_name1:dcf_node_id1,standby_name2:dcf_node_id2'**. The values of standby DN names and the corresponding DCF node IDs are separated by commas (,).

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Improper settings may cause cluster installation and upgrade failures.

dcf_candidate_names

Parameter description: Specifies the DCF candidate list. That is, names of DNs that may be selected as the primary node. In DCF automatic mode, the election policy is controlled by this parameter. DNs not in the list cannot be elected as a primary node.

Parameter type: string.

Unit: none

Value range: a value in the format of **'dn_name1,dn_name2,dn_name3'**. The parameter depends on **dcf_node_id_map**. DN names must be in **dcf_node_id_map** and separated by commas (,).

Default value: **""**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

dcf_thread_effective_time

Parameter description: Specifies the effective time of the DCF flushing thread. This parameter is used to determine whether the disk I/O hangs. If the DCF control log cannot update and I/O resources cannot be accessed within the period specified by this parameter, the DCF considers that the thread I/O hangs and the switchover is triggered. If this parameter is set to **0**, the I/O hang detection is disabled.

Parameter type: integer.

Unit: second

Value range: 0 to 1000

Default value: 160

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

dcf_pri_leader_timeout

Parameter description: Specifies the timeout interval for priority-based leader election. In DCF automatic mode, after priority-based leader election is enabled, the backup node triggers priority-based leader election. If the backup node is not elected as the leader within the timeout interval, the election is canceled. If this parameter is set to **0**, the priority-based selection of the primary node waits until the election is successful.

Parameter type: integer.

Unit: second

Value range: 0 to 3600

Default value: **60**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a large value, the cluster may be in the no-primary state for a long time.

dcf_static_leader_timeout

Parameter description: Specifies the timeout interval for preferentially electing the old leader. In DCF automatic mode, after the cluster is restarted, the old leader is preferentially elected. If the election fails within the timeout interval, the election is canceled.

Parameter type: integer.

Unit: second

Value range: 0 to 600

Default value: **60**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a large value, the cluster may be in the no-primary state for a long time.

6.3.40 Flashback

This section describes parameters related to the flashback function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_recyclebin

Parameter description: Specifies whether the recycle bin is enabled or disabled in real time.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The recycle bin is enabled in real time.
- **off**: The recycle bin is disabled in real time.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: To use the flashback table function, set **enable_recyclebin** to **on**.

Risks and impacts of improper settings: When this parameter is disabled, the flashback function cannot be used.

NOTICE

After the recycle bin function is enabled, when a table or index is dropped from the database, a hidden object starting with **BIN** is automatically generated. These objects are used to locate the original data during subsequent restoration. All bin objects are created in the schema where the original service table is located.

recyclebin_retention_time

Parameter description: Specifies the retention period of objects in the recycle bin. The objects will be automatically deleted after the retention period expires.

Parameter type: integer.

Unit: second

Value range: 1 to 2147483647

Default value: **900** (that is, 15 minutes)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **900** without a unit, **recyclebin_retention_time** indicates 900s. If the value is **15min**, **recyclebin_retention_time** indicates 15 minutes. The unit must be **s**, **min**, **h**, or **d** if required.

Setting suggestion: Set the retention period of objects in the recycle bin as required.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

undo_retention_time

Parameter description: Specifies the period for retaining undo logs of earlier versions.

Parameter type: integer.

Unit: second

Value range: 0 to 259200. **0** indicates that the flashback point snapshot information is not retained.

Default value: **0**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **900** without a unit, **undo_retention_time** indicates 900s. If the value is **15min**, **undo_retention_time** indicates 15 minutes. The unit must be **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is out of the range or the value contains a unit other than **s**, **min**, **h**, and **d**, an error message is displayed.

NOTICE

- If this parameter is set to **0** during the Ustore flashback query, the snapshot information at the flashback point will be cleared. In earlier versions, no flashback query can be performed. When a flashback query is performed, the error message "cannot find the restore point" is displayed.
- If the time within which the undo logs of earlier versions need to be retained is time1 and the SQL statement execution time for the flashback query is time2, you need to set **undo_retention_time** to a value greater than time1 + time2. That is, set **undo_retention_time** to a value greater than time1 + time2 + 3s. You are advised to set **undo_retention_time** to a value equal to time1 + 1.5 x time2. For example, if you want to retain the logs of earlier versions within the latest 3 hours, and the SQL statement execution time for the flashback query is 1 hour, set **undo_retention_time** to a value equal to 3 hours + 1.5 x 1 hour, that is, 4.5 hours.

6.3.41 Rollback Parameters

max_undo_workers

Parameter description: Specifies the number of undo worker threads called during asynchronous rollback. The parameter setting takes effect after the system is restarted.

Parameter type: integer.

Unit: none

Value range: 1 to 100

Default value: **5**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Only the maximum number of threads that can concurrently perform asynchronous rollback is affected. If the value is too small, the asynchronous rollback speed may be slow. When a large number of transactions that require asynchronous rollback exist in the system, there is an extremely low probability that the undo reclamation speed is affected. If the host configuration is not high and a large number of transactions that require asynchronous rollback exist in the system, a large number of asynchronous rollback threads will be started at the same time, occupying certain system resources.

6.3.42 AI Features

enable_hypo_index

Parameter description: Specifies whether the database optimizer considers the created virtual index when executing **EXPLAIN**. By executing **EXPLAIN** on a specific query statement, you can evaluate whether the index can improve the execution efficiency of the query statement based on the execution plan provided by the optimizer. This parameter can be set at the PDB level.

Parameter type: Boolean

Unit: none

Value range:

- **on**: A virtual index is created when the **EXPLAIN** command is executed.
- **off**: No virtual index is created when the **EXPLAIN** command is executed.

Default value: **off**. In the PDB scenario, if this parameter is not set, the global setting is inherited.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-2**.

Setting suggestion: Set the value based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_ai_stats

Parameter description: Specifies whether to create or use intelligent statistics.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Intelligent statistics are created and used.
- **off**: Intelligent statistics are not created or used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If you modify the settings during database running, the plan may deteriorate.

multi_stats_type

Parameter description: Specifies the type of statistics to be created when **enable_ai_stats** is set to **on**.

Parameter type: enumerated type

Unit: none

Value range: **"BAYESNET"**, **"MCV"**, and **"ALL"**

- **"BAYESNET"**: Only intelligent statistics are created.
- **"MCV"**: Only traditional statistics are created.
- **"ALL"**: Both traditional statistics and intelligent statistics are created.

Default value: **"BAYESNET"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ai_stats_cache_limit

Parameter description: Specifies the maximum number of models that can be cached when **enable_ai_stats** is set to **on**.

Parameter type: integer.

Unit: none

Value range: 30 to 1000

Default value: **100**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Set this parameter based on the number of multi-column statistics models.

Risks and impacts of improper settings: If the value is too large, unnecessary memory is occupied. If the value is too small, models are frequently read from hard disks, deteriorating system performance.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_operator_prefer

Parameter description: Specifies whether to enable the operator preference rule. If the estimated costs are similar, the parameterized path is preferred for table join.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Parameterized path preference is enabled.
- **off**: Parameterized path preference is disabled.

Default value: **off**

NOTE

There are two prerequisites for this parameter to take effect:

- The parameterized path is generated.
- The estimated cost of the parameterized path is similar to that of other index scan operators.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. You are advised to specify this parameter only in sessions or hints.

Risks and impacts of improper settings: If this parameter is modified, the plan may deteriorate.

enable_adaptive_cost

Parameter description: Specifies whether to enable the feedback-based optimizer cardinality and cost correction functions. If this parameter is enabled, the processes of collecting operator information and estimating cardinality are enabled, and the thread for starting backend model maintenance is enabled; otherwise, the thread exits.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The operator information and cardinality estimation processes are enabled.
- **off**: The operator information and cardinality estimation processes are disabled.

Default value:

- **on**: default value of the newly installed database.
- **off**: default value of the database in versions earlier than 506.0.0 after the database is upgraded.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Enable or disable this parameter based on whether automatic optimization is required for query statements.

Risks and impacts of improper settings: If this function is not required, for example, no complex statement that needs to be optimized exists in the service, enabling this function occupies more system resources.

enable_feedback_cardest

Parameter description: Specifies whether to enable the feedback-based optimizer cardinality and cost correction functions. This parameter is used by developers to diagnose model-related problems. If **enable_adaptive_cost** is set to **off**, and this parameter is set to **on**, the operator information is still collected and the API of cardinality estimation feedback is still called. However, in this case, the thread for backend automatic model maintenance is not enabled. Developers can use the gs_acm_analyze_workload_manual() function to manually train models for diagnosing problems.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The operator information is collected, and the API of cardinality estimation feedback is still called. However, the backend automatic model maintenance thread is not enabled.
- **off**: The **enable_adaptive_cost** parameter takes full control.

Default value:

- **on**: default value of the newly installed database.
- **off**: default value of the database in versions earlier than 506.0.0 after the database is upgraded.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

adaptive_cardest_strategy

Parameter description: Specifies the preference of the selection cardinality estimation model.

Parameter type: enumerated type

Unit: none

Value range: **"auto"**, **"use_statistics"**, and **"use_feedback"**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **"auto"**: adaptive mode. In this mode, the system automatically determines whether to use the statistical method or feedback method based on the estimation accuracy in the history.
- **"use_statistics"**: Statistics are preferentially used for cardinality estimation.
- **"use_feedback"**: The feedback model is preferentially used for cardinality estimation.

Default value: **"auto"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is modified during system running, the plan may deteriorate.

maximal_feedback_model_num

Parameter description: Specifies the maximum number of cardinality feedback models. If the actual number exceeds the value of this parameter, no new models will be trained.

Parameter type: integer.

Unit: none

Value range: –1 to 1000000. **-1** indicates no limit.

Default value: **10000**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the feedback cardinality estimation function may become invalid because new models cannot be trained. If the value is too large, too much disk space may be occupied, causing performance deterioration.

feedback_model_cache_limit

Parameter description: Specifies the maximum number of cardinality feedback models that can be cached in the global memory.

Parameter type: integer.

Unit: none

Value range: 10 to 100000

Default value: **500**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If the value is too large, unnecessary memory is occupied. If the value is too small, models are frequently read from hard disks, deteriorating system performance.

feedback_model_expired_time

Parameter description: Specifies the timeout interval for cardinality feedback models (check using gs_abo_model_statistic), which is used to clear expired models periodically.

Parameter type: integer.

Unit: milliseconds

Value range: 500 to 2147483647

Default value: **86400000** (that is, 1 day)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **6000** without a unit, **feedback_model_expired_time** indicates 6000 ms. If the value is **1min**, **feedback_model_expired_time** indicates 1 minute (that is, 60000 ms). The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the timeout interval is too long, the system performance deteriorates. If the timeout interval is too short, the model is cleared frequently, affecting functions.

feedback_collection_expired_time

Parameter description: Specifies the timeout interval for feeding back information about untrained operators. You can view the operator feedback information by using gs_stat_get_acm_feedback_operator_info(). Expired operator information is periodically deleted.

Parameter type: integer.

Unit: milliseconds

Value range: 50 to 2147483647

Default value: **3600000** (that is, 1 hour)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **6000** without a unit, **feedback_collection_expired_time** indicates 6000 ms. If the value is **1min**, **feedback_collection_expired_time** indicates 1 minute (that is, 60000 ms). The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the timeout interval is too long, the system performance deteriorates. If the timeout interval is too short, the feedback data is cleared frequently, affecting functions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

adaptive_cost_min_time

Parameter description: Specifies the execution duration threshold of SQL statements for cardinality feedback collection. Only the feedback of statements whose execution duration is greater than the value of this parameter is collected.

Parameter type: integer.

Unit: milliseconds

Value range: 0 to 2147483647

Default value: **1000**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **6000** without a unit, **adaptive_cost_min_time** indicates 6000 ms. If the value is **1min**, **adaptive_cost_min_time** indicates 1 minute (that is, 60000 ms). The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: You are advised to set this parameter to 20% of the execution duration of slow queries to be tuned.

Risks and impacts of improper settings: If the value is too small, the system performance deteriorates by 1% to 2%. If the value is too large, the query that can be automatically optimized does not take effect.

cost_update_window_size

Parameter description: Specifies the size of the sliding window for collecting regression data.

Parameter type: integer.

Unit: none

Value range: 1 to 20

Default value: **5**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the parameter value is less than 3, the estimated cost parameter may be overfitting and the actual situation may be incorrect. As a result, the optimizer generates an incorrect execution plan.

adaptive_costest_strategy

Parameter description: Specifies whether the new or old cost model is used for cost evaluation.

Parameter type: enumerated type

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **l0**: The new cost model is triggered only when the cardinality estimation is correct (for example, using the cardinality estimation feedback).
- **l1**: The new cost model is preferentially used for calculation.

Default value: **l0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. You are advised to modify this parameter only in sessions or hints.

Risks and impacts of improper settings: If this parameter is set to **l1**, there is a serious risk of plan deterioration.

adaptive_costmodel_calibration_interval

Parameter description: Specifies the interval at which the cost model correction logic is triggered.

Parameter type: integer.

Unit: milliseconds

Value range: 0 to 2147483647. The value **0** indicates that the cost models are not automatically corrected.

Default value: **3600000** (that is, 1 hour)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **6000** without a unit, **adaptive_costmodel_calibration_interval** indicates 6000 ms. If the value is **1min**, **adaptive_costmodel_calibration_interval** indicates 1 minute (that is, 60000 ms). The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the cost model frequently triggers the correction function, deteriorating the system performance. If the value is too large, the model cannot be corrected in a timely manner, affecting the function optimization effect.

adaptcost_extended_feature

Parameter description: Specifies whether to enable the adaptive cost extension function.

Parameter type: string.

Unit: none

Value range:

- **none**: No adaptive cost extension function is used.
- **support_or**: Operators that contain the OR and IN conditions are supported.
- **support_left_join**: Operators that contain the leftjoin or rightjoin condition are supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **support_stream_cost**: Adaptive cost estimation of the stream operator is supported.
- **use_knn_support**: The KNN model is preferentially used to support the IN/OR and leftjoin/rightjoin operators.

Default value:
"support_or,support_left_join,support_stream_cost,use_knn_support"

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Some operators may fail to use the adaptive cost and the plan cannot be optimized.

enable_collect_abo_history

Parameter description: Specifies whether to enable the persistence of feedback cardinality/cost estimation information.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The feedback cardinality/cost estimation information is collected.
- **off**: The feedback cardinality/cost estimation information is not collected.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this function is not required, for example, no complex statement that needs to be optimized exists in the service, enabling this function occupies more system resources.

feedback_record_expired_time

Parameter description: Specifies the timeout interval of records in the gs_abo_history system catalog. Records that exceed the timeout interval are periodically deleted.

Parameter type: integer.

Unit: milliseconds

Value range: 500 to 2147483647

Default value: **86400000** (that is, 1 day)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **6000** without a unit, **feedback_record_expired_time** indicates 6000 ms. If the value is **1min**,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

feedback_record_expired_time indicates 1 minute (that is, 60000 ms). The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the timeout interval is too long, the system performance deteriorates. If the timeout interval is too short, the record is cleared frequently, affecting functions.

enable_ai_watchdog

Parameter description: Enables or disables the AI watchdog function.

Parameter type: Boolean

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Enabling this function occupies system resources.

enable_ai_watchdog_forcible_oom_detection

Parameter description: Forcibly enables or disables the OOM detection function of the AI watchdog. If this parameter is disabled, the system automatically determines whether to enable the OOM detection function based on the current database specifications. In automatic determination mode, the OOM detection function is enabled only when **max_process_memory** is set to **64GB** or a larger value. The OOM detection function depends on the information obtained by the memory management module. Therefore, if the memory management module is disabled or invalid, the OOM detection function is not enabled.

Parameter type: Boolean

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Enabling this function occupies system resources.

enable_ai_watchdog_healing

Parameter description: Enables or disables the self-healing function of the AI watchdog.

Parameter type: Boolean

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ai_watchdog_max_cpu_usage

Parameter description: Specifies the expected upper limit of the database CPU usage. The value is normalized based on the multi-core situation. If this parameter is set to **0**, the system does not check the CPU usage.

Parameter type: floating-point number

Unit: none

Value range: 0 to 1

Default value: **0.8**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, exceptions may not be detected in a timely manner.

ai_watchdog_oom_dynamic_used_threshold

Parameter description: Specifies the expected upper limit of the dynamic memory usage of the database.

Parameter type: floating-point number

Unit: none

Value range: 0 to 1

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: 0.95

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, exceptions may not be detected in a timely manner.

ai_watchdog_oom_growth_confidence

Parameter description: Specifies the confidence level of the OOM detection algorithm.

Parameter type: floating-point number

Unit: none

Value range: 0.1 to 1

Default value: 0.95

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ai_watchdog_oom_malloc_failures

Parameter description: Specifies the maximum number of consecutive memory allocation failures tolerated. If the number of consecutive memory allocation failures exceeds this value, the OOM detection function may be triggered.

Parameter type: integer.

Unit: none

Value range: 1 to 32000

Default value: **50**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, alarms are frequently generated. If this value is too large, exceptions cannot be detected in a timely manner.

ai_watchdog_oom_other_used_memory_threshold

Parameter description: Specifies the expected upper limit of memory usage of other parts of the database.

Parameter type: integer.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: MB

Value range: –1 to 1048576

Default value: The value varies according to the following situations:

1. If this parameter is set through the OM installation, the default value is **-1**.

   NOTE

   When the value is **-1**, the upper limit of the other parts of the memory usage is specified based on the following rules:

   1. If the value of **max_process_memory** is less than 300 GB, the upper limit of the other parts of the memory usage is 20 GB.
   2. If the value of **max_process_memory** is greater than or equal to 300 GB and less than 600 GB, the upper limit of the other parts of the memory usage is 40 GB.
   3. If the value of **max_process_memory** is greater than or equal to 600 GB, the upper limit of the other parts of the memory usage is 60 GB.

   If the default value **-1** is not used, the upper limit of the other parts of the memory usage is the value set by the user.

2. If this parameter is not set through the OM installation, the default value is **20480**.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, alarms are frequently generated. If this value is too large, exceptions cannot be detected in a timely manner.

ai_watchdog_oom_process_threshold

Parameter description: Specifies the expected percentage of the database process usage to the value of **max_process_memory**. When the threshold is reached, memory leakage determination is triggered.

Parameter type: floating-point number

Unit: none

Value range: 0 to 10

Default value: The value varies according to the following situations:

1. If this parameter is set through the OM installation, the default value is **1**.
2. If this parameter is not set through the OM installation, the default value is **1.1**.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

ai_watchdog_oom_shared_threshold

Parameter description: Specifies the expected upper limit of the shared memory usage of the database.

Parameter type: floating-point number

Unit: none

Value range: 0 to 1

Default value: 0.4

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ai_watchdog_rto_restriction_time

Parameter description: Specifies the RTO threshold of the AI watchdog self-healing function. If the RTO threshold is exceeded, self-healing is not performed.

Parameter type: integer.

Unit: seconds

Value range: 0 to 36000

Default value: **600**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ai_watchdog_tolerance_times

Parameter description: Specifies the maximum number of consecutive abnormal events that can be tolerated by the AI watchdog before self-healing is started. This parameter can be used to avoid incorrect operations.

Parameter type: integer.

Unit: none

Value range: 0 to 100

Default value: **4**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ai_watchdog_tps_threshold

Parameter description: Specifies the lower limit of the expected TPS usage of the database instance. If the TPS usage is lower than the value of this parameter, the exception determination logic is triggered.

Parameter type: integer.

Unit: none

Value range: 0 to 32000

Default value: **2**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ai_watchdog_wait_time

Parameter description: Adjusts the waiting time. To prevent the database from frequently performing self-healing operations, the database waits for a period of time after startup.

Parameter type: integer.

Unit: seconds

Value range: 0 to 36000

Default value: **1800**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value cannot contain a unit.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, false alarms may be reported when the database is just started. If the value is too large, exceptions within the waiting time may not be detected.

ai_watchdog_warning_retention

Parameter description: Specifies the maximum number of alarm records that the AI watchdog can retain in the dbe_perf.ai_watchdog_detection_warnings view.

Parameter type: integer.

Unit: none

Value range: 0 to 32000

Default value: 20

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.43 Global SysCache Parameters

enable_global_syscache

Parameter description: Specifies whether to enable the global system cache function.

Parameter type: Boolean

Unit: none

Value range:

- **on** indicates that the global system cache function is enabled.
- **off** indicates that the global system cache function is disabled.

Default value: **on**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Enabling the global system cache function can reduce the system cache memory usage and improve the concurrent expansion capability. If this parameter is enabled, **global_syscache_threshold** must be used together. In this case, the thread pool mode is recommended. After this parameter is enabled, you are advised to set **wal_level** of the standby node to **hot_standby** or higher level if you need to access the standby node.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

global_syscache_threshold

Parameter description: Specifies the maximum memory usage of the global system cache. To use this parameter, you need to enable the **enable_global_syscache** parameter.

Parameter type: integer.

Unit: KB

Value range: 16384 to 1073741824

Default value: **163840** (160 MB)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **163840** without a unit, it indicates 163840 KB. If this parameter is set to **160MB**, it indicates 160 MB. The unit must be KB, MB, or GB if required.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Set it to the smaller value of the number of hot databases and the number of threads multiplied by the memory size allocated to each database, that is, **global_syscache_threshold** = **min(count(hot dbs),count(threads))** x **memofdb**.

The number of hot databases is the number of databases that are frequently accessed. In thread pool mode, the number of threads is the sum of the number of threads in the thread pool and the number of background threads. In non-thread pool mode, the number of hot databases is used.

memofdb indicates the average memory allocated to each database. The background noise memory of each database is 2 MB. Each time a table or index is added, 11 KB memory is added.

Risks and impacts of improper settings: If the value is too small, memory is frequently evicted, and a large number of memory fragments cannot be recycled. As a result, memory control fails.

6.3.44 Read Parameters of the Standby Node in a Distributed System

enable_standby_read

Parameter description: Specifies whether to enable the read function of the standby node for a session.

NOTICE

This function can be enabled only in GTM-free, GTM-lite, non-standby cluster for DR, non-transaction block, and hot-standby modes.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The read function of the standby node in a distributed system is enabled for the session.
- **off**: The read function of the standby node in a distributed system is disabled for the session.

Default value: **off**

Setting method: This parameter is a USERSET parameter and can be set only at the session level (**enable_standby_read** is set to **on**). It cannot be set using gs_guc set/reload.

Setting suggestion: This parameter cannot be set using gs_guc. If you need to query the standby node, enable it in the session.

Risks and impacts of improper settings: If this parameter is set at a non-session level (for example, enabling this parameter in the configuration file), the cluster cannot be started when a parameter conflict occurs. If no parameter conflict

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

occurs, the read function of the standby node in a distributed system is also enabled for background threads such as autovacuum and WorkloadMonitor. As a result, DDL and DML operations are affected and errors are reported.

enable_direct_standby_datanodes

Parameter description: Specifies whether to enable the EXECUTE DIRECT ON extension capability for a session. After this parameter is enabled, the EXECUTE DIRECT ON statement can be connected to any standby DN or all standby DNs of a shard to execute queries. For details about the EXECUTE DIRECT ON syntax, see **EXECUTE DIRECT**.

NOTICE

- This function can be enabled only in GTM-free mode, non-transaction block, and hot-standby mode.
- In the same session, an error is reported if **enable_direct_standby_datanodes** is set to **on** after **enable_standby_read** is set to **on**. An error is also reported if **enable_standby_read** is set to **on** after **enable_direct_standby_datanodes** is set to **on**.
- After this parameter is enabled, the current session is read-only, and DDL and DML jobs are affected and errors are reported.
- After this parameter is enabled, you can use EXECUTE DIRECT ON to query only system DFX information such as system views, system functions, and system catalogs. Common user tables cannot be queried.
- After the **enable_direct_standby_datanodes** parameter is enabled, EXECUTE DIRECT ON supports distributed standby node query. Before using this function during the upgrade, ensure that the CN and DN instance processes have been replaced with the latest binary files. Otherwise, packet verification fails and an error is reported.
- After this parameter is enabled, the data queried from system views such as dbe_perf.statement and dbe_perf.summary_statement is the data on independent CNs and does not include the data on the primary DN. You are advised not to enable this parameter for data query.
- To query data on DNs, use EXECUTE DIRECT ON DATANODES.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The EXECUTE DIRECT ON extension capability is enabled for the session.
- **off**: The EXECUTE DIRECT ON extension capability is disabled for the session.

Default value: **off**

Setting method: This parameter is a USERSET parameter and can be set only at the session level (**enable_direct_standby_datanodes** is set to **on**). It cannot be set using gs_guc set/reload.

Setting suggestion: This parameter is used to specify whether to enable EXECUTE DIRECT ON to perform the connection behavior in the distributed standby node

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

read scenario. This function is mainly used for WDR, database test, and O&M. You are advised not to enable this function in other scenarios. This parameter cannot be set using gs_guc. If you need to query the standby node, enable it in the session.

Risks and impacts of improper settings: If this parameter is enabled in the configuration file, the database node cannot be started.

standby_read_delay

Parameter description: Specifies the maximum difference between the primary and standby nodes when data is read from the standby node. If the difference exceeds the value of this parameter, data cannot be read from the standby node.

Parameter type: integer.

Unit: ms

Value range: –1 to 2147483647

- **-1**: The difference between the primary and standby nodes does not need to be verified when data is read from the standby node.
- **0**: Data can be read from the standby node only when there is no difference between the primary and standby nodes.

Default value: **10000000**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **86400** without a unit, it indicates 86400 ms. If this parameter is set to **1d**, it indicates 1 day. The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Set this parameter based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

standby_read_rto

Parameter description: Specifies the maximum RTO of the system when data is read from the standby node. If the RTO exceeds the value of this parameter, data cannot be read from the standby node.

Parameter type: integer.

Unit: ms

Value range: –1 to 2147483647

- **-1**: The system RTO does not need to be verified when data is read from the standby node.
- **0**: Data can be read from the standby node only when the RTO of at least one standby node is 0.

Default value: **60000**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. For example, if this parameter is set to **86400** without a unit, it

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

indicates 86400 ms. If this parameter is set to **1d**, it indicates 1 day. The unit must be **ms**, **s**, **min**, **h**, or **d** if required.

Setting suggestion: Set this parameter based on service requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.45 Restoring Data on the Standby Node

standby_page_repair

Parameter description: Specifies whether to automatically repair the page after a fault is detected on the replay target page during replay on the standby node. The current version supports three types of rectification: CRC failure, uninitialized page, and page LSN verification error. Bad blocks of VM, Xlog, Clog, MOT, and undo metadata cannot be repaired, and LSN errors of FSM cannot be repaired. Undo records can be repaired only when CRC fails. Temporary tables, global temporary tables, and unlogged tables do not involve primary and standby restoration. The page file-level loss cannot be repaired.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The standby node automatically detects and repairs pages during replay.
- **off**: The standby node does not automatically detect and repair pages during replay.

Default value: **on**

NOTICE

- If a large number of bad blocks exist due to a standby node exception, the replay performance of the standby node is affected, and Xlogs may be stacked on the standby node. In addition, an exception log is printed each time an Xlog is replayed to a bad block.
- To ensure the consistency between the primary and standby nodes, if the standby node has unrepaired bad blocks, the checkpoint needs to be stopped. As a result, Xlogs may be stacked, and the replay on the bad blocks will be suspended, which may cause inconsistency between the primary and standby nodes. If bad blocks exist when the standby node is restarted or promoted to primary, the instance will exit abnormally. Before the exit, the Xlog chain on the bad blocks will be collected, which may prolong the RTO.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If automatic repair is disabled, bad blocks will be stored in the bad block table once bad blocks are detected, causing the adverse impact described in NOTICE.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3.46 Ledger Database

enable_ledger

Parameter description: Specifies whether to enable the ledger database. If this parameter is set to **on**, the ledger database is enabled, and a new tamper-proof mode can be created and the common mode can be changed to the tamper-proof mode.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The ledger database is enabled.
- **off**: The ledger database is disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: The ledger database is disabled by default. To enable it, set **enable_ledger** to **on**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ledger_hist_level

Parameter description: Specifies whether to record SQL statements in the global blockchain table.

Parameter type: integer

Unit: none

Value range:

- **0**: SQL statements are not recorded in the global blockchain table.
- **1**: SQL statements are recorded in the global blockchain table.

Default value: **1**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.47 Creating an Index Online

delete_cctmp_table

Parameter description: Specifies whether to delete temporary tables generated during Ustore online index creation or rebuilding.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: indicates that the temporary tables are deleted.
- **off**: indicates that the temporary tables are retained.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is enabled by default. Set this parameter to **off** if you want to retain temporary tables generated during online index creation or rebuilding.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_bypass_insert_sequence

Parameter description: Specifies whether SQL Bypass is supported when an INSERT statement of a table that contains auto-increment columns.

NOTE

- This parameter takes effect only when the GUC parameter **enable_opfusion** is enabled.
- The types of auto-increment columns are smallint, int, and bigint.
- The constraints of the SQL Bypass insert statement are inherited.
- Auto-increment sequence is supported only for INSERT statements.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: SQL Bypass optimization is supported.
- **off**: SQL Bypass optimization is not supported.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **18.2-Table 1 GUC parameter types**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, INSERT statements containing auto-increment lists cannot be optimized using SQL Bypass, which reduces the execution efficiency of INSERT statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3.48 Online DDL Operations

enable_online_ddl

Parameter description: Specifies whether to enable online DDL or online VACUUM. This parameter is used together with the ONLINE/OFFLINE keyword. (For details about the keyword, see **ALTER TABLE** and **VACUUM**.)

Parameter type: Boolean.

Unit: none

Value range:

- **on**: Online DDL is enabled and does not take effect in the current version.
- **off**: Online DDL is disabled and does not take effect in the current version.

Default value: **off**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: You are advised to disable this parameter. This parameter does not take effect in the current version. If you need to use the new online DDL features, add the keyword ONLINE to the syntax.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.49 Data Lifecycle Management: OLTP Table Compression

enable_ilm

Parameter description: Specifies whether to enable the OLTP table compression of data lifecycle management.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: The default value is **off**. To enable this feature, contact Huawei technical support to purchase a license and then set this parameter to **on**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

ilm_instance_compress

Parameter description: Sets a specified ILM compression policy to an instance-level compression policy. Currently, only the cross-page transparent compression (that is, TURBO) policy can be specified as an instance-level policy. When **ilm_instance_compress** is set to **turbo**, the TURBO compression policy is automatically added to all objects that support cross-page transparent compression in the instance.

Parameter type: enumerated type

Unit: none

Value range:

- **off**: disabled.
- **turbo**: enabled. The instance-level compression policy type is **turbo**.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is disabled by default. To enable instance-level compression, set it to **turbo**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ilm_instance_compress_configuration

Parameter description: Specifies the default configuration of an instance-level compression policy, which consists of the compression level and compression cold and hot management time. For example, the default value indicates that the data that is not modified within one day in the compressed object is compressed at the low level. The time range cannot exceed 3 years.

Parameter type: string.

Unit: none

Value range:

- Compression level: **low**, **medium**, or **high**
- Hot and cold management time: an integer
- Cold and hot management time unit: **hour**, **day**, **month**, or **year**

Default value: **low,1,day**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value **low,1,day**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

turbo_compress_shared_buffers

Parameter description: Specifies the size of the metadata buffer dedicated for cross-page transparent compression.

Parameter type: integer.

Unit: 8 KB (per page)

Value range: 0 to 1073741823

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**. For example, if the value is **1024** without a unit, the value of **turbo_compress_shared_buffers** is 1024 x 8 KB, that is, 8192 KB. If the value is **8MB** with a unit, the value of **turbo_compress_shared_buffers** is 8 MB. If the value contains a unit, the value must be KB, MB, or GB and must be an integer multiple of 8 KB.

Setting suggestion: The default value is **0**, indicating that cross-page transparent compression is disabled. If this function is required, set this parameter to 1/128 of **shared_buffers**. If a transparently compressed table already exists, this parameter cannot be set to **0**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

turbo_compress_default_configuration

Parameter description: Specifies the default configuration of cross-page transparent compression. A semicolon (;) is used to separate different levels, which are **low**, **medium**, and **high** in sequence. Commas (,) are used to separate compression configurations at different levels. The configurations are the encoder size, compressor size, and compression unit size in sequence.

Parameter type: string.

Unit: none

Value range:

- Encoder: **tiling**, **smartdiff**, and **pattern**
- Compressor: **zstd** and **lz4**
- Compression unit size: **1**, **2**, **4**, **8**, **16**, **32**, and **64**

Default value: **tiling,zstd,4;tiling,zstd,16;pattern,zstd,16**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value **tiling,zstd,4;tiling,zstd,16;pattern,zstd,16**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

turbo_compress_pattern_encoder

Parameter description: Specifies the mode encoder for cross-page transparent compression. When the encoder in **turbo_compress_default_configuration** is set to **pattern**, you can set this parameter to specify the mode encoding algorithm to be used. Multiple mode encoding algorithms can be combined and separated by vertical bars (|).

Parameter type: string.

Unit: none

Value range:

- **none**: no encoding.
- **transpose**: row and column transpose.
- **transpose|column_encode**: row and column transpose and column encoding.
- **transpose|dedup|column_encode**: row and column transpose, column encoding, and deduplication.

Default value: **transpose**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value **transpose**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTICE

- Whether the pattern code has positive benefits depends on the data feature. If the evaluation does not generate positive benefits, the pattern code is automatically closed.
- Mode encoding requires extra memory. When the memory resources are insufficient, mode encoding is automatically disabled.

turbo_compress_enable_double_write

Parameter description: Specifies whether to enable cross-page transparent compression doublewrite. After this function is disabled, compressed data may be damaged in abnormal breakdown scenarios.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **on**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- A complete cross-page transparent compression doublewrite requires recording both metadata and actual data.
- If both **turbo_compress_enable_double_write** and **enable_double_write** are set to **on**, doublewrite is recorded for both parts. If both parameters are set to **off**, doublewrite is not recorded for both parts. If only one parameter is set to **on**, doublewrite is recorded only for metadata.
- If the doublewrite operation is not recorded, the damaged metadata cannot be restored. If only the doublewrite operation is recorded, the damaged data can be restored using HA.

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value **on**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

turbo_compress_preload_workers_num

Parameter description: Specifies the number of threads for prefetching compressed pages in parallel replay or ultimate RTO mode.

Parameter type: integer.

Unit: none

Value range: 0 to 16

Default value: **0**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value **0**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

turbo_compress_force_fast_allocate

Parameter description: Specifies whether to use fast preemption to allocate space during table compression. By default, **0** is used for allocation.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Set this parameter to **on** for file systems, such as ext4, that support fast preemption to improve the expansion speed of compressed tables.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.50 Session-level Transaction Pushdown and Related Parameters

session_type

Parameter description: Specifies the type of the current session. The value is case-insensitive. An execution plan is generated based on the actual value.

Parameter type: enumerated type

Unit: none

Value range:

- **single_node**: single-node session. Read and write operations can be pushed down. Nodes to be pushed down are determined and an execution plan is generated based on the GUC parameter **exec_target_node**.
- **global_read_only**: cross-node read-only session. A distributed plan is generated normally, but distributed write operations are forbidden.
- **normal**: default session, which is the same as the original behavior. The plan is executed based on the original logic of the database.

Default value: **normal**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-2**. Note that this parameter cannot be set globally using gs_guc.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: When data is inserted in single-node session mode (**session_type** is set to **single_node**), correct data distribution is ensured by the application layer. If an incorrect node is specified, data may be inconsistent.

exec_target_node

Parameter description: Specifies the name of the DN where DML statements and some DDL statements (TRUNCATE and ALTER TABLE...TRUNCATE PARTITION...) of the current session are executed. The name is case-sensitive. If the DN name is invalid, an error is reported during the setting.

NOTE

This parameter is valid only when the GUC parameter **session_type** is set to **single_node**.

Parameter type: string.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: names of all DNs in the current cluster. You can obtain the names of all DNs by querying the PGXC_NODE system catalog.

Default value: ""

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-2**. Note that this parameter cannot be set globally using gs_guc.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: In single-node session mode, if the parameters are not used according to the restrictions, data inconsistency may occur.

NOTE

The single-node session mode has the following restrictions:

- The router function does not take effect in this scenario.
- The DN specified by **node_name hint** does not take effect and is delivered by **exec_target_node**.
- Statements involving row-level security are not supported.
- Reading on the distributed standby node is not supported.
- Advanced package statements are not supported.
- GPC is not supported.
- In single-node session mode, when the PBE statement is used, the plan type is directly set to **GPlan**. If the single-node session mode is specified after a plan is generated, the original plan type remains unchanged.

Examples

```
gaussdb=# SET log_min_messages = debug2;
gaussdb=# SET logging_module = 'on(opt_choice)';
gaussdb=# CREATE TABLE t1(a int ,b int);
gaussdb=# PREPARE x(int) AS SELECT * FROM t1;
gaussdb=# EXPLAIN EXECUTE x(10);
NOTICE:  [Choosing C/G/A Plan]: GPlan, reason: Using pbe optimization.
NOTICE:  [Choosing C/G/A Plan]: CPlan, reason: First 5 times using CPlan.
                QUERY PLAN
--------------------------------------------------
 Data Node Scan  (cost=0.00..0.00 rows=0 width=0)
   Node/s: All datanodes
(2 rows)
gaussdb=# DEALLOCATE x;

gaussdb=# PREPARE x(int) AS SELECT * FROM t1;
gaussdb=# SET session_type=single_node;
gaussdb=# SET exec_target_node=datanode1;
gaussdb=# EXPLAIN EXECUTE x(10);
NOTICE:  [Choosing C/G/A Plan]: GPlan, reason: Using pbe optimization.
NOTICE:  [Choosing C/G/A Plan]: GPlan, reason: Using pbe optimization.
                QUERY PLAN
--------------------------------------------------
 Data Node Scan  (cost=0.00..0.00 rows=0 width=0)
   Node/s: datanode1
(2 rows)
gaussdb=# DEALLOCATE x;
gaussdb=# DROP TABLE t1;
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3.51 Enhanced TOAST

enable_enhance_toast_table

Parameter description: Specifies whether the enhanced TOAST out-of-line storage table is used.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: The default value of this parameter is **on**. To disable the enhanced TOAST out-of-line storage function, set this parameter to **off**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.52 Global Temporary Table

max_active_global_temporary_table

Parameter description: Specifies whether global temporary tables can be created. The value of this parameter determines the memory reserved in the shared cache for hash tables required by global temporary tables. The total number of active global temporary tables in all sessions is not forcibly limited.

Parameter type: integer.

Unit: none

Value range: 0 to 1000000

- **0**: The global temporary table function is disabled.
- > 0: The global temporary table function is enabled.

Default value: **1000**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

vacuum_gtt_defer_check_age

Parameter description: Specifies the difference between the minimum value of **relfrozenxid** of all global temporary tables and that of all ordinary tables in the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

current database after VACUUM is executed. If the minimum value of **relfrozenxid** of all global temporary tables is greater than the specified value, the alarm "global temp table oldest relfrozenxid %lu is the oldest in the entire db" is generated to remind users to manually execute VACUUM. Otherwise, Clogs may not be recycled.

Parameter type: integer.

Unit: none

Value range: 0 to 1000000

Default value: **10000**.

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_gtt_concurrent_truncate

Parameter description: Specifies whether to support concurrent execution of TRUNCATE TABLE and DML operations on global temporary tables and concurrent execution of TRUNCATE TABLE on global temporary tables.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The preceding operations can be performed concurrently.
- **off**: The preceding operations cannot be performed concurrently.

Default value: **on**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

global_gtt_cn_session_key

Parameter description: Specifies the ID of the current session, that is, the global unique session ID of a distributed global temporary table.

Parameter type: string.

Unit: none

Value range: a string.

Default value: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This is an internal parameter and cannot be set by users. No query result is returned.

Setting suggestion: No setting is allowed.

Risks and impacts of improper settings: none

gtt_reuse_wait_timeout

Parameter description: Specifies the maximum timeout interval for the current session to wait for the previous session to release related resources when the current session reuses the GTT session mapping entry during the REUSE operation on a distributed global temporary table.

Parameter type: integer

Unit: milliseconds

Value range: 0 to 1200000

Default value: **5000** (5s)

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If the network latency is high or errors are frequently reported when long transactions frequently use global temporary tables, you can increase the value of this parameter to prevent REUSE failures caused by waiting timeout.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing. If the value is too small, REUSE timeout errors may occur within the normal cleanup period, affecting service continuity. If the value is too large, DN threads may be blocked in REUSE waiting for a long time, affecting the response of the threads to other requests.

6.3.53 Single-Node Table

default_nodegroup

Parameter description: Specifies whether to enable the function of setting single-node attributes by default when a schema is created.

Parameter type: string.

Unit: none

Value range:

- **NONE**: The function of setting single-node attributes is disabled by default when a schema is created.
- **SCHEMA**: The function of setting single-node attributes is enabled by default when a schema is created.

Default value: **NONE**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

If **default_nodegroup** is set to **SCHEMA**, only common users are allowed to set single-node attributes by default. Administrators, O&M personnel, and initial user are not allowed to set single-node attributes by default.

single_node_table_strategy

Parameter description: Specifies the behavior of creating single-node tables.

Parameter type: string.

Unit: none

Value range: **FORBID**, **ALLOW_STRICT**, **ALLOW_DEFAULT_NON_PART**, and **ALLOW_DEFAULT_ALL**

- **FORBID**: Disables the function of creating single-node tables, including the capability of converting other tables into single-node tables.
- **ALLOW_STRICT**: Enables the function of explicitly creating single-node tables, including partitioned and non-partitioned tables.
- **ALLOW_DEFAULT_NON_PART**: Enables the function of creating a new single-node table by default based on the single-node attribute of the schema when no distribution policy is specified and the TO GROUP clause is not specified for non-partitioned tables. This function is enabled on the basis of the function of explicitly creating single-node tables.
- **ALLOW_DEFAULT_ALL**: Enables the function of creating a single-node table by default based on the single-node attribute of the schema when no distribution policy is specified and the TO GROUP clause is not specified for partitioned tables and non-partitioned tables. This function is enabled on the basis of the function of explicitly creating single-node tables.

Default value: **ALLOW_STRICT**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

NOTE

Single-node tables support backup and restoration. If the GUC parameter **single_node_table_strategy** is enabled in the source database and a single-node table exists, but the GUC parameter **single_node_table_strategy** is disabled in the target database, the following situations may occur during backup and restoration:

- For database-level backup and restoration: The restoration can be performed properly, and the table has the single-node attribute.
- For table-level backup and restoration: An error is reported. You need to modify the parameter (enable the GUC parameter **single_node_table_strategy**) before performing the restoration.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.3.54 HTAP

This section describes the parameters related to the hybrid transactional/analytical processing (HTAP) feature.

enable_htap

Parameter description: Specifies whether to enable the HTAP feature. After the HTAP feature is enabled, the inventory row-store data of the column specified by the COLVIEW keyword is loaded to the in-memory column view (IMCV). It is used to improve the query efficiency of the analytical processing (AP).

Parameter type: Boolean

Unit: none

Value range:

- **on**: The HTAP feature is enabled.
- **off**: The HTAP feature is disabled.

Default value: **off**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: This parameter is disabled by default. If you need to enable this parameter, consider the database application scenario. When a large number of APs need to be queried, enable this parameter to improve the query efficiency. You are advised not to enable this parameter in only TP query scenarios.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

htap_max_mem_size

Parameter description: Specifies the upper limit of the available memory for the HTAP feature. This parameter takes effect only when the HTAP feature is enabled. After the dynamic memory usage of GaussDB reaches the upper limit, no more memory can be allocated even if the HTAP memory usage does not reach the upper limit.

Parameter type: integer.

Unit: KB

Value range: 102400 to 1073741824

Default value: **1048576**, that is, 1 GB.

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**. The value can be specified without a unit, for example, **1000000** (indicating 1000000 KB), or with a unit, for example, **1GB** (indicating 1 GB). The unit must be KB, MB, or GB if required. You can increase the value of this parameter without restarting the system. If you decrease the value of this parameter, no error is reported, but the change takes effect only after the system is restarted. That is, the value before the restart is the original value, and the value after the restart is the new value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: The value of this parameter cannot exceed the value of **max_process_memory** and the maximum shared memory. Allocate idle memory as the value of this parameter based on the actual application scenario. If **htap_memctl_policy** is set to **swap** or **escape**, the value of **htap_used_memory** may be slightly greater than that of **htap_max_mem_size** during loading and rebuilding. After the loading or rebuilding is complete, the value of **htap_used_memory** automatically falls below that of **htap_max_mem_size**.

In extreme cases, the memory overflow during manual loading or rebuilding is calculated as follows: Number of tables loaded at the same time x **htap_rowgroup_pagenum** x 8 KB x Degree of parallelism (for details, see **gs_imcv_flush**).

The overflow caused by automatic rebuilding in extreme cases during service running is calculated as follows: **htap_flush_workers** (number of background threads) x **htap_rowgroup_pagenum** x 8 KB.

Risks and impacts of improper settings: The **htap_max_mem_size** setting is based on the dynamic memory and memory management mechanism of GaussDB. If **max_process_memory** is set improperly, **htap_max_mem_size** is set to a large value, or logical memory management is disabled, the system memory may be insufficient. If **htap_max_mem_size** is set to a small value, an insufficient memory exception may be frequently triggered or the IMCV table may be flushed to disks during IMCV creation based on the value of **htap_memctl_policy**.

NOTE

- The maximum available memory of this parameter refers to the memory occupied by user data in column-store format in HTAP. In addition, the memory of metadata such as IMCU descriptors is not counted in this range. Therefore, the actual memory used by HTAP is greater than this value.
- A row group in HTAP column-store format is mapped to a certain number of pages in row-store format, that is, the parameter **htap_rowgroup_pagenum**. The number of IMCUs contained in a row group corresponds to the number of columns in a table. Therefore, the metadata background noise is affected by the data volume, number of columns, and number of row-store page mappings of the table. In the large data volume scenario where the number of page mappings is 8192, the number of columns in a single table is 100, and the data volume loaded in the column-store format is 1 TB, the HTAP metadata background noise is within 1.5 GB.
- You can query the **max_dynamic_memory** (maximum dynamic memory) and **dynamic_used_memory** (used dynamic memory) in the gs_total_memory_detail view.

htap_rowgroup_pagenum

Parameter description: Specifies the relationship between the number of row groups and the number of row-store page mappings of the HTAP feature. This parameter takes effect only when the HTAP feature is enabled.

Parameter type: integer.

Unit: none

Value range: **1000**, **2048**, **4096**, and **8192**

Default value: **1000**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value for small data volume scenarios where the column-store data is less than 100 GB. Use the maximum value 8192 for column-store data scenarios where the column-store data is more than 1 TB.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

htap_eviction_threshold

Parameter description: Specifies the threshold for swapping in and out the HTAP feature buffer. This parameter takes effect only when the HTAP feature is enabled.

Parameter type: integer.

Unit: %

Value range: 0 to 100

- **0**: No IMCU is loaded to the shared buffer.
- **100**: The buffer evicts only when the memory usage exceeds the upper limit. No eviction or loading is performed in other time.
- Other values: The buffer starts the eviction task only when the HTAP memory usage reaches **htap_max_mem_size** x **htap_eviction_threshold**/100.

Default value: **90**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. If you set this parameter to **0** or **100**, you need to fully understand the parameter meaning and evaluate the impact.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

htap_memctl_policy

Parameter description: Specifies the memory management policy mode of the HTAP feature. This parameter takes effect only when the HTAP feature is enabled.

Parameter type: enumerated type

Unit: none

Value range:

- **delete**: This value is deprecated. After the value is forcibly changed, the value that takes effect is **escape**.
- **swap**: When the HTAP memory is insufficient, data exchange occurs. The Anti-Cache and lazy loading policies are used. Infrequently used data is asynchronously flushed to the local disk. If a table is loaded in the column-store format by using DDL statements, the excess data is flushed to disks when the memory is insufficient.
- **escape**: When the HTAP memory is insufficient, only the metadata of new column-store data is stored. During data query, the data is read back from

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

rows instead of being stored in the column-store format, which is similar to the row group where all pages are dirty pages.

Default value: **swap**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

This policy takes effect in the following two memory insufficiency scenarios:

1. The HTAP memory usage reaches the upper limit specified by **htap_max_mem_size**.
2. The global dynamic memory usage reaches the upper limit of dynamic memory management.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_imcvscan

Parameter description: Specifies whether to enable the HTAP memory column scanning operator IMCVScan. This parameter takes effect only when the HTAP feature is enabled.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The HTAP memory column scanning operator IMCVScan is enabled.
- **off**: The HTAP memory column scanning operator IMCVScan is disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_hybrid_imcvscan

Parameter description: Specifies whether to enable the HTAP column-to-row hybrid scan feature. This parameter takes effect only when the HTAP feature is enabled and depends on the value of **enable_imcvscan**.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The HTAP column-row hybrid scan feature is enabled. This feature is enabled only when **enable_imcvscan** is set to **on**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **off**: The HTAP column-row hybrid scan feature is disabled, regardless of the value of **enable_imcvscan**.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

imcvscan_consistency

Parameter description: Specifies the consistency semantics of column-store scanning. This parameter takes effect only when the HTAP feature is enabled and depends on the value of **enable_imcvscan**.

In strong consistency semantics, column-store scanning always keeps consistent with the row-store result regardless of whether there are unsynchronized row-store changes in the column-store data.

In weak consistency semantics, only column-store data is scanned. If there are unsynchronized row-store changes in the column-store data, consistency with the row-store result is not guaranteed. Column-to-row conversion and standby node IMCV scan are not supported.

Parameter type: enumerated type

Unit: none

Value range:

- **strong**: strong consistency, which ensures strong consistency with the row-store scan result.
- **weak**: weak consistency. Only column-store data is scanned. If column-store data has unsynchronized row-store changes, the consistency with the row-store result is not guaranteed.

Default value: **strong**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. You are advised to use weak consistency semantics only when there is no strong consistency requirement between column-store scan results and row-store data. Weak consistency semantics does not support column-to-row conversion. When **enable_hybrid_imcvscan** is set to **on**, the setting of **imcvscan_consistency** to **weak** is ignored and strong consistency semantics is used.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing. In weak consistency semantics, do not use the column-store scan result as the basis for service operations to avoid data problems.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

htap_router_mode

Parameter description: Specifies the HTAP transparent routing mode. This parameter takes effect only when the HTAP feature is enabled.

Parameter type: enumerated type

Unit: none

Value range:

- **row**: The query is forcibly loaded to select a row-store plan.
- **column**: When columns in a query request are loaded to IMCV, the column-store plan IMCVScan is forcibly executed regardless of the cost. If some columns are not loaded to IMCV, the row-store plan is executed.
- **auto**: The system automatically selects a hybrid plan based on whether the columns in the query request are loaded to IMCV and the cost of the row-column plan.

Default value: **row**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If there are hybrid loads, you are advised to set this parameter to **auto**.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

htap_flush_workers

Parameter description: Specifies the maximum number of background threads for HTAP. This parameter takes effect only when the HTAP feature is enabled. The background threads are used to synchronize incremental row-store data to IMCV row groups, asynchronously load data, and swap data in and out of the buffer.

Parameter type: integer.

Unit: none

Value range: 1 to 256

Default value: **3**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: If multiple tables need to be loaded, increase the value of this parameter. After the loading is complete, decrease the value to the default one.

Risks and impacts of improper settings: If the value is too large, the TP performance may be affected. If the value is too small, the AP performance may be affected.

htap_flush_pagenum

Parameter description: Specifies the dirty page threshold for synchronizing incremental row-store data to the IMCV row group. If the number of dirty pages is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

greater than the threshold, synchronization is triggered. This parameter takes effect only when the HTAP feature is enabled.

Parameter type: integer.

Unit: none

Value range: -1 or 1 to 8192. The value **-1** indicates that the threshold is infinite and synchronization is not performed based on the number of dirty pages.

Default value: **512**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Decreasing the value of this parameter improves AP performance, but may affect TP performance. Increasing the value of this parameter affects AP performance. The value of this parameter is associated with that of **htap_rowgroup_pagenum**. If the value is greater than the number of page mappings, the value **htap_rowgroup_pagenum** is used.

Risks and impacts of improper settings: If the value is too small, the TP performance may be affected. If the value is too large, the AP performance may be affected. The default value is recommended.

htap_flush_interval

Parameter description: Specifies the maximum time interval for synchronizing incremental row-store data to the IMCV row group. This parameter takes effect only when the HTAP feature is enabled.

Parameter type: integer.

Unit: seconds

Value range: -1 or 1 to 36000. The value **-1** indicates that the threshold is infinite and synchronization is not performed based on the interval.

Default value: **300**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. Decreasing the value of this parameter improves AP performance, but may affect TP performance. Increasing the value of this parameter affects AP performance.

Risks and impacts of improper settings: If the value is too small, the TP performance may be affected. If the value is too large, the AP performance may be affected. The default value is recommended.

enable_async_populate

Parameter description: Specifies whether the **ALTER ... COLVIEW** command waits for column-store data to be loaded before returning a result. This parameter depends on whether the HTAP feature is enabled. In distributed mode, column-store data cannot be synchronously loaded. The value **on** takes effect regardless of the setting.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The **ALTER ... COLVIEW** command returns a result immediately after creating metadata. The actual data loading to IMCV is performed in the background.
- **off**: After the **ALTER ... COLVIEW** command creates metadata, the command returns a result only after the background thread loads data to IMCV.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

htap_deploy_mode

Parameter description: Specifies the HTAP deployment mode. This parameter takes effect only when the HTAP feature is enabled. In distributed mode, HTAP can be deployed only on the primary node. In distributed mode, the value **primary** takes effect regardless of the setting.

Parameter type: enumerated type

Unit: none

Value range:

- **primary**: deployed on the primary node. Only the HTAP function is enabled on the primary node. IMCV DDL logs are not replayed on the standby node. During a switchover, IMCV is unloaded when the primary node is demoted to the standby node, and IMCV is reloaded when the standby node is promoted to the primary node.
- **all**: deployed on both the primary and standby nodes. IMCV data on the primary and standby nodes is retained during a switchover.
- **standby**: deployed on the standby node. Only the HTAP function is enabled on the standby node. When IMCV is created on the primary node, only metadata is created, and data is not loaded. During a switchover, IMCV is loaded when the primary node is demoted to the standby node, and IMCV is unloaded when the standby node is promoted to the primary node. Only metadata is created.

Default value: **primary**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Retain the default value. In a primary/standby cluster, set this parameter to the same value on both nodes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

htap_compression_methods

Parameter description: Specifies whether to enable the IMCV column-store compression feature and the list of enabled compression algorithms. This parameter takes effect only when the HTAP feature is enabled.

Parameter type: enumerated type

Unit: none

Value range:

- **none**: The IMCV column-store compression feature is disabled.
- **delta_plus_rle**: The IMCV column-store compression feature is enabled, and the delta_plus_rle algorithm is used.
- **dict**: The IMCV column-store compression feature is enabled, and the dict algorithm is used.
- **lz4**: The IMCV column-store compression feature is enabled, and the LZ4 algorithm is used.
- **all**: The IMCV column-store compression feature is enabled, and the delta_plus_rle, dict, and LZ4 algorithms are used.

Default value: **none**

Setting method: This is a POSTMASTER parameter. Set it based on instructions provided in **Table 6-1**.

Setting suggestion: Use the default value when the memory is sufficient. In an environment with small memory and large data volume, enable IMCV column-store compression to keep as much data as possible in the memory.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.3.55 Reserved Parameters

NOTE

The parameters below are reserved and do not take effect in this version.

acce_min_datasize_per_thread

dfs_partition_directory_length

enable_fstream

enable_hdfs_predicate_pushdown

enable_orc_cache

schedule_splits_threshold

enable_hadoop_env

undo_zone_count

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

walwriter_cpu_bind

walwriteraux_bind_cpu

wal_rec_writer_bind_cpu

wal_receiver_bind_cpu

wal_sender_bind_cpu_attr

autoanalyze

cmserver_self_vote_timeout

transaction_deferrable

character_set_client

character_set_database

collation_database

character_set_server

collation_server

max_que_size

enable_global_plsqlcache

max_execute_functions

max_compile_packages

net_write_timeout

plat_compat_allow_public_key_retrieval

plat_compat_server_port

delimiter_name

enable_parallel_populate

enable_startup_populate

catchup2normal_wait_time

Deprecated Parameters

enable_user_metric_persistent

user_metric_retention_time

enable_verify_active_statements

enable_dywlm_adjust

enable_force_memory_control

streaming_batch_memory

streaming_batch_size

streaming_batch_timeout

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

streaming_collect_memory

streaming_flush_interval

streaming_gather_window_interval

streaming_num_collectors

streaming_num_queues

streaming_num_workers

streaming_router_port

enable_slow_query_log

tde_cmk_id

transparent_encrypted_string

transparent_encrypt_kms_url

transparent_encrypt_kms_region

crypto_module_info

plpgsql_dependency

enable_gs_opt

prefetch_quantity

backwrite_quantity

fast_extend_file_size

enable_adio_debug

wal_flush_delay

query_log_file

query_log_directory

hll_default_regwidth

hll_default_expthresh

hll_default_sparseon

hll_max_sparse

enable_compress_hll

az_switchover_threshold

az_check_and_arbitrate_interval

az_connect_check_interval

az_connect_check_delay_time

enable_segment_remain_cleanup

enable_fast_allocate

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_dummy_time

enable_incremental_catchup

enable_expr_fusion

enable_roach_standby_cluster

6.3.56 Vector Database Parameters

diskann_probe_ncandidates

Parameter description: Specifies the size of the candidate set when the GsDiskANN index is used to retrieve vectors.

Parameter type: integer

Unit: none

Value range: 1 to 32768

Default value: **128**

Setting method: This is a USERSET parameter. Set it based on instructions provided in "Database GUC Parameters > Setting Parameters" in *Reference*.

Setting suggestion:

- Retain the default value. Obtain the optimal parameter settings through experiments.
- The **diskann_probe_ncandidates** parameter can be set and take effect for some queries that use the GsDiskANN index. You are advised to set the parameter only in a session using SET. You are advised not to use gs_guc to set the parameter globally.

Risks and impacts of improper settings: If the value is too large, the query performance deteriorates. If the value is small, the retrieval rate is insufficient.

diskann_hybrid_search_type

Parameter description: Specifies the query policy for vector-scalar hybrid searches. This parameter can be set and take effect for queries in the vector-scalar hybrid search.

NOTE

This parameter takes effect at the session level. You are advised to set the GUC parameter using method 2 in "Database GUC Parameters > Setting Parameters" in *Reference*. Otherwise, the specified policy cannot be selected on DNs.

Parameter type: string

Unit: none

Value range:

- **"AUTO"** (default): The policy is selected based on the values of **diskann_hybrid_pre_filter_selectivity** and **diskann_hybrid_flat_upper_bound**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **"PRE_FILTER_FLAT"**: The policy is to perform filtering and then flat search.
- **"PRE_FILTER_ANN"**: The policy is to perform filtering and then graph search.
- **"POST_FILTER_ANN"**: The policy is to perform graph search and then filtering.

Default value: **"AUTO"**

Setting method: This is a USERSET parameter. Set it based on instructions provided in "Database GUC Parameters > Setting Parameters" in *Reference*.

Setting suggestion: Retain the default value. Obtain the optimal parameter settings through experiments.

Risks and impacts of improper settings: If this parameter is set to an improper value in different data set scenarios, the retrieval rate may be insufficient or the query performance may deteriorate.

diskann_hybrid_pre_filter_selectivity

Parameter description: Specifies the selectivity threshold of the pre-filter policy in the vector-scalar hybrid search. If the selectivity is less than the threshold, pre-filter is performed. If the selectivity is greater than or equal to the threshold, post-filter is performed.

NOTE

- If the value of **diskann_hybrid_search_type** is not the default value, this parameter does not take effect.
- This parameter takes effect at the session level. You are advised to set the GUC parameter using method 2 in "Database GUC Parameters > Setting Parameters" in *Reference*. Otherwise, the specified policy cannot be selected on DNs.

Parameter type: floating-point

Unit: none

Value range: 0 to 1

Default value: **0.15**

Setting method: This is a USERSET parameter. Set it based on instructions provided in "Database GUC Parameters > Setting Parameters" in *Reference*.

Setting suggestion: Retain the default value. Obtain the optimal parameter settings through experiments.

Risks and impacts of improper settings: If this parameter is set to an improper value, the retrieval rate may be insufficient or the query performance may deteriorate.

diskann_hybrid_pullup_selectivity

Parameter description: Specifies the selectivity threshold for moving pre-filter conditions to post-filter when vector-scalar hybrid search is performed in the pre-filter scenario. A smaller value indicates that more pre-filter conditions are moved to post-filter. The default value is **1.0**, indicating that no pre-filter conditions are moved to post-filter.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When conditions are moved to post-filter, each sub-condition connected by AND at the outermost layer is considered as a whole to determine whether to move it. For example, in **A > 5 AND (B > 3 OR C > 3)**, **A > 5** is a whole, and **(B > 3 OR C > 3)** is a whole. **(B > 3 OR C > 3)** is either placed in pre-filter or post-filter, and will not be split.
- The priority of this parameter is lower than that of the **diskann_hybrid_search_type** and **diskann_hybrid_pre_filter_selectivity** parameters. Pre-filter can be moved to post-filter only when **diskann_hybrid_search_type** is set to the default value and the overall selectivity of the filter conditions is less than the value of **diskann_hybrid_pre_filter_selectivity**.
- The algorithm moves as many conditions as possible from pre-filter to post-filter in descending order of selectivity until the following limitations are no longer met: post-filter selectivity > **diskann_hybrid_pullup_selectivity**; post-filter selectivity ≥ **diskann_hybrid_pre_filter_selectivity**.
- This parameter does not take effect when the overall filter selectivity is **0**.
- The behavior of **diskann_hybrid_pullup_selectivity < diskann_hybrid_pre_filter_selectivity** is equivalent to that of **diskann_hybrid_pullup_selectivity = diskann_hybrid_pre_filter_selectivity**.
- The algorithm ensures that not all pre-filter conditions are moved to post-filter.

Parameter type: floating-point

Unit: none

Value range: 0 to 1

Default value: **1**

Setting method: This is a USERSET parameter. Set it based on instructions provided in "Database GUC Parameters > Setting Parameters" in *Reference*.

Setting suggestion: In the pre-filter flat or graph search policy, the overhead of generating a bitmap based on the filter conditions is directly proportional to the sum of the data set size multiplied by the selectivity of each filter condition. Therefore, if the sum of the data set size multiplied by the selectivity is too large, causing the query delay of the pre-filter policy to be excessively high, you can decrease the value of this parameter to move some pre-filter conditions to post-filter, reducing the overhead of bitmap generation.

Risks and impacts of improper settings: This parameter setting will cause the execution plan to change. If it is set improperly, the query may not use the optimal execution plan.

diskann_hybrid_flat_upper_bound

Parameter description: Specifies the maximum data volume threshold of the flat search policy before performing the vector-scalar hybrid search in pre-filter scenarios. If the data volume is less than the threshold, flat search is performed. If the data volume is greater than the threshold, vector search is performed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If this parameter is set to **0**, the value **8000** takes effect in the Arm64 architecture, and the value **11000** takes effect in other architectures.
- If the value of **diskann_hybrid_search_type** is not the default value, this parameter does not take effect.
- This parameter takes effect at the session level. You are advised to set the GUC parameter using method 2 in "Database GUC Parameters > Setting Parameters" in *Reference*. Otherwise, the specified policy cannot be selected on DNs.

Parameter type: integer

Unit: none

Value range: 0 to 1000000

Default value: **0**

Setting method: This is a USERSET parameter. Set it based on instructions provided in "Database GUC Parameters > Setting Parameters" in *Reference*.

Setting suggestion: Retain the default value. Obtain the optimal parameter settings through experiments.

Risks and impacts of improper settings: If this parameter is set to an improper value, the retrieval rate may be insufficient or the query performance may deteriorate.

diskann_hybrid_neighbor_search_prune_factor

Parameter description: Specifies the pruning factor for neighbors that do not meet the query conditions during the search process in the scenario where the filtering-before-ANN policy is used for vector-scalar hybrid indexing. The value **0** indicates no pruning, and the value **1** indicates full pruning. A larger value indicates a higher pruning ratio.

Parameter type: floating-point

Unit: none

Value range: 0 to 1

Default value: **0**.

Setting method: This is a USERSET parameter. Set it based on instructions provided in "Database GUC Parameters > Setting Parameters" in *Reference*.

Setting suggestion:

- The value **0.3** is recommended. In most scenarios, a larger pruning factor improves query performance but reduces the retrieval rate. Conversely, a smaller pruning factor improves the retrieval rate but reduces query performance. It is recommended that you obtain the optimal parameter configuration through experiments.
- This parameter takes effect at the session level. You are advised to set GUC parameters using method 2 in "Database GUC Parameters > Setting Parameters" in *Reference*. Otherwise, the specified filtering policy cannot be used on DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If this parameter is set to a large value, the retrieval rate may be insufficient.

diskann_hybrid_enable_lazy_flip_bitmap

Parameter description: Specifies whether to enable the lazy flip bitmap feature. In the case of a scalar query or IS NOT NULL query condition with a selectivity greater than 50% under the pre-filter flat or graph search policy, the lazy flip bitmap calculates the complement of the query to reduce the overhead of inserting data into the bitmap.

Parameter type: Boolean

Unit: none

Value range:

- **on**: enabled.
- **off**: disabled.

Default value: **on**

Setting method: This is a USERSET parameter. Set it based on instructions provided in "Database GUC Parameters > Setting Parameters" in *Reference*.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gsivfflat_probes

Parameter description: Specifies the number of inverted tables to be searched when the GsIVFFLAT index is used to retrieve vectors. If this value exceeds the total number of inverted tables of the GsIVFFLAT index, the entire table is searched. The total number of inverted tables of the GsIVFFLAT index is specified by the **ivf_nlist** parameter during index build.

Parameter type: integer

Unit: none

Value range: 1 to 32768

Default value: **5**

Setting method: This is a USERSET parameter. Set it based on instructions provided in "Database GUC Parameters > Setting Parameters" in *Reference*.

Setting suggestion: Set this parameter to 3% of the value of **ivf_nlist** during index build. You are advised to obtain the optimal parameter configuration through experiments.

Risks and impacts of improper settings: If this parameter is set to a larger value, the search takes a longer time but the search result is more accurate.

gsivfflat_secondary_probes

Parameter description: Specifies the number of secondary inverted tables to be searched when the GsIVFFLAT index is used to retrieve vectors. If this value

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

exceeds the total number of secondary inverted tables, the entire table is searched. The total number of secondary inverted tables is specified by the **ivf_nlist2** parameter during index build.

Parameter type: integer

Unit: none

Value range: 1 to 32768

Default value: **5**

Setting method: This is a USERSET parameter. Set it based on instructions provided in "Database GUC Parameters > Setting Parameters" in *Reference*.

Setting suggestion: You are advised to set this parameter to a value between 1/4 and 1/2 of the value of **ivf_nlist2** during index build. You are advised to obtain the optimal parameter configuration through experiments.

Risks and impacts of improper settings: If this parameter is set to a larger value, the search takes a longer time but the search result is more accurate.

NOTE

gsivfflat_secondary_probes has the same effect as **gsivfflat_probes**. It takes effect only when the vector index is a double-layer index, which effectively accelerates the query speed. You are advised to obtain the optimal parameter configuration through experiments.

enable_vectordb

Parameter description: Specifies whether vector indexes can be created and whether vector indexes can be added, modified, and queried.

Parameter type: Boolean

Unit: none

Value range:

- **on**: allowed.
- **off**: not allowed.

Default value: **off**

Setting method: This is a SIGHUP parameter. Set it based on instructions provided in "Database GUC Parameters > Setting Parameters" in *Reference*.

Setting suggestion: If users are not allowed to use vector database features such as vector index, set this parameter to **off**. Otherwise, set this parameter to **on**.

Risks and impacts of improper settings: The parameter value determines whether users can use the vector database features. If the parameter value is incorrect, users may be unable to use the intended database features.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 6-53 GUC parameters related to the vector database

GUC Parameter	Level	Value Range (Default Value)	Description
maintenance_work_mem	Session	[1024,2147483647]/[1MB,2048GB) (65536/64MB)	Maximum amount of memory used in maintenance operations. The default unit is KB.
diskann_probe_ncandidates	Session	[1, 32768](128)	Size of the candidate set when the GsDiskANN index is used to retrieve vectors.
gsivfflat_probes	Session	[1, 32768](5)	Number of inverted tables to be searched. If it exceeds the total number of inverted tables, the entire table is searched.
gsivfflat_secondary_probes	Session	[1, 32768](5)	Number of secondary inverted tables to be searched. If it exceeds the total number of secondary inverted tables, the entire table is searched.
enable_vectordb	SIGHUP	[off,on](off)	An advanced feature that specifies whether vector indexes can be created and whether vector indexes can be added, modified, and queried.

6.4 CM Parameters

Modifying CM parameters affects the running mechanism of GaussDB. You are advised to ask GaussDB engineers to do it for you. For details about how to modify CM parameters, see method 1 in **Table 6-2** or the corresponding method in **Table 6-3**. All CM parameters can be set using methods in **Table 6-3**. You are advised to use the cm_ctl tool.

You can view CM Agent parameters in the **cm_agent.conf** file in the CM Agent data directory and CM Server parameters in the **cm_server.conf** file in the CM Server data directory.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

6.4.1 CM Agent Parameters

log_dir

Parameter description: Specifies the directory where CM Agent logs are stored. The value can be an absolute path, or a path relative to the CM Agent data directory.

Parameter type: string

Unit: none

Value range: a valid directory.

Default value: "*$GAUSSLOG*/cm/cm_agent", indicating that the CM Agent logs are generated in the CM directory in *$GAUSSLOG*.

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-2**. The modification takes effect only after the cluster is restarted.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_file_size

Parameter description: Specifies the size of a log file. If a log file exceeds the specified size, a new one is created to record logs.

Parameter type: integer

Unit: MB

Value range: 0 to 2047

Default value: **16**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**. For example, if this parameter is set to **16** without a unit, it indicates 16 MB. If this parameter is set to **16MB**, it indicates 16 MB. The unit must be MB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_min_messages

Parameter description: Specifies which message levels are written to the CM Agent log. A higher level covers the messages of all the lower levels. The lower the level is, the fewer messages will be written into the log.

Value type: enumerated

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: **debug5**, **debug1**, **log**, **warning**, **error**, and **fatal**.

Default value: **warning**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Set this parameter based on the hard disk size.

Risks and impacts of improper settings: If a low level is set, debugging information may be lost.

log_threshold_check_interval

Parameter description: Specifies the interval for compressing and clearing logs.

Parameter type: integer

Unit: seconds (s)

Value range: 0 to 2147483647

Default value: **1800**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value. You can adjust the value based on the disk space and scenario requirements.

Risks and impacts of improper settings: If this parameter is set to a large value, too much disk space may be occupied.

log_max_size

Parameter description: Specifies the maximum size of a log file.

Parameter type: integer

Unit: MB

Value range: 0 to 2147483647

Default value: **10240**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value. You can adjust the value based on the disk space and scenario requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_max_count

Parameter description: Specifies the maximum number of logs that can be stored on hard disks.

Parameter type: integer

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: none

Value range: 0 to 2147483647

Default value: **10000**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value. You can adjust the value based on the disk space and scenario requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_saved_days

Parameter description: Specifies the number of days for storing logs.

Parameter type: integer

Unit: day

Value range: 0 to 1000

Default value: **90**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value. You can adjust the value based on the disk space and scenario requirements.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_log_compress

Parameter description: Specifies whether to enable the log compression function.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The log compression function is enabled.
- **off**: The log compression function is disabled.

Default value: **on**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

incremental_build

Parameter description: Specifies whether a standby DN is incrementally built.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Standby DNs are incrementally rebuilt.
- **off**: Standby DNs are fully rebuilt.

Default value: **on**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

alarm_component

Parameter description: Specifies the location of the alarm component that processes alarms.

Parameter type: string

Unit: none

Value range: a valid file path.

- If **--alarm-type** in the gs_preinstall script is set to **5**, no third-party component is connected and alarms are written into system_alarm logs. In this case, the value of **alarm_component** is **/opt/huawei/snas/bin/snas_cm_cmd**.
- If **--alarm-type** in the gs_preinstall script is set to **1**, a third-party component is connected. In this case, the value of **alarm_component** is the absolute path of the executable program of the third-party component.

Default value: **"/opt/huawei/snas/bin/snas_cm_cmd"**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

alarm_report_interval

Parameter description: Specifies the interval at which an alarm is reported.

Parameter type: integer

Unit: seconds

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 1 to 2147483647

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, alarms may not be reported in a timely manner. If the value is too small, the CPU load may increase.

alarm_report_max_count

Parameter description: Specifies the maximum number of times an alarm is reported.

Parameter type: integer

Unit: none

Value range: 1 to 2592000

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the CPU load may increase.

agent_report_interval

Parameter description: Specifies the interval at which CM Agent reports the instance status.

Parameter type: integer

Unit: seconds

Value range: 1 to 2147483647

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, CM fault detection and recovery cannot be performed in a timely manner.

agent_phony_dead_check_interval

Parameter description: Specifies the interval at which CM Agent checks whether the CN, DN, or GTM process is suspended.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer

Unit: seconds

Value range: 0 to 2147483647

Default value: **10**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, CM fault detection and recovery cannot be performed in a timely manner. If the value is too small, the CPU load increases.

agent_check_interval

Parameter description: Specifies the interval at which the CM Agent queries the status of instances, such as the DNs, CN, and GTM.

Parameter type: integer

Unit: seconds

Value range: 1 to 2147483647

Default value: 2

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, CM fault detection and recovery cannot be performed in a timely manner. If the value is too small, the CPU load increases.

agent_heartbeat_timeout

Parameter description: Specifies the timeout interval for CM Agent to detect the CM Server heartbeat. If CM Agent does not receive the heartbeat message from CM Server within the timeout interval, the CM Agent terminates the connection with CM Server.

Parameter type: integer

Unit: seconds

Value range: 2 to 2147483647

Default value: **8**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

agent_connect_timeout

Parameter description: Specifies the time to wait before the attempt of CM Agent to connect to CM Server times out.

Parameter type: integer

Unit: seconds

Value range: 0 to 2147483647

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

agent_connect_retries

Parameter description: Specifies the number of times CM Agent retries to connect to the CM Server.

Parameter type: integer

Unit: none

Value range: 0 to 2147483647

Default value: **15**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

agent_conn_db_timeout

Parameter description: Specifies the connection timeout interval for checking instance status. The CM Agent monitors the status of CN and DN instances; if no connection is established within this timeframe, it is considered a connection failure.

Parameter type: integer

Unit: seconds

Value range: 1 to 2147483647

Default value: 2

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

agent_conn_db_phony_dead_timeout

Parameter description: Specifies the connection timeout interval for checking zombie instances. The CM Agent checks for zombie CN and DN instances; if no connection is established within this timeframe, it is considered a connection failure.

Parameter type: integer

Unit: seconds

Value range: 1 to 2147483647

Default value: **5**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

agent_status_check_connection_type

Parameter description: Specifies the type of the connection used for the CM Agent to check node status.

Parameter type: integer

Unit: none

Value range:

- **0**: libpq by default and UDS in case of communication failure through libpq in high load state. When UDS connections are used for communication, the reserved memory status is queried.
- **1**: libpq only. The logic is the same as that in 506.0.0 and earlier versions.
- **2**: UDS only.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

agent_kill_instance_timeout

Parameter description: Specifies the interval from the time when CM Agent fails to connect to the primary CM Server to the time when CM Agent terminates all instances on the node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer

Unit: seconds

Value range: 0 to 2147483647

Default value: **0**, indicating that the operation of stopping all instances on the node is not initiated.

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: If you want to set this parameter to a value other than 0, ensure that the value is greater than the time required for promoting the CM Server to primary.

Risks and impacts of improper settings: If the value is too large, CM faults may not be rectified in a timely manner.

agent_backup_open

Parameter description: Specifies whether the CM runs in the standby cluster for DR.

Parameter type: integer

Unit: none

Value range: 0 to 3

- **0**: No DR relationship is established.
- **1**: The OceanStor Dorado-based DR relationship is established.
- **2**: The streaming replication-based DR relationship is established.
- **3**: The logical DR relationship is established.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-2**. The modification takes effect only after the cluster is restarted.

Setting suggestion: This parameter is internally called. You are advised not to modify it.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

enable_gtm_phony_dead_check

Parameter description: Specifies whether to enable the GTM zombie check function.

Parameter type: integer

Unit: none

Value range: 0 to 1

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **0**: The GTM zombie check function is disabled.
- **1**: The GTM zombie check function is enabled.

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

security_mode

Parameter description: Specifies whether CNs and DNs are started in secure mode.

Parameter type: Boolean

Unit: none

Value range:

- **on**: CNs and DNs are started in secure mode.
- **off**: CNs and DNs are started in non-secure mode.

Default value: **off**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

upgrade_from

Parameter description: Specifies the internal version number of the cluster before an in-place upgrade. Do not modify the value of this parameter.

Parameter type: integer

Unit: none

Value range: 0 to 4294967295

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Manual modification may cause in-place upgrade failures.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

process_cpu_affinity

Parameter description: Specifies whether to bind a primary DN process to a CPU core before starting the process. Only Arm is supported.

Parameter type: integer

Unit: none

Value range: 0 to 2

- If the value is **0**, core binding is not optimized.
- If the value is greater than 0, core binding is optimized and the number of physical CPU slices is 2^n.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-2**. The modification takes effect only after the cluster and CM Agent are restarted.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

dilatation_shard_count_for_disk_capacity_alarm

Parameter description: Specifies the number of shards to be added in the scale-out scenario. This parameter is used to calculate the threshold for reporting a disk capacity alarm.

Parameter type: integer

Unit: none

Value range: 0 to 2147483647

- If the value is **0**, the disk scale-out alarm reporting is disabled.
- If the value is greater than 0, disk scale-out alarm reporting is enabled, and the alarm reporting threshold is calculated based on the number of shards specified by this parameter.

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Set the number of shards to be the same as the actual number of shards to be added.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_cn_auto_repair

Parameter description: Specifies whether to enable the automatic CN repair function.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The automatic CN recovery is enabled. That is, after a CN is removed, the agent automatically attempts to recover the CN and resume the CN.
- **off**: The automatic CN recovery is disabled.

Default value: **on**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Automatic CN recovery is enabled by default. If it is disabled, CNs will not be automatically resumed. If you want to adjust the value, make sure you understand the parameter meaning and adjust the value with caution to avoid risks caused by misoperations.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_xc_maintenance_mode

Parameter description: Specifies whether the pgxc_node system catalog can be modified when the cluster is in read-only mode.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The pgxc_node system catalog can be modified.
- **off**: The pgxc_node system catalog cannot be modified.

Default value: **on**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-2**. The modification takes effect only after the cluster is restarted.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

unix_socket_directory

Parameter description: Specifies the directory location of the Unix socket.

Parameter type: string.

Unit: none

Value range: a valid directory.

Default value: ""

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_dcf

Parameter description: Specifies the status of the DCF mode.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The DCF mode is enabled.
- **off**: The DCF mode is disabled.

Default value: **off**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_e2e_rto

Parameter description: Specifies whether to enable the E2E RTO function. After this function is enabled, the zombie detection period and network detection timeout interval are shortened. The CM can reach the E2E RTO value (RTO for a single instance: ≤ 10s; RTO for overlay faults: ≤ 30s).

Parameter type: integer

Unit: none

Value range: 0 to 1

- **0**: The E2E RTO function is disabled.
- **1**: The E2E RTO function is enabled.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value. You can adjust the value as required.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

environment_threshold

Parameter description: Specifies the thresholds for the physical environment and node status monitored by the agent. If the thresholds are exceeded, logs will be printed. The thresholds include the memory usage threshold, CPU usage threshold, disk usage threshold, instance memory usage threshold, and instance thread pool usage threshold.

Parameter type: string

Unit: none

Value range: This parameter consists of five parts: memory usage threshold, CPU usage threshold, disk usage threshold, instance memory usage threshold, and instance thread pool usage threshold. The value range of each threshold is 0 to 100, in percentage (%). The value **0** indicates that the detection is disabled. Different thresholds are separated by commas (,). The value must be enclosed with parentheses. For example, **"(0,0,0,0,0)"** indicates that none of the five thresholds is detected.

Default value: **"(0,0,0,0,0)"**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

thread_effective_time

Parameter description: Specifies the timeout interval for a round of thread running. If the timeout occurs, the thread is suspended and automatically exits.

Parameter type: integer

Unit: seconds

Value range: 0 to 1200. The value **0** indicates that the thread monitoring capability is disabled.

Default value: **60**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: You are advised not to change the parameter value. If you need to change the parameter value, contact Huawei technical support.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

ping_fault_kill_inst_timeout

Parameter description: Specifies the timeout interval for restarting a primary DN after the primary DN cannot ping all standby DNs.

Parameter type: integer

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Unit: seconds

Value range: 0 to 1000. The value **0** indicates that the ping fault processing capability is disabled.

Default value: **30**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

high_cpu_threshold

Parameter description: Specifies the CPU usage threshold. When the CPU usage exceeds the threshold, the CM Agent prints CPU usage information in logs at a specified check interval (1s by default).

Parameter type: integer

Unit: %

Value range: 0 to 100

Default value: **95**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

high_mem_threshold

Parameter description: Specifies the memory usage threshold. When the memory usage exceeds the threshold, the CM Agent prints memory usage information in logs at a specified check interval (1s by default).

Parameter type: integer

Unit: %

Value range: 0 to 100

Default value: **95**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

high_disk_threshold

Parameter description: Specifies the disk I/O threshold. When the I/O exceeds the threshold, the CM Agent prints I/O information in logs at a specified check interval (1s by default). The iostat -xm information is recorded in the **system_stat** log file for 10 consecutive times at an interval of 1s.

Parameter type: integer

Unit: %

Value range: 0 to 100

Default value: **95**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

high_load_io_rw_times

Parameter description: Determines the number of I/O read and write times in high load state. When the number of database I/O read and write times reaches the threshold, the database is considered to be in the high load state and does not rectify issues like disk hang or D-state.

Parameter type: integer

Unit: times/second

Value range: 0 to 2147483647. The value **0** indicates that no judgment is made on the number of I/O read/write times.

Default value: **10**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

high_load_io_traffic

Parameter description: Determines the I/O traffic in high load state. When the database I/O traffic volume reaches the threshold, the database is considered to be in the high-load state and does not rectify issues like disk hang or D-state.

Parameter type: integer

Unit: KB/s

Value range: 0 to 2147483647. The value **0** indicates that no judgment is made on the I/O read and write traffic.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **256**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

sys_check_interval

Parameter description: Specifies the system resource check period, that is, the interval for checking the CPU, memory, and disk I/O usage. The value **0** indicates that the check is disabled.

Parameter type: integer

Unit: seconds

Value range: 0 to 10

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, alarms may not be reported in a timely manner.

network_check_concurrency

Parameter description: Specifies the number of concurrent network checks, that is, the number of network links to be checked at the same time. The value **0** indicates that the check is disabled.

Parameter type: integer

Unit: none

Value range: 0 to 1000

Default value: **10**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, CM fault detection and recovery cannot be performed in a timely manner. If the value is too large, the CPU load increases.

network_check_interval

Parameter description: Specifies the network check interval. The value **0** indicates that the check is disabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer

Unit: seconds

Value range: 0 to 1000

Default value: **5**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, CM fault detection and recovery cannot be performed in a timely manner. If the value is too small, the CPU load increases.

asyn_log_enable

Parameter description: Specifies whether to enable asynchronous logging. After this parameter is enabled, the CM Agent starts asynchronous logging.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The CM Agent starts asynchronous logging.
- **off**: The CM Agent does not start asynchronous logging but uses serial logging.

Default value: **on**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, logs may be lost.

asyn_log_max_buf_size

Parameter description: Specifies the size of the asynchronous log buffer. When asynchronous logs are stacked and the stacked logs exceed the buffer size, subsequent logs cannot be recorded.

Parameter type: integer

Unit: KB

Value range: 1 to 1048576

Default value: **1024**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Adjust the value based on the memory and hard disk size.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

asyn_log_fault_solution

Parameter description: Specifies the asynchronous log processing solution after a log disk fault is identified. When the CM detects that a log disk is faulty or the asynchronous log buffer overflows, the CM processes asynchronous logs based on the configuration.

Parameter type: integer

Unit: none

Value range: 0 to 2

- **0**: No extra processing is performed on logs. All logs are retained in the buffer until the buffer size reaches the upper limit. After the buffer size reaches the upper limit, no more logs are recorded.
- **1**: Logs are filtered. After a log disk fault is identified, only key logs, alarms, and logs at the ERROR level or higher are saved to the buffer until the buffer size reaches the upper limit. After the buffer size reaches the upper limit, no more logs are recorded.
- **2**: Logs are discarded. After a log disk fault is identified, only key logs and alarms are saved to the buffer until the buffer size reaches the upper limit. After the buffer size reaches the upper limit, no more logs are recorded.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Adjust the value based on the memory and hard disk size.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

disk_check_hung_timeout

Parameter description: Specifies the timeout interval for a disk I/O hang. When a single disk I/O check exceeds the timeout interval, the disk I/O hang occurs. The value **0** indicates that the disk hang check is disabled.

Parameter type: integer

Unit: seconds

Value range: 0 to 1000

Default value: **20**. If you cannot determine whether the detected disk is striped during installation, the default value is **60**.

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

disk_check_block_timeout

Parameter description: If file operations such as open are blocked due to certain reasons in the file system and the blocking duration exceeds the value of this parameter, the system determines that the disk is faulty. If this parameter is set to **0**, the determination is disabled.

Parameter type: integer

Unit: seconds

Value range: 0 to 1000

Default value: **40**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

disk_check_interval

Parameter description: Specifies the traffic of disk I/O hang check, that is, the minimum duration within which specified data needs to be written. The default check interval is 5 seconds. The value **0** indicates that the disk check is disabled.

Parameter type: integer

Unit: seconds

Value range: 0 to 1000

Default value: **5**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Improper settings may cause inaccurate fault detection and delayed fault rectification.

disk_check_io_size

Parameter description: Specifies the disk I/O hang check traffic, that is, the size of data written to the disk within the check duration. This is an I/O operation directly performed on the disk. By default, 512-byte data is written. If the detected disk is striped, the default value is changed to 262144 bytes (256 KB) during installation. A maximum of 1 GB data can be written. The value **0** indicates that the disk hang check is disabled. Disk detection has a whitelist, including the CM/CN/DN/GTM data directory, disk where the log directory resides, and disk where the user-defined tablespace directory resides.

Parameter type: integer

Unit: byte

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 0 to 1073741824

Default value: **512**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Improper settings may cause inaccurate fault detection and delayed fault rectification.

disk_check_shared_storage_hung_timeout

Parameter description: Specifies the timeout interval for detecting I/O hang of the OceanStor Dorado shared disk. If the I/O hang detection time of the OceanStor Dorado shared disk exceeds the value of this parameter, the system determines that the OceanStor Dorado shared disk has I/O hang.

Parameter type: integer

Unit: seconds

Value range: 0 to 1000. The value **0** indicates that the I/O hang detection function of the OceanStor Dorado shared disk is disabled.

Default value: **30**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is incorrectly set, the system may incorrectly determine that the OceanStor Dorado shared disk has I/O hang.

process_unhealthy_alarm_time

Parameter description: Specifies the detection timeout interval for triggering the process status alarm. When a process is in an abnormal state (**D**, **T**, or **Z**) for a long time exceeding the value of this parameter, a process status alarm is triggered.

Parameter type: integer

Unit: seconds

Value range: 0 to 2147483647. The value **0** indicates that this function is disabled.

Default value: **50**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

enable_tblspc_disk_check

Parameter description: Specifies whether to enable the tablespace disk check function.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The tablespace disk check is enabled. That is, the disk where the tablespace is located is hung and the corresponding DN is stopped.
- **off**: The tablespace disk check function is disabled.

Default value: **on**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_network_package_loss

Parameter description: Checks the packet loss rate between the primary and standby DNs on the data plane. If the packet loss rate of the primary DN exceeds the threshold, a restart is triggered.

Parameter type: integer

Unit: %

Value range: 0 to 99. The value **0** indicates that this function is disabled.

Default value: **30**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, faults may not be rectified in a timely manner.

enable_auto_start

Parameter description: Specifies whether CM Agent automatically starts DNs and CNs.

Parameter type: integer

Unit: none

Value range: 0 to 1

- **0**: The automatic startup function is disabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **1**: The automatic startup function is enabled.

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_check_az_network

Parameter description: Specifies whether to enable AZ-level data plane network detection.

Parameter type: Boolean

Unit: none

Value range:

- **on**: AZ-level data plane network detection is enabled.
- **off**: AZ-level data plane network detection is disabled.

Default value: **on**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, the fault detection function deteriorates.

expect_stop_az_name

Parameter description: Specifies the name of the AZ to be stopped by the AZ-level network detection function.

Parameter type: string

Unit: none

Value range: name of an existing AZ in the environment. If this parameter is left empty, instances in the AZ opposite to the primary CM Server are stopped.

Default value: **""**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

check_az_network_concurrency

Parameter description: Specifies the number of concurrent AZ-level network detection tasks.

Parameter type: integer

Unit: none

Value range: 1 to 10

Default value: **2**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, CM fault detection and recovery cannot be performed in a timely manner. If the value is too large, the CPU load increases.

cm_hadr_process_type

Parameter description: Specifies the DR process ID.

Value type: enumerated

Unit: none

Value range:

- **none**: There is no process.
- **failover**: A standby cluster for DR is promoted to primary.
- **switchover_promote**: A standby cluster for DR is promoted to primary during the cluster switchover.
- **switchover_demote**: A primary cluster is demoted to a standby cluster for DR during the cluster switchover.

Default value: **none**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

emergence_sql_check_max_cn_count

Parameter description: Specifies the maximum number of CNs in a cluster allowed in a first aid. If the number of CNs exceeds the value of this parameter, no SQL statement detection is performed. If this parameter is set to **0**, the CN detection is disabled.

Parameter type: integer

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 0 to 256

Default value: **16**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_topology_network_gtm_gtm_collect

Parameter description: Controls whether to enable network topology detection between GTMs.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Network detection is enabled.
- **off**: Network detection is disabled.

Default value: **off**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 or method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

topology_network_collecting_group_size

Parameter description: Specifies the number of DN shards collected by each collector thread on the data plane network topology. This parameter can be used only in quorum mode.

Parameter type: integer

Unit: none

Value range: 8 to 32

Default value: **16**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

topology_network_collecting_failed_times

Parameter description: Specifies the number of consecutive network disconnections on the data plane. If the number of disconnections exceeds the value of this parameter, the data plane network topology is abnormal. This parameter can be used only in quorum mode.

The value must be the same as that of **topology_network_collecting_failed_times** in the CM Server parameters.

Parameter type: integer

Unit: none

Value range: 3 to 50

Default value: **5**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

topology_network_collecting_interval

Parameter description: Specifies the data plane network topology detection period. If this parameter is set to **0**, network topology detection is disabled. This parameter can be used only in quorum mode.

Parameter type: integer

Unit: seconds

Value range: 0 to 600

Default value: 2

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

topology_network_check_max_cn_count

Parameter description: Specifies the maximum number of CNs in a cluster supported by data plane network topology detection. If the number of CNs exceeds the value of this parameter, the data plane network topology detection is not performed. If this parameter is set to **0**, network topology detection is disabled. This parameter can be used only in quorum mode.

Parameter type: integer

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 0 to 256

Default value: **16**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_cma_check_inst_timeout

Parameter description: Specifies the timeout interval for the CM Agent to check instances each round. When the timeout interval expires, the AIM_AI_CheckInstanceTooLong alarm is reported. If this parameter is set to **0**, the alarm is not reported.

Parameter type: integer

Unit: seconds

Value range: 0 to 1000

Default value: **20**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, alarms might be frequently reported.

enable_gsmonitor

Parameter description: Specifies whether to enable the GSMonitor system resource statistics tool.

Parameter type: Boolean

Unit: none

Value range:

- **on**: GSMonitor is enabled.
- **off**: GSMonitor is disabled.

Default value: **on**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

gsmonitor_max_store_duration

Parameter description: Specifies the maximum number of days for storing GSMonitor logs.

Parameter type: integer

Unit: day

Value range: 2 to 30

Default value: **14**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gsmonitor_max_store_space

Parameter description: Specifies the maximum space for storing GSMonitor logs.

Parameter type: integer

Unit: MB

Value range: 512 to 10240

Default value: **2048**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

gsmonitor_interval

Parameter description: Specifies the GSMonitor profiling interval.

Parameter type: string

Unit: none

Value range:

- "interval1,interval2,interval3,interval4"

  This parameter is divided into four parts, formatted as "*interval1*,*interval2*,*interval3*,*interval4*".

  interval1 indicates the collection interval when the CPU usage ranges from 0% to 49%. The value ranges from 2 to 2147483647, in seconds.

  interval2 indicates the collection interval when the CPU usage ranges from 50% to 69%. The value ranges from 2 to 2147483647, in seconds.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

interval3 indicates the collection interval when the CPU usage ranges from 70% to 89%. The value ranges from 2 to 2147483647, in seconds.

interval4 indicates the collection interval when the CPU usage ranges from 90% to 100%. The value ranges from 2 to 2147483647, in seconds.

- **"auto"**: equivalent to **"30,10,5,2"**

Default value: **"30,10,5,2"**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

hadr_replconninfo_config_mode

Parameter description: Specifies the DR type of the current database instance.

Parameter type: string.

Unit: none

Value range: **"local_file"** or **"cluster_manager"**

Default value: **"local_file"**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: This parameter is internally called. You are advised not to modify it.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

default_build_mode

Parameter description: Specifies the mode used by CM Agent to automatically initiate primary/standby instance rebuild.

Value type: enumerated

Unit: none

Value range:

- **auto**: indicates that incremental rebuild is performed three times during the rebuild of the primary and standby nodes. If all the three incremental rebuild attempts fail and the failure is not caused by the primary node searching failure, the full rebuild mode is used.
- **incremental**: indicates that the incremental rebuild mode is used to rebuild the primary and standby nodes.
- **full**: indicates that the full rebuild mode is used to rebuild the primary and standby nodes.

Default value: **auto**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value. If you need to change the value, contact Huawei technical support.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_preparse

Parameter description: Specifies whether to check DN pre-parsing. After this function is enabled, the DN pre-parsing result is checked and used as the basis for DN arbitration.

Parameter type: integer.

Unit: none

Value range:

- **0**: disabled.
- **1**: enabled.

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value. If you need to change the value, contact Huawei technical support.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

6.4.2 CM Server Parameters

log_dir

Parameter description: Specifies the directory where CM Server logs are stored. The value can be an absolute path, or a path relative to the CM Server data directory.

Parameter type: string.

Unit: none

Value range: valid path of a directory.

Default value: "*$GAUSSLOG*/cm/cm_server", indicating that the CM Server logs are generated in the CM directory in *$GAUSSLOG*.

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-2**. The modification takes effect only after the cluster is restarted.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_file_size

Parameter description: Specifies the size of a log file. If a log file exceeds the specified size, a new one is created to record logs.

Parameter type: integer.

Unit: MB

Value range: 0 to 2047

Default value: **16**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**. For example, if this parameter is set to **16** without a unit, it indicates 16 MB. If this parameter is set to **16MB**, it indicates 16 MB. The unit must be MB if required.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

log_min_messages

Parameter description: Specifies the level of messages to be written to the CM Server log. Each level covers all the levels following it. The lower the level is, the fewer messages will be written into the log.

Parameter type: enumerated type

Unit: none

Value range: **debug5**, **debug1**, **log**, **warning**, **error**, and **fatal**.

Default value: **warning**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Set this parameter based on the hard disk size.

Risks and impacts of improper settings: If a low level is set, debugging information may be lost.

thread_count

Parameter description: Specifies the number of threads in the agent thread pool.

Parameter type: string.

Unit: none

Value range: This parameter consists of two parts: '**worker_thread_count** and **io_ratio**. The meanings of the two parts are as follows:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **worker_thread_count**: specifies the number of AgentWorker threads. The value ranges from 2 to 1000.
- **io_ratio**: specifies the ratio of I/O threads to worker threads. The value *n* indicates that one AgentIO thread corresponds to *n* AgentWorker threads. The value ranges from 1 to 100. At least one AgentIO thread is required.

NOTE

If the value of **worker_thread_count**, namely the number of AgentWorker threads, is greater than the sum of the number of cluster nodes and the number of threads for processing cm_ctl requests (if the number of cluster nodes is less than 32, one thread is used by default; otherwise, four threads are used), the value that takes effect is the sum of the number of cluster nodes and the number of threads for processing cm_ctl requests.

Default value: **"(1000,1)"**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-2**. The modification takes effect only after the cluster is restarted.

Setting suggestion: Retain the default value. If you want to adjust the value, make sure you understand the parameter meaning and adjust the value with caution to avoid risks caused by misoperations.

Risks and impacts of improper settings: If this parameter is incorrectly set, CM faults may not be detected and rectified in a timely manner, and the CPU load may increase.

thread_effective_time

Parameter description: Specifies the timeout interval for a round of thread running. If the timeout occurs, the thread is suspended and the process automatically exits.

Parameter type: integer.

Unit: seconds

Value range: 0 to 1200. The value **0** indicates that the thread monitoring capability is disabled.

Default value: **20**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-3**.

Setting suggestion: You are advised not to change the parameter value. If you need to change the parameter value, contact Huawei technical support.

Risks and impacts of improper settings: Improper settings may cause inaccurate fault detection and delayed fault rectification.

ctl_thread_count

Parameter description: Specifies the number of threads in the ctl thread pool.

Parameter type: string.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

The value is a string of more than 0 characters. This parameter consists of two parts: **worker_thread_count** and **io_ratio**. The meanings of the two parts are as follows:

- **worker_thread_count**: specifies the number of CtlWorker threads. The value ranges from 0 to 1000. If the value is **0**, the number of CtlWorker threads is equal to the number of nodes in the cluster.
- **io_ratio**: specifies the ratio of I/O threads to worker threads. The value *n* indicates that one CtlIO thread corresponds to *n* CtlWorker threads. The value ranges from 1 to 100. At least one CtlIO thread is required.

Default value: **"(0,1)"**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: CM performance may deteriorate.

alarm_component

Parameter description: Specifies the location of the alarm component that processes alarms.

Parameter type: string.

Unit: none

Value range: valid file path.

- If **--alarm-type** in the gs_preinstall script is set to **5**, no third-party component is connected and alarms are written into system_alarm logs. In this case, the value of **alarm_component** is **/opt/huawei/snas/bin/snas_cm_cmd**.
- If **--alarm-type** in the gs_preinstall script is set to **1**, a third-party component is connected. In this case, the value of the GUC parameter **alarm_component** is the absolute path of the executable program of the third-party component.

Default value: **"/opt/huawei/snas/bin/snas_cm_cmd"**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

alarm_report_interval

Parameter description: Specifies the interval at which an alarm is reported.

Parameter type: integer.

Unit: seconds

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 0 to 2147483647

Default value: **3**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, alarms may not be reported in a timely manner. If the value is too small, the CPU load may increase.

alarm_report_max_count

Parameter description: Specifies the maximum number of times an alarm is reported.

Parameter type: integer.

Unit: none

Value range: 1 to 2592000

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the CPU load may increase.

instance_failover_delay_timeout

Parameter description: Specifies the delay in CM Server failover after the primary CM Server breakdown is detected. It takes effect for GTM arbitration.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, database faults may not be rectified in a timely manner.

instance_heartbeat_timeout

Parameter description: Specifies the timeout interval for CM Server to detect the instance heartbeat.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: seconds

Value range: 1 to 2147483647

Default value: **6**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

instance_keep_heartbeat_timeout

Parameter description: The CM Agent periodically checks the instance status and reports the status to the CM Server. If the instance status cannot be detected for a long time and the accumulated time exceeds the value of this parameter, the CM Server delivers a command to the CM Agent to restart the instance.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647

Default value: **40**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

instance_phony_dead_restart_interval

Parameter description: Specifies the interval at which the CM Agent process restarts and kills a zombie CN, DN, or GTM instance. The interval between two consecutive kill operations cannot be less than the value of this parameter. Otherwise, the CM Agent process does not deliver commands.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

coordinator_heartbeat_timeout

Parameter description: Specifies the heartbeat timeout that triggers the automatic removal of faulty CNs. The setting of this parameter takes effect immediately, and you do not need to restart the cluster.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2592000. **0** indicates that a faulty CN will not be automatically removed.

Default value: 25

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, the CN may fail to be removed for a long time. If the value is too small, the CN may fail to be removed.

phony_dead_effective_time

Parameter description: Specifies the maximum number of times CN, DN, or GTM processes are detected as zombie. If the number of times a process is detected as zombie is greater than the specified value, the process is considered as a zombie process and will be restarted.

Parameter type: integer.

Unit: none

Value range: 1 to 2147483647

Default value: **5**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

enable_transaction_read_only

Parameter description: Specifies whether to enable the threshold detection function of the CM Server disk. After this function is enabled, CM Server automatically sets the database to read-only when the disk usage is greater than the value of **datastorage_threshold_value_check**.

Parameter type: Boolean

Unit: none

Value range:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **on**: The threshold detection function is enabled.
- **off**: The threshold detection function is disabled.

Default value: **on**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Disabling this function may cause high disk usage.

enable_disk_full_check

Parameter description: Specifies whether to enable the check for full disks. (If the disk usage exceeds the value of **disk_full_threshold**, instance processes on the faulty node will be stopped.)

Parameter type: Boolean

Unit: none

Value range:

- **on**: The disk full check is enabled.
- **off**: The disk full check is disabled.

Default value: **on**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

disk_full_threshold

Parameter description: Specifies the disk usage threshold in a database. When the disk usage of the data directory exceeds the specified value, the disk is full and the database is automatically stopped. When the disk usage is less than the value specified by **datastorage_threshold_value_check**, the database is automatically started. The value of **disk_full_threshold** takes effect only when it is greater than that of **datastorage_threshold_value_check**.

Parameter type: integer.

Unit: %

Value range: 1 to 99

Default value: **98**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If the value is too large, other processes may be unavailable.

datastorage_threshold_check_interval

Parameter description: Specifies the interval for checking the disk usage. The system checks the disk usage at the interval specified by the user.

Parameter type: integer.

Unit: seconds

Value range: 1 to 2592000

Default value: **10**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, CM fault detection and recovery cannot be performed in a timely manner. If the value is too small, the CPU load increases.

datastorage_threshold_value_check

Parameter description: Specifies the usage threshold of a read-only disk in a database. When the disk usage of the data directory exceeds the specified value, the database is automatically set to read-only mode. In the read-only mode, log replay cannot be restricted and the disk usage increases. Therefore, you need to pay attention to the cluster disk capacity and handle read-only alarms in a timely manner to prevent the disk space from being used up; otherwise, the database cannot be restored quickly.

Parameter type: integer.

Unit: %

Value range: 1 to 99

Default value: **85**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value. You can adjust this parameter based on the disk space. When adjusting this parameter, you are advised to adjust the **max_size_for_xlog_retention** parameter of the DN at the same time to prevent the cluster read-only threshold from being triggered by backup operations.

This parameter is default to 85% considering the following factors:

- GaussDB is usually used to carry core services featuring large storage capacity and high concurrency requirements. Therefore, reserving 15% of the space ensures that there is enough space to allow for data increase during the period before storage scale-up when the disk space is insufficient.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Disk space is reserved for O&M operations after the database enters the read-only mode, such as log processing, space recycling, and backup, to ensure sufficient space for O&M operations when a problem occurs.
- Most customers prefer the default setting of 85%.

Risks and impacts of improper settings: If the value is too large, the disk space usage is too high and the restoration cannot be achieved quickly.

datastorage_threshold_recover

Parameter description: Specifies the buffer threshold for canceling the database read-only mode. If the disk usage of the data directory exceeds the value of **datastorage_threshold_value_check**, the data directory is set to read-only. The read-only mode is disabled only when the disk usage is less than the value of **datastorage_threshold_value_check** minus **datastorage_threshold_recover**. For a removed CN, to prevent repeated removal and addition, the CN can be added only when the disk usage is less than the value of **datastorage_threshold_value_check** minus **datastorage_threshold_recover**. If the disk usage exceeds the value, the CN cannot be added back. After the primary database becomes read-only, a switchover can be performed only when the disk usage of the standby database is less than the value of **datastorage_threshold_value_check** minus **datastorage_threshold_recover**.

Parameter type: integer.

Unit: %

Value range: 0 to 99

Default value: **5**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

max_datastorage_threshold_check

Parameter description: Specifies the maximum interval for checking the disk usage. After you modify the read-only mode parameter, the system automatically checks whether the disk usage reaches the threshold at the specified interval.

Parameter type: integer.

Unit: seconds

Value range: 1 to 2592000

Default value: **43200**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If the value is too large, CM fault detection and recovery cannot be performed in a timely manner.

enable_az_auto_switchover

Parameter description: Specifies whether to enable the automatic copy reduction function.

Parameter type: integer.

Unit: none

Value range: 0 to 1

- **0**: disabled.
- **1**: enabled.

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this function is disabled, the CM cannot restore the cluster by reducing copies when a majority of nodes are faulty.

enable_az_conn_check

Parameter description: Specifies whether to enable AZ network arbitration. The CM Server will shut down the isolated AZ.

Parameter type: integer.

Unit: none

Value range: 0 to 1

- **0**: disabled.
- **1**: enabled. After this function is enabled, the CM Server performs AZ network arbitration.

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

cm_auth_method

Parameter description: Specifies the port authentication mode of the CM module.

Parameter type: enumerated type

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range:

- **trust**: Port authentication is not configured.
- **gss**: Kerberos port authentication is used.

Default value: **trust**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: You can change the value to **"gss"** only after the Kerberos server and client are successfully installed. Otherwise, the CM cannot communicate properly, affecting the cluster status.

cm_krb_server_keyfile

Parameter description: Specifies the location of the key file on the Kerberos server. The value must be an absolute path. The file is usually stored in the *${GAUSSHOME}***/kerberos** directory and ends with keytab. The file name is the same as the name of the user who runs the cluster.

Parameter type: string.

Unit: none

Value range: absolute path of the key file of the Kerberos server.

Default value: *${GAUSSHOME}***/kerberos/***{UserName}***.keytab**. The default value cannot take effect and is used only as a prompt.

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: This parameter is paired with the preceding parameter **cm_auth_method**. When **cm_auth_method** is set to **"gss"**, this parameter must be set to a correct path. Otherwise, the cluster status will be affected.

cm_hadr_process_type

Parameter description: Specifies the DR process ID.

Parameter type: enumerated type

Unit: none

Value range:

- **none**: There is no process.
- **failover**: A standby cluster for DR is promoted to primary.
- **switchover_promote**: A standby cluster for DR is promoted to primary during the cluster switchover.
- **switchover_demote**: A primary cluster is demoted to a standby cluster for DR during the cluster switchover.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **none**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

cma_fault_kill_instance_timeout

Parameter description: Specifies timeout interval for waiting for a DN to be reported after the internal status of the primary CM Server is set to **unknown**. The CM Server runs the **pssh** command to remotely kill the DN that fails to be reported and restart the DN process. This parameter is not added to the CM Server configuration file by default. You need to manually add it.

Parameter type: integer.

Unit: seconds

Value range: 0 to 21474836. **0**: disabled.

Default value: This parameter is enabled by default. The default value is (**cm_server_num** x **agent_connect_timeout** x **agent_connect_retries** + **agent_heartbeat_timeout** + 10).

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Faults may not be rectified in a timely manner.

cma_disconnect_to_cms_alarm_timeout

Parameter description: Specifies the timeout period for the CM Agent to disconnect from the primary CM Server. After the timeout, the cluster reports the ALM_AI_AbnormalConn alarm. When the duration of consecutive disconnections doubles the threshold, the ALM_AI_LongTimeAbnormalConn alarm is continuously reported.

Parameter type: integer.

Unit: seconds

Value range: 0 to 1000

- **0**: None of ALM_AI_AbnormalConn or ALM_AI_LongTimeAbnormalConn is reported.
- 1 to 1000: indicates the duration when the CM Agent is disconnected from the primary CM Server.

Default value: **40**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: none

cm_server_arbitrate_delay_base_time_out

Parameter description: Specifies the basic delay duration for CM Server arbitration. If the primary CM Server is disconnected, the arbitration starts to be timed. If the disconnection duration exceeds the arbitration delay duration, a new primary CM Server will be selected. The arbitration delay duration is determined by the basic delay duration, the node index (server ID), and the incremental delay duration. The formula is as follows: Arbitration delay duration = Basic delay duration + Node index x Incremental delay duration.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647

Default value: 10

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the CM Server disconnection fault is false positive. If the value is too large, the CM fault cannot be rectified in a timely manner.

cm_server_arbitrate_delay_incrememtal_time_out

Parameter description: Specifies the incremental delay duration for CM Server arbitration. If the primary CM Server is disconnected, the arbitration starts to be timed. If the disconnection duration exceeds the arbitration delay duration, a new primary CM Server will be selected. The arbitration delay duration is determined by the basic delay duration, the node index (server ID), and the incremental delay duration. The formula is as follows: Arbitration delay duration = Basic delay duration + Node index x Incremental delay duration.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647

Default value: **3**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the CM Server disconnection fault is false positive. If the value is too large, the CM fault cannot be rectified in a timely manner.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

cmserver_ha_connect_timeout

Parameter description: Specifies the time to wait before the attempt of CM Server to connect to other CM Servers times out.

Parameter type: integer.

Unit: seconds

Value range: 1 to 2147483647

Default value: **2**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

cmserver_ha_heartbeat_timeout

Parameter description: Specifies the timeout interval for a CM Server to receive heartbeat signals from other CM Servers.

Parameter type: integer.

Unit: seconds

Value range: 1 to 2147483647

Default value: **6**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large or too small, false negative and false positive may occur during fault detection.

cmserver_ha_status_interval

Parameter description: Specifies the interval between synchronizations of primary and standby CM Server status.

Parameter type: integer.

Unit: seconds

Value range: 1 to 2147483647

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: If the value is too large, CM fault detection and recovery cannot be performed in a timely manner. If the value is too small, the CPU load increases.

cmserver_demote_delay_on_etcd_fault

Parameter description: Specifies the delay within which CM Server switches from the primary state to the standby state due to unhealthy ETCD.

Parameter type: integer.

Unit: seconds

Value range: 1 to 2147483647

Default value: **8**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the ETCD fault detection result is false positive. If the value is too large, the CM fault cannot be rectified in a timely manner.

force_promote

Parameter description: Specifies whether CM Server enables forcible switchover (that is, when the cluster status is unknown, the basic functions of the cluster are available at the cost of data loss). This parameter applies to both CNs and DNs.

Parameter type: integer.

Unit: none

Value range: 0 to 1

- **0**: The forced switching logic function is disabled.
- **1**: The forcible switchover logic function is enabled.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: If you can accept the loss of some data caused by forcible service (system) switchover, enable this parameter when the service recovery has the highest priority.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

switch_rto

Parameter description: Specifies the delay for the forcible switchover of CM Server. When **force_promote** is set to **1** and a shard in the cluster does not have primary node, the system starts timing. After the delay, forcible switchover is executed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: seconds

Value range: 60 to 2147483647

Default value: **600**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too small, the database fault is false positive. If the value is too large, the fault cannot be rectified in a timely manner.

enable_finishredo_retrieve

Parameter description: Specifies whether to retrieve data of Xlogs that are cut off by redo logs after a forcible CM Server switchover. If this parameter is set to **on**, data is automatically retrieved after a forcible switchover.

Parameter type: Boolean

Unit: none

Value range:

- **on**: Data is retrieved.
- **off**: Data is not retrieved.

Default value: **off**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Adjust the value according to the actual situation.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

backup_open

Parameter description: Specifies whether the CM runs in the standby cluster for DR.

Parameter type: integer

Unit: none

Value range: 0 to 3

- **0**: No DR relationship is established.
- **1**: The OceanStor Dorado-based DR relationship is established.
- **2**: The streaming replication-based DR relationship is established.
- **3**: The logical DR relationship is established.

Default value: **0**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-2**. The modification takes effect only after the cluster is restarted.

Setting suggestion: Do not enable this parameter for clusters that are not standby clusters for DR. This parameter is internally called. You are advised not to modify it.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

install_type

Parameter description: Specifies the settings related to the standby cluster for DR to distinguish the cluster type.

Parameter type: integer

Unit: none

Value range: 0 to 2

- **0**: a cluster for which no DR relationship is established.
- **1**: an OceanStor Dorado-based cluster.
- **2**: a streaming-based cluster.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-2**. The modification takes effect only after the cluster is restarted.

Setting suggestion: Do not enable this parameter for clusters that are not standby clusters for DR.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_ssl

Parameter description: Specifies whether to enable SSL.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The SSL certificate is enabled.
- **off**: The SSL certificate is disabled.

Default value: **off**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-2**. The modification takes effect only after the cluster is restarted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting suggestion: Disable this function by default. The CM component stores only the control information about clusters and does not involve user data. Therefore, disabling this function by default does not affect user data security.

Risks and impacts of improper settings: none

ssl_cert_expire_alert_threshold

Parameter description: Specifies the SSL certificate expiration alarm time. If the certificate expiration time is less than the value of this parameter, an alarm indicating that the certificate is about to expire is reported.

Parameter type: integer

Unit: day

Value range: 7–180

Default value: **90**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-2**. The modification takes effect only after the cluster is restarted.

Setting suggestion: Adjust the value according to the actual situation.

Risks and impacts of improper settings: If the value is too small, alarms may not be rectified in a timely manner.

ssl_cert_expire_check_interval

Parameter description: Specifies the period for checking whether the SSL certificate expires.

Parameter type: integer

Unit: seconds

Value range: 0 to 2147483647

Default value: **86400** (that is, one day)

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-2**. The modification takes effect only after the cluster is restarted.

Setting suggestion: Adjust the value according to the actual situation.

Risks and impacts of improper settings: If the value is too small, the CPU load increases. If the value is too large, alarms cannot be reported in a timely manner.

delay_arbitrate_timeout

Parameter description: Specifies the waiting time for a node in the same AZ as the primary DN to be promoted to primary after redo replay.

Parameter type: integer

Unit: seconds

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 0 to 2147483647

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If the value is too large, DN faults may not be rectified in a timely manner.

dcf_heartbeat_timeout

Parameter description: Specifies the time to wait before the DCF instance heartbeat times out in the CM&DCF joint arbitration. If the primary DCF instance heartbeat times out, the DCF instance is unreachable by the CM and the CM will automatically trigger a DCF switchover. It takes effect only in DCF self-arbitration mode.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647. If this parameter is set to **0**, heartbeat timeout detection is disabled for DCF instances in joint arbitration.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: After this function is disabled, the CM does not process the unreachable DCF instances. If a DCF instance is unreachable and abnormal, the CM cannot assist the arbitration. Before modifying this parameter, make sure you understand its meaning and fully verify the change in a test environment.

enable_dcf

Parameter description: Specifies the status of the DCF mode.

Parameter type: Boolean.

Unit: none

Value range:

- **on**: The DCF mode is enabled.
- **off**: The DCF mode is disabled.

Default value: **off**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ddb_type

Parameter description: Specifies whether to switch between ETCD and DCC modes.

Parameter type: integer

Unit: none

Value range: 0 to 1

- **0**: The ETCD mode is enabled.
- **1**: The DCC mode is enabled.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 in **Table 6-2**. The modification takes effect only after the cluster is restarted.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ddb_log_level

Parameter description: Specifies the DDB log level.

Parameter type: string.

Unit: none

Value range:

- **"NONE"**: The log function is disabled. It cannot be used together with the following log levels.
- **"RUN_ERR|RUN_WAR|RUN_INF|DEBUG_ERR|DEBUG_WAR|DEBUG_INF|TRACE|PROFILE|OPER"**: Enable the log function. You can set this parameter to one or a combination of the preceding log levels. If two or more log levels are used together, separate them with vertical bars (|). The log level cannot be set to an empty character string.

Default value: **"RUN_ERR|RUN_WAR|DEBUG_ERR|OPER|RUN_INF|PROFILE"**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

ddb_log_backup_file_count

Parameter description: Specifies the maximum number of log files that can be saved.

Parameter type: integer

Unit: none

Value range: 1 to 100

Default value: **10**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ddb_max_log_file_size

Parameter description: Specifies the maximum number of bytes in a log.

Parameter type: string.

Unit: none

Value range: The value of this parameter consists of digits and a unit. The value ranges from 1 to 1000, in MB.

Default value: **"10M"**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ddb_log_suppress_enable

Parameter description: Specifies whether to enable the log suppression function.

Parameter type: integer

Unit: none

Value range: 0 to 1

- **0**: The log suppression function is disabled.
- **1**: The log suppression function is enabled.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ddb_election_timeout

Parameter description: Specifies the DCC election timeout period.

Parameter type: integer

Unit: seconds

Value range: 1–600

Default value: **3**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ddb_dcf_sync_nodes

Parameter description: Specifies the standby node list for DCC synchronous replication.

Parameter type: string.

Unit: none

Value range: a valid standby node list. The value can be in the following format:

ANY *num_sync:node_id*[, ...]

NOTE

num_sync indicates the number of standby nodes to be synchronized and *node_id* indicates the sequence number of each standby node. Example: ddb_dcf_sync_nodes='ANY 1:2,3'.

Default value: **""**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

ddb_dcf_extreme_mode

Parameter description: Specifies the DCC log commit policy.

Parameter type: integer

Unit: none

Value range: 0 to 2

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- **0**: maximum protection, that is, majority commit.
- **1**: maximum availability. When the standby node is faulty, the primary node can choose direct commit. When the standby node is normal, the primary node can choose majority commit.
- **2**: maximum performance, that is, local commit.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

alarm_ddb_unhealth_threshold

Parameter description: Specifies the threshold for reporting an alarm for an unhealthy DDB.

Parameter type: integer.

Unit: none

Value range: 0 to 2592000

Default value: **5**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_e2e_rto

Parameter description: Specifies whether to enable the E2E RTO function. After this function is enabled, the zombie detection period and network detection timeout time are shortened. The CM can reach the E2E RTO value (RTO for a single instance: ≤ 10s; RTO for overlay faults: ≤ 30s).

Parameter type: integer.

Unit: none

Value range: 0 to 1

- **0**: The E2E RTO function is disabled.
- **1**: The E2E RTO function is enabled.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value. You can adjust the value as required.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_synclist_single_inst

Parameter description: Specifies whether to reduce copies to one primary and zero standby. After this function is enabled, the number of copies will be reduced to one primary and zero standby. If the parameter is incorrectly set, the default value is used.

Parameter type: Boolean

Unit: none

Value range:

- **on**: This function is enabled.
- **off**: This function is disabled.

Default value: **off**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value. If you need to change the value, contact Huawei technical support.

Risks and impacts of improper settings: If this parameter is enabled and all standby nodes are faulty, the primary node automatically switches to the local commit mode. In this case, if the primary node is faulty again, unrecoverable impacts may occur.

asyn_log_enable

Parameter description: Specifies whether to enable asynchronous logging. After this parameter is enabled, the CM Server starts asynchronous logging.

Parameter type: Boolean

Unit: none

Value range:

- **on**: The CM Server starts asynchronous logging.
- **off**: The CM Server does not start asynchronous logging but uses serial logging.

Default value: **on**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is disabled, logs may be lost.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

asyn_log_max_buf_size

Parameter description: Specifies the size of the asynchronous log buffer. When asynchronous logs are stacked and the stacked logs exceed the buffer size, subsequent logs cannot be recorded.

Parameter type: integer.

Unit: KB

Value range: 1 to 1048576

Default value: **1024**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Adjust the value based on the memory and hard disk size.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

asyn_log_fault_solution

Parameter description: Specifies the asynchronous log processing solution after a log disk fault is identified. When the CM detects that a log disk is faulty or the asynchronous log buffer overflows, the CM processes asynchronous logs based on the configuration.

Parameter type: integer.

Unit: none

Value range: 0 to 2

- **0**: No extra processing is performed on logs. All logs are retained in the buffer until the buffer size reaches the upper limit. After the buffer size reaches the upper limit, no more logs are recorded.
- **1**: Logs are filtered. After a log disk fault is identified, only key logs, alarms, and logs at the ERROR level or higher are saved to the buffer until the buffer size reaches the upper limit. After the buffer size reaches the upper limit, no more logs are recorded.
- **2**: Logs are discarded. After a log disk fault is identified, only key logs and alarms are saved to the buffer until the buffer size reaches the upper limit. After the buffer size reaches the upper limit, no more logs are recorded.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Adjust the value based on the memory and hard disk size.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

enable_isolate

Parameter description: Specifies whether to isolate instances.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: none

Value range: 0 to 1

- **0**: This function is disabled.
- **1**: This function is enabled. After this function is enabled, CM Server automatically isolates faulty instances.

Default value: **1**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

barrier_restart_threshold

Parameter description: Specifies the interval of waiting for restarting the barrier of the standby cluster for DR. When the barrier of the standby cluster for DR is suspended for a period longer than the value of this parameter, the barrier is restarted.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2592000. **0** indicates that barrier restart is disabled.

Default value: **30**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Improper settings may cause inaccurate fault detection and delayed fault rectification.

enable_topology_network_arbitrate

Parameter description: Specifies whether to perform network topology arbitration. If this parameter is enabled, GaussDB performs arbitration based on the network topology. This parameter can be used only in quorum mode.

Parameter type: integer.

Unit: none

Value range:

- **0**: disabled.
- **1**: enabled. Once enabled, CM Server automatically performs arbitration based on the network topology.

Default value: **1**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

topology_network_arbitrate_wait_time

Parameter description: Specifies the waiting time for network topology data to become stable during network topology arbitration. If the network topology data changes within the period specified by this parameter, network topology arbitration is not performed. This parameter can be used only in quorum mode.

Parameter type: integer.

Unit: seconds

Value range: 5 to 600

Default value: **10**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

topology_network_collecting_failed_times

Parameter description: Specifies the number of consecutive network disconnections on the data plane. If the number of disconnections exceeds the value of this parameter, the data plane network topology is abnormal. The value must be the same as that of **topology_network_collecting_failed_times** set for the CM Agent. This parameter can be used only in quorum mode.

Parameter type: integer.

Unit: none

Value range: 3 to 50

Default value: **5**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

topology_network_stable_cn_percent

Parameter description: Specifies the percentage of remaining CNs that can be loaded on the data plane network topology. The network arbitration module

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

calculates the minimum number of reserved CNs based on this parameter. The minimum number of retained CNs is the number of all CNs multiplied by the value of this parameter. If the calculated number is less than 1, the value **1** is used. This parameter can be used only in quorum mode.

Parameter type: integer.

Unit: none

Value range: 0 to 100

Default value: **50**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

topology_network_check_max_cn_count

Parameter description: Specifies the maximum number of CNs in a cluster supported by data plane network topology detection. If the number of CNs exceeds the value of this parameter, data plane network topology arbitration is not performed. If this parameter is set to **0**, network topology arbitration is disabled. This parameter can be used only in quorum mode.

Parameter type: integer.

Unit: none

Value range: 0 to 256

Default value: **16**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

hardware_max_cn_delete_threshold

Parameter description: Specifies the maximum threshold for removing CNs due to hardware faults. The value is rounded down.

Parameter type: integer.

Unit: %

Value range: 0 to 99

- **0**: CNs are not removed.
- 1–99: indicates the percentage of CNs to be removed. At least one available CN must be retained.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Default value: **50**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

cluster_starting_aribt_delay

Parameter description: Specifies the time that CM Server waits for the static primary DN to be promoted to primary during cluster startup.

Parameter type: integer.

Unit: seconds

Value range: 1 to 2592000

Default value: **180**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 1 and method 2 in **Table 6-2**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

cluster_unavailable_alarm_timeout

Parameter description: Specifies the timeout period for a cluster in the unavailable state. After the timeout, the cluster reports the ALM_AI_ClusterUnavailable alarm.

Parameter type: integer.

Unit: seconds

Value range: 0 to 1000

- **0**: This alarm is not reported.
- 1–1000: indicates the time when the cluster is in an unavailable state.

Default value: **40**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: none

cluster_degrade_alarm_timeout

Parameter description: Specifies the timeout period for a cluster to be in the **Degrade** state. After the timeout, the cluster reports the ALM_AI_ClusterDegrade alarm.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter type: integer.

Unit: seconds

Value range: 0 to 1000

- **0**: This alarm is not reported.
- 1–1000: indicates the time when the cluster is in the **Degrade** state.

Default value: **40**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: none

max_frequency_heal_timeout

Parameter description: The ALM_AI_ClusterHealFrequent alarm is triggered if the cluster is healed for 10 times or more within this period.

Parameter type: integer.

Unit: seconds

Value range: 0 to 1000

- **0**: This alarm is not reported.
- 1 to 1000: indicates the time range for continuously triggering the healing process.

Default value: **300**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: none

cn_auto_repair_max_concurrency

Parameter description: Specifies the maximum number of CNs that can be concurrently added back.

Parameter type: integer.

Unit: none

Value range: 0 to 256. **0** indicates that the number of CNs concurrently added back is the same as the number of normal CNs in the cluster but cannot be greater than 10.

Default value: **0**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

cn_auto_repair_timeout

Parameter description: Specifies the timeout period for adding CNs back. If a CN fails to be added back repeatedly in a period longer than this threshold and the number of attempt failures exceeds this retry threshold, it will be abandoned from automatic re-addition.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **1800**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

cn_auto_repair_retry_times

Parameter description: Specifies the number of CN re-addition retries. If a CN fails to be added back repeatedly in a period longer than the timeout threshold and the number of attempt failures exceeds this retry threshold, it will be abandoned from automatic re-addition. Though it is abandoned, if you adjust the value of this threshold, automatic re-addition of the faulty CN will start again.

Parameter type: integer.

Unit: none

Value range: 0 to 2147483647

Default value: **3**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

cn_dn_disconnect_to_delete_time

Parameter description: Specifies the number of times that a CN fails to check SQL statements on all primary DNs. When the number of failures exceeds the value of this parameter, the CN is removed.

Parameter type: integer.

Unit: none

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value range: 1 to 600

Default value: **20**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

majority_standby_lock_timeout

Parameter description: Specifies the timeout interval for locking the majority of standby nodes. The cluster enters the leader election (arbitration) process and locks the majority of standby nodes. When the primary node is faulty, the arbitration process is blocked until this timeout period expires. After the timeout period expires, the original primary node is stopped and the arbitration process is resumed.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647. The value **0** indicates that the function is disabled.

Default value: **120**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: Change the parameter value after fully understanding the parameter meaning and verifying it through testing.

replica_reduce_delay_time

Parameter description: Specifies the delay time for reducing copies. If the majority of DNs are in the **unknown** state, the copy reduction operation is triggered only after the detected state persists for a period of time specified by this parameter. If the majority of DNs are in the **building**, **need_repair**, or **starting** state, the copy reduction operation is triggered only after the detected state persists for a period of time five times the specified delay.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647

Default value: **12**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a small value, the probability of incorrect copy reduction increases in a short period of time due

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

to faults such as network disconnection. If this parameter is set to a large value, the cluster is unavailable for a long time when the majority of nodes are faulty. Before modifying this parameter, make sure you understand its meaning and fully verify the change in a test environment.

replica_increase_delay_time

Parameter description: Specifies the delay time for adding copies. When a copy is recovered and can be added back (that is, the copy status is **normal** and the log synchronization rate is greater than or equal to 99%), the automatic copy addition is triggered only after a delay specified by this parameter. This parameter is used to prevent resource flapping caused by immediate copy addback after copy reduction.

Parameter type: integer.

Unit: seconds

Value range: 0 to 2147483647

Default value: **60**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: Retain the default value.

Risks and impacts of improper settings: If this parameter is set to a small value, resource flapping caused by immediate copy addition after copy reduction may occur. If this parameter is set to a large value, the time for adding copies back after they are recovered is prolonged. Before modifying this parameter, make sure you understand its meaning and fully verify the change in a test environment.

dr_dest_name

Parameter description: Specifies the target clusters to which logs can be synchronized from the current cluster. This parameter is valid only when the cluster is deployed in a single-node system. In addition, you do not need to reload the parameter after setting this parameter.

Parameter type: string

Unit: none

Value range: a valid target cluster instance ID.

Default value: **""**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**. After this parameter is set, you do not need to reload the parameter.

Setting suggestion: This parameter is internally called. You are advised not to modify it.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

hadr_replconninfo_config_mode

Parameter description: Specifies the DR type of the current database instance.

Parameter type: string

Unit: none

Value range: **"local_file"** or **"cluster_manager"**

Default value: **"local_file"**

Setting method: This parameter is a CM parameter. For details about how to modify this parameter, see method 2 in **Table 6-3**.

Setting suggestion: This parameter is internally called. You are advised not to modify it.

Risks and impacts of improper settings: Changing the value of this parameter may cause unexpected results in the cluster. Retain the default value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7 System Catalogs and System Views

7.1 Overview of System Catalogs and System Views

System catalogs store the structured metadata of GaussDB. They are the source of information used by GaussDB to control system running and are a core component of the database system.

System views provide ways to query system catalogs and internal database status.

System catalogs and system views are visible to either system administrators or all users. Some of the following system catalogs and views have marked the need of administrator permissions. They are accessible only to administrators.

You can delete and rebuild the catalogs, add columns to them, and insert and update values in them, but doing so may make system information inconsistent and cause system faults. Generally, users should not modify system catalogs or system views, or rename their schemas. They are automatically maintained by the system.

NOTICE

- You are advised not to modify the permissions on system catalogs or system views.
- Do not add, delete, or modify system catalogs because doing so will result in exceptions or even cluster unavailability.
- System catalogs and views do not support foreign key-related columns.
- For details about column types in system catalogs and system views, see **Data Types**.
- For ADM views, the accessed objects are all objects of this type in the database. Therefore, unified permission management is required for ADM views. By default, only system administrators have the permission to access ADM views. Some ADM view data comes from public and non-sensitive columns in base tables. For DB views, the objects that the current user has the permission to access in the database can be queried. Common users can access these views. For MY views, objects to which the current user belongs can be queried. Common users can access these views.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2 System Catalogs

7.2.1 Partitioned Table

7.2.1.1 PG_PARTITION

PG_PARTITION records information about all partitioned tables, table partitions, and index partitions in the database, but it does not store information about partitioned indexes. Since partitioned tables do not have actual physical files, PG_PARTITION does not record information such as **relfilenode**, **relpages**, **reltuples**, **reltoastrelid**, or **reltoastidxid**.

Table 7-1 PG_PARTITION columns

Name	Type	Description
oid	OID	Row identifier (implicit attribute, which must be explicitly selected).
relname	Name	Name of a partitioned table, a partition, a TOAST table on a partition, or an index partition.
parttype	"Char"	Object type. • **'r'**: partitioned table. • **'p'**: table partition. • **'s'**: table subpartition. • **'x'**: index partition.
parentid	OID	• OID of a partitioned table in PG_CLASS when the object is a partitioned table or partition. • OID of a partitioned index when the object is an index partition.
rangenum	Integer	Reserved column.
intervalnum	Integer	Reserved column.
partstrategy	"Char"	Partitioning policy of the partitioned table. Currently, only the following policies are supported: • **'r'**: range partitioning. • **'l'**: list partitioning. • **'h'**: hash partitioning. • **'n'**: no partitioning because the object is not a table partition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
relfilenode	OID	Physical storage location of a table partition, an index partition, or a TOAST table on a partition.
reltablespace	OID	OID of the tablespace containing a table partition, an index partition, or a TOAST table on a partition.
relpages	Double precision	Statistics: number of data pages in a table partition or an index partition.
reltuples	Double precision	Statistics: number of tuples in a table partition or an index partition.
relallvisible	Integer	Statistics: number of visible data pages in a table partition or an index partition.
reltoastrelid	OID	OID of the TOAST table corresponding to a table partition.
reltoastidxid	OID	Index OID of the TOAST table corresponding to a table partition.
indextblid	OID	OID of the table partition corresponding to an index partition.
indisusable	Boolean	Specifies whether an index partition is available. • **t** (true): available. • **f** (false): unavailable.
reldeltarelid	OID	Not supported currently.
reldeltaidx	OID	Not supported currently.
relcudescrelid	OID	Not supported currently.
relcudescidx	OID	Not supported currently.
relfrozenxid	XID32	Frozen transaction ID. To ensure compatibility with earlier versions, this column is reserved. The **relfrozenxid64** column is added to record the information.
intspnum	Integer	Number of tablespaces to which an interval partition belongs.
partkey	Int2vector	Column number of the partition key.
intervaltablespace	Oidvector	Tablespace to which an interval partition belongs. Interval partitions fall into tablespaces in a round-robin manner.
interval	Text[]	Interval value of an interval partition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
boundaries	Text[]	Upper limit of a range partition or an interval partition.
transit	Text[]	Transit of an interval partition.
reloptions	Text[]	Partition storage attribute used to collect online scale-out information. Same as **pg_class.reloptions**, it is expressed as a string in "keyword=value" format.
relfrozenxid64	XID	Frozen transaction ID.
relminmxid	XID	Frozen multi-transaction ID.
partitionno	Integer	Used to maintain the partition map of a partitioned table. • If the object is a partition, this column indicates the partition ID, which increments from 1. • If the object is a partitioned table, this column indicates the maximum partition ID, which is specially marked with a negative value. This value increases with certain partition-related DDL syntax. • For other object types, this column is null and holds no specific meaning. **partitionno** is a permanent auto-increment column, which can be reset or reclaimed by using the ALTER TABLE t_name RESET PARTITION syntax.
subpartitionno	Integer	Reserved column.
partkeyexpr	pg_node_tree	Not supported currently.

7.2.1.2 GS_SUBPARTITION_TEMPLATE

GS_SUBPARTITION_TEMPLATE records information about subpartition templates.

Table 7-2 GS_SUBPARTITION_TEMPLATE columns

Name	Type	Description
table_oid	OID	OID of the base table to which a template belongs.
position	Integer	Partition number defined in the template, starting from 1.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
spart_name	Name	Partition name defined in the template.
spart_tablespace	OID	Partition tablespace OID defined in the template.
spart_boundaries	Text[]	Partition boundary defined in the template. For range partitioning, it is defined as an upper limit. For list partitioning, it is defined as a value list. For hash partitioning, it is defined as a hash remainder.

7.2.2 Workload Management

7.2.2.1 GS_ABNORMAL_SQL_INFO

GS_ABNORMAL_SQL_INFO provides information about SQL statements that cause abnormal CPU usage of the database.

Table 7-3 GS_ABNORMAL_SQL_INFO columns

Column Name	Type	Description
unique_sql_id	bigint	Unique SQL ID.
abnormal_count	integer	Number of exceptions.
current_controllable	boolean	Specifies whether the CPU usage is controlled.
is_logged_when_controlled	boolean	Specifies whether to print logs when the CPU usage is controlled.
control_expiration_time	timestamp without time zone	Control expiration time.
abnormal_type	name	Exception type.
user_id	oid	OID of the user to which the abnormal SQL statement belongs.

7.2.2.2 GS_ABNORMAL_SQL_WHITE_LIST

GS_ABNORMAL_SQL_WHITE_LIST provides information about the SQL whitelist for abnormal database CPU usage.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-4 GS_ABNORMAL_SQL_INFO columns

Column Name	Type	Description
unique_sql_id	bigint	Unique SQL ID.
user_id	oid	OID of the user to which the abnormal SQL statement belongs.

7.2.2.3 PG_APP_WORKLOADGROUP_MAPPING

PG_APP_WORKLOADGROUP_MAPPING provides load mapping group information in the database.

Table 7-5 PG_APP_WORKLOADGROUP_MAPPING columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
appname	name	Application name
workload_gpname	name	Mapped workload group name

7.2.3 OLTP Table Compression

7.2.3.1 GS_ILM

GS_ILM provides the main information about the ILM policy, including the policy name, policy owner, policy type, policy ID, and policy status.

Table 7-6 GS_ILM columns

Name	Type	Description
pidx	integer	Policy sequence number, which is globally unique and starts with 1.
creator	oid	Policy owner.
name	name	Policy name. Currently, the policy name cannot be customized. The default name is p +pidx.
ptype	"char"	Policy type: • **m**: data movement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
flag	smallint	Value range: • **0**: The policy is enabled. • **1**: The policy is disabled. • **2**: Database-level policy.

7.2.3.2 GS_ILM_JOBDETAIL

GS_ILM_JOBDETAIL records information including the job type, job status, and job statistics.

Table 7-7 GS_ILM_JOBDETAIL columns

Name	Type	Description
taskoid	bigint	OID of an ADO task.
jobtype	"char"	Current job type: compression ('c').
jobstatus	smallint	Current job status: • **1**: JOB CREATED • **2**: COMPLETED SUCCESSFULLY • **3**: FAILED • **4**: STOPPED • **5**: JOB CREATION FAILED
jobname	text	The ADO job name is unique in the table and is joined with **job_name** in the PG_JOB system catalog.
starttime	timestamp with time zone	Start time.
completetime	timestamp with time zone	Completion time.
payload	text	PL/SQL code of the current job.
statistics	text	Statistics of the current job.
comments	text	Description of the current job execution, such as the failure cause.

7.2.3.3 GS_ILM_OBJECT

GS_ILM_OBJECT records the relationships between data objects and policies as well as the scheduling information of policies on the data objects. When a policy is set for a partitioned table, this catalog generates an independent record for each partition or subpartition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-8 GS_ILM_OBJECT columns

Name	Type	Description
pidx	integer	Policy ID.
objoid	oid	OID of a logical data object which has a partitioned table.
dataobjoid	oid	OID of a physical data object, which is the entity that needs to execute the policy and the leaf of the tree-level partition relationship.
objtype	"char"	Current data object type. The options are as follows: • Table ('r') • Partition ('p') • Subpartition ('s') • Ordinary index ('i') • Global index ('I') • Global secondary index ('G') • Index Partition ('x')
origobjoid	oid	OID of the original object that defines the policy.
origobjtype	"char"	Type of the original data object that defines the policy. The options are as follows: • Table ('r') • Partition ('p') • Subpartition ('s') • Ordinary index ('i') • Global index ('I') • Global secondary index ('G')
lastchktime	timestamp with time zone	Time of the last ADO task.
lastexetime	timestamp with time zone	Time of the last ADO job.
roundcnt	smallint	Number of successful rounds.
failcnt	smallint	Number of failures.
lastjobstatus	"char"	Execution status of the last job.
lastroundstarttime	timestamp with time zone	Start time of the full table scan.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
lastjobblkid	bigint	ID of the block that has been processed by the job last time.
flag	smallint	Value range: • **0**: The object compression policy is enabled. • **1**: The object compression policy is disabled. • **2**: The object compression policy is enabled, and a full table scan is completed during the last compression scheduling.
lasthashbktpos	oidvector	Records the last compression position of the previous compression scheduling by storing two values: **[bucketid, blockid]**.

7.2.3.4 GS_ILM_PARAM

GS_ILM_PARAM is a feature parameter table, which records the parameters related to the ILM feature, such as the control parameters of background scheduling.

Table 7-9 GS_ILM_PARAM columns

Name	Type	Description
idx	smallint	Parameter ID.
name	name	Parameter name.
value	double precision	Parameter value. For details, see the description in **Table 7-10**.

Table 7-10 GS_ILM_PARAM feature parameter range

Parameter ID	Parameter	Description
1	EXECUTION_INTERVAL	Specifies the frequency of executing an ADO task, in minutes. The default value is **15**. The value is an integer or floating-point number greater than or equal to 1 and less than or equal to 2147483647. The value is rounded down.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter ID	Parameter	Description
2	RETENTION_TIME	Specifies the retention period of ADO-related history records, in days. The default value is **30**. The value is an integer or floating-point number greater than or equal to 1 and less than or equal to 2147483647. The value is rounded down.
7	ENABLE	Background scheduling status. This parameter cannot be modified using DBE_ILM_ADMIN.CUSTOMIZE_ILM. Use disable_ilm () and enable_ilm () to modify it.
11	POLICY_TIME	Specifies whether the time unit of ADO is day or second. The time unit second is used only for testing. Value range: ● **0** (default): The unit is day. ● **1**: The unit is second, which is generally used in test scenarios.
12	ABS_JOBLIMIT	Specifies the maximum number of ADO jobs generated by an ADO task. The default value is **10**. The value is an integer or floating-point number greater than or equal to 0 and less than or equal to 2147483647. The value is rounded down.
13	JOB_SIZELIMIT	Specifies the maximum number of bytes that can be processed by a single ADO job. The unit is MB. The default value is **1024**. The value is an integer or floating-point number greater than or equal to 0 and less than or equal to 2147483647. The value is rounded down.
14	WIND_DURATION	Specifies the maintenance window duration, in minutes. The default value is 240 minutes (4 hours). The value is an integer or floating-point number greater than or equal to 0 and less than 1440 (24 hours). The value is rounded down for a floating-point number before taking effect.
15	BLOCK_LIMITS	Specifies the upper limit of the instance-level row-store compression speed. The default value is **40**. The value ranges from 0 to 10000, in blocks/ms, indicating the maximum number of blocks that can be compressed per millisecond. **0** indicates that the speed is not limited.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter ID	Parameter	Description
16	ENABLE_META_COMPRESSION	Specifies whether to enable header compression. The default value is **0** (disabled), and the value **1** indicates that header compression is enabled. **NOTE** If this parameter is set to **1**, the compression ratio is improved for tables with short rows (less data in a single row), but the performance of accessing compressed rows greatly deteriorates. If most tables in the database have long rows, you are advised not to enable this parameter.
17	SAMPLE_MIN	Minimum sampling step for constant encoding and equivalent encoding. The default value is **10**. The value range is [1,100]. Decimals can be entered and will be automatically rounded down.
18	SAMPLE_MAX	Maximum sampling step for constant encoding and equivalent encoding. The default value is **10**. The value range is [1,100]. Decimals can be entered and will be automatically rounded down.
19	CONST_PRIO	Constant encoding priority. The default value is **40**. The value range is [0,100]. The value **100** indicates that constant encoding is disabled. Decimals can be entered and will be automatically rounded down.
20	CONST_THRESHOLD	Constant encoding threshold. The default value is **90**. The value range is [1,100], indicating that constant encoding is performed when the proportion of constant values in a column exceeds the threshold. Decimals can be entered and will be automatically rounded down.
21	EQVALUE_PRIO	Equivalent encoding priority. The default value is **60**. The value range is [0,100]. The value **100** indicates that equivalent encoding is disabled. Decimals can be entered and will be automatically rounded down.
22	EQVALUE_THRESHOLD	Equivalent encoding threshold. The default value is **80**. The value range is [1,100], indicating that equivalent encoding is performed when the ratio of equal values in two columns exceeds the threshold. Decimals can be entered and will be automatically rounded down.
23	ENABLE_DELTA_ENCODE_SWITCH	Specifies whether to enable difference encoding. The default value is **1**. The value **0** indicates that difference encoding is disabled, and the value **1** indicates that difference encoding is enabled. Decimals can be entered and will be automatically rounded down.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter ID	Parameter	Description
24	LZ4_COMPRESSION_LEVEL	LZ4 compression level. The default value is **0**. The value range is [0,16]. Decimals can be entered and will be automatically rounded down.
25	ENABLE_LZ4_PARTIAL_DECOMPRESSION	Specifies whether to enable partial decompression. The default value is **1**. The value **0** indicates that partial decompression is disabled, and the value **1** indicates that partial decompression is enabled. Decimals can be entered and will be automatically rounded down.
26	INDEXJOB_SIZELIMIT	Specifies the maximum number of bytes that can be processed by a single ADO job for index compression. The unit is MB. The default value is **1024**. The value is an integer or floating-point number greater than or equal to 0 and less than or equal to 2147483647. The value is rounded down.
27	ENABLE_INDEX_COMPRESS	Specifies whether to enable index compression. The value can be **0** (disabled) or **1** (enabled). The default value is **0**.

7.2.3.5 GS_ILM_POLICY

GS_ILM_POLICY records policy action types and judgment conditions.

Table 7-11 GS_ILM_POLICY columns

Name	Type	Description
pidx	integer	Policy ID.
action	"char"	Policy action. Only compression ('c') is supported.
ctype	"char"	Compression type. Only advanced row compression ('a') and cross-page transparent compression ('t') are supported.
condition	"char"	Condition type. Only LAST MODIFICATION('m') is supported.
days	smallint	Number of judgment days.
scope	"char"	Policy scope. Only Row ('r') and Unit ('u') are supported.
predicate	pg_node_tree	Policy row-level expression.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
csubtype	"char"	Compression level. Advanced compression supports only Medium ('m') and High ('h'). Cross-page transparent compression supports Low ('l'), Medium ('m'), and High ('h').

7.2.3.6 GS_ILM_TASK

GS_ILM_TASK records ADO task information, such as the creation time, start time, end time, and execution status.

Table 7-12 GS_ILM_TASK columns

Name	Type	Description
taskoid	bigint	OID of an ADO task.
creator	oid	Creator of an ADO task.
createtime	timestamp with time zone	Creation time.
starttime	timestamp with time zone	Start time.
completetime	timestamp with time zone	Completion time.
executestate	smallint	Value range: • **1**: 'INACTIVE' • **2**: 'ACTIVE' • **3**: 'COMPLETED'
flag	smallint	Reserved column.

7.2.3.7 GS_ILM_TASKDETAIL

GS_ILM_TASKDETAIL records the evaluation result of a specified data object and policy.

Table 7-13 GS_ILM_TASKDETAIL columns

Name	Type	Description
pidx	integer	Policy ID.
objoid	oid	Data OID.
objtype	"char"	Data object type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
taskoid	bigint	OID of an ADO task.
evalresult	smallint	Evaluation result: • **0**: SELECTED FOR EXECUTION. The evaluation is passed. • **1**: PRECONDITION NOT SATISFIED. The condition is not met. • **2**: JOB ALREADY EXISTS. The job already exists.
jobname	text	ADO job name generated after the evaluation is passed is joined with the **jobname** column in the **GS_ILM_JOBDETAIL** system catalog.

7.2.3.8 GS_ILM_TICKER

GS_ILM_TICKER records the mapping between LSNs and time. A maximum of 3650 lines of records can be retained.

Table 7-14 GS_ILM_TICKER columns

Name	Type	Description
ilm_ticker_lsn	bigint	LSN when the dotting operation is triggered.
ilm_ticker_time	timestamp with time zone	Time when the dotting operation is triggered.

7.2.3.9 GS_ILM_TURBO_OBJECT

GS_ILM_TURBO_OBJECT records information about cross-page transparent compression policies.

Table 7-15 GS_ILM_TURBO_OBJECT columns

Name	Type	Description
objoid	oid	OID of the original object that defines the policy.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
objtype	"char"	Current data object type. The options are as follows: • Table ('r') • Partition ('p') • Subpartition ('s') • Ordinary index ('i') • Global index ('I') • Global secondary index ('G') • Index Partition ('x')
csubtype	"char"	Cross-page transparent compression policy level: • Low ('l') • Medium ('m') • High ('h')
encodetype	tinyint	Encoding type: • Tile ('1') • Fuzzy splitting ('2') • Pattern encoding ('3')
compresstype	tinyint	General compressor type: • lz4 ('1') • zstd ('2')
compressunit	tinyint	Length of the compression unit: • 1 ('1') • 2 ('2') • 4 ('3') • 8 ('4') • 16 ('5') • 32 ('6') • 64 ('7')
lmtinterval	integer	Interval of cold and hot data management, in hours.

7.2.4 Encrypted Equality Query

7.2.4.1 GS_CLIENT_GLOBAL_KEYS

GS_CLIENT_GLOBAL_KEYS records information about the CMK in the encrypted equality feature. Each record corresponds to a CMK.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-16 GS_CLIENT_GLOBAL_KEYS columns

Name	Type	Description
oid	oid	Row identifier (hidden column).
global_key_name	name	CMK name.
key_namespace	oid	A namespace OID that contains this CMK.
key_owner	oid	CMK owner.
key_acl	aclitem[]	Access permissions that the key should have on creation
create_date	timestamp without time zone	Time when the key is created

7.2.4.2 GS_CLIENT_GLOBAL_KEYS_ARGS

GS_CLIENT_GLOBAL_KEYS_ARGS records the metadata about the CMK in the encrypted equality feature. Each record corresponds to a key-value pair of the CMK.

Table 7-17 GS_CLIENT_GLOBAL_KEYS_ARGS columns

Name	Type	Description
oid	oid	Row identifier (hidden column).
global_key_id	oid	CMK OID.
function_name	name	The value is **encryption**.
key	name	CMK metadata name.
value	bytea	Value of the CMK metadata name.

7.2.4.3 GS_COLUMN_KEYS

GS_COLUMN_KEYS records information about the CEK in the encrypted equality feature. Each record corresponds to a CEK.

Table 7-18 GS_COLUMN_KEYS columns

Name	Type	Description
oid	oid	Row identifier (hidden column).
column_key_name	name	CEK name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
column_key_distributed_id	oid	ID obtained based on the hash value of the fully qualified domain name (FQDN) of the CEK.
global_key_id	oid	Foreign key, which is the CMK OID.
key_namespace	oid	A namespace OID that contains this CEK.
key_owner	oid	CEK owner.
create_date	timestamp without time zone	Time when the CEK is created
key_acl	aclitem[]	Access permissions that the CEK should have on creation

7.2.4.4 GS_COLUMN_KEYS_ARGS

GS_COLUMN_KEYS_ARGS records the metadata about the CMK in the encrypted equality feature. Each record corresponds to a key-value pair of the CMK.

Table 7-19 GS_COLUMN_KEYS_ARGS columns

Name	Type	Description
oid	oid	Row identifier (hidden column).
column_key_id	oid	CEK OID
function_name	name	The value is **encryption**.
key	name	CEK metadata name
value	bytea	Value of the CEK metadata name

7.2.4.5 GS_ENCRYPTED_COLUMNS

GS_ENCRYPTED_COLUMNS records information about encrypted columns in the encrypted equality feature. Each record corresponds to an encrypted column.

Table 7-20 GS_ENCRYPTED_COLUMNS columns

Name	Type	Description
rel_id	oid	OID of the table.
column_name	name	Name of an encrypted column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
column_key_id	oid	A foreign key, which is the CEK OID
encryption_type	tinyint	Encryption type. The value can be **2(DETERMINISTIC)** or **1(RANDOMIZED)**.
data_type_original_oid	oid	ID of the original data type of the encrypted column. For details about the values, see the **oid** columns in the **PG_TYPE** system catalog.
data_type_original_mod	integer	Modifiers of the original data type of the encrypted column. For details about the values, see the **atttypmod** columns in the **PG_ATTRIBUTE** system catalog.
create_date	timestamp without time zone	Time when an encrypted column is created

7.2.4.6 GS_ENCRYPTED_PROC

GS_ENCRYPTED_PROC provides information such as the parameters of encrypted functions and stored procedure functions, original data type of return values, and encrypted columns.

Table 7-21 GS_ENCRYPTED_PROC columns

Name	Type	Description
oid	oid	Row identifier (hidden column).
func_id	oid	OID of the function, corresponding to the row identifier (OID) in the **PG_PROC** system catalog.
prorettype_orig	integer	Original data type of the return value
last_change	timestamp without time zone	Last modification time of the encrypted function.
proargcachedcol	oidvector	OID of the encrypted column corresponding to the **INPUT** parameter of the function, corresponding to the row identifier (OID) in the **GS_ENCRYPTED_COLUMNS** system catalog.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
proallargtypes_orig	oid[]	Original data type of all function parameters

7.2.5 Communications

7.2.5.1 PGXC_NODE

PGXC_NODE records information about cluster nodes.

NOTICE

- PGXC_NODE stores information about database instance nodes. The PGXC_NODE system catalog has specific meanings only on CNs and its data is valid and correct.
- In the result of querying the PGXC_NODE system catalog on a DN, only the **node_id** column is meaningful. Other columns are meaningless and invalid.

Table 7-22 PGXC_NODE columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
node_name	name	Node name.
node_type	"char"	Node type. The **R**, **M**, and **O** types are unique to the standby cluster for DR. • **C**: CN. • **D**: DN • **S**: standby node • **R**: read-only CN of the standby cluster. • **M**: static primary DN of the standby cluster. • **O**: other data nodes in the standby cluster.
node_port	integer	Port number of the node.
node_host	name	Host name or IP address of a node. (If a virtual IP address is configured, its value is a virtual IP address.)
node_port1	integer	Port number of a replication node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
node_host1	name	Host name or IP address of a replication node. (If a virtual IP address is configured, its value is a virtual IP address.)
hostis_primary	Boolean	Specifies whether a primary/standby switchover occurs on the current node. • **t** (true): yes • **f** (false): no
nodeis_primary	Boolean	Specifies whether the current node is preferred to execute non-query operations in the **replication** table. • **t** (true): yes • **f** (false): no
nodeis_preferred	Boolean	Specifies whether the current node is preferred to execute queries in the **replication** table. • **t** (true): yes • **f** (false): no
node_id	integer	Node identifier. The value is obtained by calculating the value of **node_name** using the hash function.
sctp_port	integer	Port used by the TCP proxy communications library of the primary node to listen to the data channel. (The SCTP communications library is no longer supported in the current version.)
control_port	integer	Port used by the TCP proxy communications library of the primary node to listen to the control channel.
sctp_port1	integer	Port used by the TCP proxy communications library of the standby node to listen to the data channel. (The SCTP communications library is no longer supported in the current version.)
control_port1	integer	Port used by the TCP proxy communications library of the standby node to listen to the control channel.
nodeis_central	Boolean	Specifies whether the current node is a central control node. It is used only for CNs and is invalid for DNs. • **t** (true): yes • **f** (false): no

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
nodeis_active	Boolean	Specifies whether the current node is normal. It is used to mark whether the CN is removed and is invalid for DNs. • **t** (true): yes • **f** (false): no
cluster_name	text	Specifies the cluster name.
slice_order	integer	Sequence number of the shard that a node belongs to.

7.2.6 Ledger Database

7.2.6.1 GS_GLOBAL_CHAIN

GS_GLOBAL_CHAIN records information about modification operations performed by users on the tamper-proof user table. Each record corresponds to a table-level modification operation. Users with the AUDITADMIN permission can query this system catalog, but no user is allowed to modify this system catalog.

Table 7-23 GS_GLOBAL_CHAIN columns

Name	Type	Description
blocknum	bigint	Block number, which is the sequence number of the current user operation recorded in the ledger.
dbname	name	Name of the database to which the modified tamper-proof user table belongs.
username	name	Username, which is the name of the user who performs the operation of modifying the user table.
starttime	timestamp with time zone	Time when a user performs an operation.
relid	oid	OID of the modified tamper-proof user table
relnsp	name	Schema name, which is the name of the schema to which the modified tamper-proof user table belongs.
relname	name	User table name, which is the name of the modified tamper-proof user table.
relhash	hash16	Table-level hash change amount generated by operations.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
globalhash	hash32	Global digest, which is calculated based on the information of the current row and **globalhash** of the previous row. It connects the entire table to verify the integrity of GS_GLOBAL_CHAIN data.
txcommand	text	SQL statement whose operations are recorded.

7.2.7 SPM

7.2.7.1 GS_SPM_SQL

GS_SPM_SQL is a system catalog in the SPM feature. Currently, this feature is not supported in the distributed system. It is a system catalog used to store SPM information. Users with the SYSADMIN permission can read the system catalog, but only the initial user can write data to the system catalog.

Table 7-24 GS_SPM_SQL columns

Name	Type	Description
sql_namespace	oid	Schema OID.
sql_hash	bigint	Unique ID of the SQL statement in the SPM.
sql_text	text	SQL text string.
param_num	integer	Number of parameters required for SQL execution.
user	oid	OID of the login user.
creation_time	timestamp with time zone	Time when a record is created.

7.2.7.2 GS_SPM_PARAM

GS_SPM_PARAM is a system catalog in the SPM feature. Currently, this feature is not supported in the distributed system. It is a system catalog used to store SQL parameters. Each SQL statement stores only one group of parameters. Users with the SYSADMIN permission can read the system catalog, but only the initial user can write data to the system catalog.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-25 GS_SPM_PARAM columns

Name	Type	Description
sql_namespace	oid	Schema OID
sql_hash	bigint	Unique ID of the SQL statement in the SPM.
position	integer	Position index of a parameter in the SQL statement, starting from 0.
datatype	integer	OID of the parameter type.
datatype_string	text	Character string of a parameter type.
value_string	text	Character string of a parameter value.
is_null	Boolean	Determines whether a parameter value is **NULL**.
hash_value	bigint	Hash value of a parameter value.
user	oid	User who creates the record.
creation_time	timestamp with time zone	Time when a record is created.
namespace_oids	text	OID list in the current **search_path**. The OIDs must meet the following requirements: • OIDs of the schema on which the user has permissions. • OIDs of the existing schemas. • OIDs of the schemas where the non-local temporary tables are located.

7.2.7.3 GS_SPM_BASELINE

GS_SPM_BASELINE is a system catalog in the SPM feature. Currently, this feature is not supported in the distributed system. It is a system catalog used to store baseline information. Users with the SYSADMIN permission can read the system catalog, but only the initial user can write data to the system catalog.

Table 7-26 GS_SPM_BASELINE columns

Name	Type	Description
sql_namespace	oid	Schema OID.
sql_hash	bigint	Unique ID of the SQL statement in the SPM.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
plan_hash	bigint	Unique ID of a plan in the current SQL statement.
outline	text	Outline text, which can be used to fix a group of hints of the current plan.
cost	double precision	Total plan cost.
user	oid	User who creates the baseline.
status	integer	Baseline status. The options are as follows: ● **0** (UNACC): indicates that the plan is not accepted. ● **1** (ACC): indicates that the plan has been accepted. ● **2** (FIXED): indicates a special ACC plan. The matching priority of this plan is higher than that of other ACC plan.
source	text	Baseline source.
gplan	Boolean	Determines whether the plan is a gplan.
creation_time	timestamp with time zone	Time when the baseline is created.
last_used_time	timestamp with time zone	Last time when a plan was used.
modification_time	timestamp with time zone	Time when the baseline is modified.
jump_intercept_cnt	bigint	Number of baseline interception plan changes.
invalid	Boolean	Specifies whether the current baseline is invalid.
namespace_oids	text	OID list in the current **search_path**. The OIDs must meet the following requirements: ● OIDs of the schema on which the user has permissions. ● OIDs of the existing schemas. ● OIDs of the schemas where the non-local temporary tables are located.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.7.4 GS_SPM_EVOLUTION

GS_SPM_EVOLUTION is a system catalog in the SPM feature. Currently, this feature is not supported in the distributed system. It is a system catalog used to store plan evolution result. Users with the SYSADMIN permission can read the system catalog, but only the initial user can write data to the system catalog.

Table 7-27 GS_SPM_EVOLUTION columns

Name	Type	Description
sql_namespace	oid	Schema OID.
sql_hash	bigint	Unique ID of the SQL statement in the SPM.
plan_hash	bigint	Plan ID.
better	Boolean	Determines whether it is a positive evolution. ● **t** indicates positive evolution. ● **f** indicates negative evolution.
refer_plan	bigint	Plan hash used as a reference for report generation.
status	integer	Determines whether exceptions occur during the evolution. ● The value **0** indicates that no exception occurs. ● The value **1** indicates that an exception occurs.
reason	text	Content of the evolution report.
user	oid	User who generates the evolution result.
creation_time	timestamp with time zone	Time when the evolution result is created.

7.2.7.5 GS_SPM_ID_HASH_JOIN

GS_SPM_ID_HASH_JOIN is a system catalog in the SPM feature. Currently, this feature is not supported in the distributed system. The system catalog stores the relationship between unique_sql_id and sql_hash. Users with the SYSADMIN permission can read the system catalog, but only the initial user can write data to the system catalog.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-28 GS_SPM_ID_HASH_JOIN columns

Name	Type	Description
unique_sql_id	bigint	Unique ID of an SQL statement in the database.
sql_hash	bigint	Unique ID of the SQL statement in the SPM.

7.2.8 AI

7.2.8.1 GS_MODEL_WAREHOUSE

GS_MODEL_WAREHOUSE stores AI engine training models, including the models and detailed description of the training process.

Table 7-29 GS_MODEL_WAREHOUSE columns

Name	Type	Description
oid	OID	Implicit column.
modelname	Name	Model name, which is unique.
modelowner	OID	OID of the model owner.
createtime	Timestamp without time zone	Time when the model was created.
processedtuples	Integer	Number of tuples involved in training.
discardedtuples	Integer	Number of unqualified tuples not used in training.
preprocesstime	Real	Data preprocessing duration.
exectime	Real	Training duration.
iterations	Integer	Number of iterations.
outputtype	OID	OID of the output data type.
modeltype	Text	AI operator type.
query	Text	Query statement executed to create the model.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
modeldata	Bytea	Stored binary model information.
weight	Real[]	Currently, this column applies only to GD operator models.
hyperparametersnames	Text[]	Name of an involved hyperparameter.
hyperparametersvalues	Text[]	Value of the hyperparameter.
hyperparametersoids	OID[]	OID of the data type corresponding to the hyperparameter.
coefnames	Text[]	Name of a model parameter.
coefvalues	Text[]	Value of the model parameter.
coefoids	OID[]	OID of the data type corresponding to the model parameter.
trainingscoresname	Text[]	Name of the method used to measure the model performance.
trainingscoresvalue	Real[]	Value used to measure the model performance.
modeldescribe	Text[]	Model description.

7.2.8.2 GS_OPT_MODEL

GS_OPT_MODEL is a data table used when the AI engine is enabled to predict the planned time. It records the configurations, training results, features, system functions, and training history of machine learning models.

NOTE

This system catalog is provided in distributed mode, but AI capabilities remain unavailable.

7.2.8.3 GS_ABO_MODEL_STATISTIC

GS_ABO_MODEL_STATISTIC stores the metadata, names, and operators of the models estimated by ABO based on feedback cardinality.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-30 GS_ABO_MODEL_STATISTIC columns

Name	Type	Description
oid	OID	Implicit column.
modelname	Name	Model name, which is unique.
createtime	Timestamp without time zone	Time when the model was created or updated.
groupkey	OID	Operator hash of the model.
joinkey	OID	Hash of the model's operator connection shape.
fixed_selectivity	Double precision	Fixed selectivity for operators that do not contain numeric conditions.
avg_qerror	Double precision	Average qerror of the model.
max_qerror	Double precision	Maximum qerror of the model.
feedback_num	Integer	Training data volume of the model.
status	Integer	Model status. • **0**: The model is ineffective. • **1**: The model is effective. • **2**: The model is to be updated. • **3**: The model is blacklisted. • **4**: The model will be replaced by the default model when it is used next time.
train_count	Integer	Total number of model training times.
train_failure_count	Integer	Number of model training failures.
prediction_count	Integer	Total number of model prediction times.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
prediction_failure_count	Integer	Number of model prediction failures.
rtable_ids	oidvector	Sequence of including the model conditions into the base table.
attnums	int2vector	Sequence of columns involved in model input conditions.

7.2.8.4 GS_ABO_HISTORY

GS_ABO_HISTORY stores the usage statistics of ABO (cardinality estimation feedback) models.

Table 7-31 GS_ABO_HISTORY columns

Name	Type	Description
sql_id	Bigint	SQL ID.
user_id	OID	User ID.
acm_sql_status	Tinyint	Specifies whether it is blacklisted. • **0**: not blacklisted. • **1**: blacklisted. • **2**: not blacklisted and being executed.
explore_count	Bigint	Number of exploration times for the plan used for the blacklist mechanism.
execution_time	Double precision	Execution time of this query, in milliseconds.
usage_count	Bigint	Number of times that the feedback cardinality is used.
create_time	Timestamp without time zone	Time when this record was created.

7.2.9 Audit

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.9.1 GS_AUDITING_POLICY

GS_AUDITING_POLICY records main information for each unified audit policy. Each record corresponds to an audit policy. By default, only users with the SYSADMIN or POLADMIN permission can access it. Common users can access it only after being authorized.

Table 7-32 GS_AUDITING_POLICY columns

Name	Type	Description
oid	OID	Row identifier (implicit attribute, which must be explicitly selected).
polname	Name	Policy name, which must be unique.
polcomments	Name	Policy description, which is represented by the COMMENTS keyword.
modifydate	Timestamp without time zone	Timestamp when the policy was created or last modified.
polenabled	Boolean	Specifies whether the policy is enabled. • **t** (true): The policy is enabled. • **f** (false): The policy is disabled.

7.2.9.2 GS_AUDITING_POLICY_ACCESS

GS_AUDITING_POLICY_ACCESS records unified audit information for ACCESS database operations. By default, only users with the SYSADMIN or POLADMIN permission can access it. Common users can access it only after being authorized.

Table 7-33 GS_AUDITING_POLICY_ACCESS columns

Name	Type	Description
oid	OID	Row identifier (implicit attribute, which must be explicitly selected).
accesstype	Name	Type related to ACCESS, for example, SELECT, INSERT, or DELETE.
labelname	Name	Resource label name, corresponding to **polname** in the **GS_AUDITING_POLICY** system catalog.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
policyoid	OID	OID of the audit policy, corresponding to **oid** in the GS_AUDITING_POLICY system catalog.
modifydate	Timestamp without time zone	Timestamp of creation or latest modification.

7.2.9.3 GS_AUDITING_POLICY_FILTERS

GS_AUDITING_POLICY_FILTERS records filter policies for unified audit. Each record corresponds to an audit policy. By default, only users with the SYSADMIN or POLADMIN permission can access it. Common users can access it only after being authorized.

Table 7-34 GS_AUDITING_POLICY_FILTERS columns

Name	Type	Description
oid	OID	Row identifier (implicit attribute, which must be explicitly selected).
filtertype	Name	Filter type. Currently, the value can only be **logical_expr**.
labelname	Name	Name. Currently, the value can only be **logical_expr**.
policyoid	OID	OID of the audit policy, corresponding to **oid** in the GS_AUDITING_POLICY system catalog.
modifydate	Timestamp without time zone	Timestamp of creation or latest modification.
logicaloperator	Text	Logical character string of a filter criterion.

7.2.9.4 GS_AUDITING_POLICY_PRIVILEGES

GS_AUDITING_POLICY_PRIVILEGES records unified audit information for PRIVILEGES database operations. Each record corresponds to an audit policy. By default, only users with the SYSADMIN or POLADMIN permission can access it. Common users can access it only after being authorized.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-35 GS_AUDITING_POLICY_PRIVI columns

Name	Type	Description
oid	OID	Row identifier (implicit attribute, which must be explicitly selected).
privilegetype	Name	Type related to PRIVILEGES, for example, CREATE, ALTER, or DROP.
labelname	Name	Resource label name, corresponding to **polname** in the GS_AUDITING_POLICY system catalog.
policyoid	OID	OID of the audit policy, corresponding to **oid** in the GS_AUDITING_POLICY system catalog.
modifydate	Timestamp without time zone	Timestamp of creation or latest modification.

7.2.10 User and Permission Management

7.2.10.1 GS_DB_PRIVILEGE

GS_DB_PRIVILEGE records grant information for ANY permissions, with each entry representing a distinct grant.

Table 7-36 GS_DB_PRIVILEGE columns

Name	Type	Description
oid	OID	Row identifier (implicit attribute, which must be explicitly selected).
roleid	OID	User ID.
privilege_type	Text	ANY permissions of the user. For details about such permissions, see Table 1-402.
admin_option	Boolean	Specifies whether the ANY permissions recorded in the **privilege_type** column can be re-granted. • **t**: yes. • **f**: no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.10.2 PG_DB_ROLE_SETTING

PG_DB_ROLE_SETTING stores default values for the configuration items bound to each role and data entry during database runtime.

Table 7-37 PG_DB_ROLE_SETTING columns

Name	Type	Description
setdatabase	OID	Database corresponding to the configuration item (**0** if no database is specified).
setrole	OID	Role corresponding to the configuration item (**0** if no role is specified).
setconfig	Text[]	Default value of the configuration item during runtime.

7.2.10.3 PG_DEFAULT_ACL

PG_DEFAULT_ACL stores the initial permissions assigned to newly created objects.

Table 7-38 PG_DEFAULT_ACL columns

Name	Type	Description
oid	OID	Row identifier (implicit attribute, which must be explicitly selected).
defaclrole	OID	ID of a role associated with the permission.
defaclnamespace	OID	Namespace associated with the permission (**0** if no namespace is associated).
defaclobjtype	"Char"	Object type of the permission. • **r**: table or view. • **S**: sequence. • **f**: function. • **T**: type. • **K**: client master key. • **k**: column encryption key.
defaclacl	ACLItem[]	Access permission that this type of object should have on creation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.10.4 PG_RLSPOLICY

PG_RLSPOLICY stores row-level security policies.

Table 7-39 PG_RLSPOLICY columns

Name	Type	Description
oid	OID	Row identifier (implicit attribute, which must be explicitly selected).
polname	Name	Name of a row-level security policy.
polrelid	OID	OID of the table object to which the row-level security policy is applied.
polcmd	"Char"	SQL operations affected by the row-level security policy.
polpermissive	Boolean	Attribute of the row-level security policy. • **t**: expression concatenation using the OR condition. • **f**: expression concatenation using the AND condition.
polroles	OID[]	OIDs of users affected by the row-level security policy (if unspecified, all users are affected).
polqual	pg_node_tree	Expression of the row-level security policy.

7.2.10.5 PG_SECLABEL

PG_SECLABEL stores security labels for database objects.

PG_SHSECLABEL serves a similar purpose, but it stores security labels for database objects shared within the same database cluster.

Table 7-40 PG_SECLABEL columns

Name	Type	Reference	Description
objoid	OID	Any OID	OID of the object to which the security label belongs.
classoid	OID	**PG_CLASS**.oid	OID of the system catalog that contains the current object.
objsubid	Integer	-	SN of a column in this object.
provider	Text	-	Provider associated with the current label.
label	Text	-	Security label applied to the object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.10.6 PG_SHSECLABEL

PG_SHSECLABEL stores security labels for shared database objects. Security labels can be manipulated using the **SECURITY LABEL** command.

PG_SECLABELS provides an easier way to view security labels.

PG_SECLABEL serves a similar purpose, but it stores security labels for objects within a single database.

Unlike most system catalogs, PG_SHSECLABEL is shared across all databases within a cluster: there is only one PG_SHSECLABEL per database cluster, not one per database.

Table 7-41 PG_SHSECLABEL columns

Name	Type	Reference	Description
objoid	OID	Any OID	OID of the object to which the security label belongs.
classoid	OID	**PG_CLASS**.oid	OID of the system catalog that contains the current object.
provider	Text	-	Provider associated with the current label.
label	Text	-	Security label applied to the current object.

7.2.10.7 PG_USER_MAPPING

PG_USER_MAPPING stores the mappings from local users to remote.

This system catalog is accessible only to users with the SYSADMIN permission. Common users can query the **PG_USER_MAPPINGS** view.

Table 7-42 PG_USER_MAPPING columns

Name	Type	Reference	Description
oid	OID	-	Row identifier (implicit attribute, which must be explicitly selected).
umuser	OID	**PG_AUTHID**.oid	OID of the local user being mapped (**0** if the user mapping is public).
umserver	OID	**PG_FOREIGN_SERVER**.oid	OID of the foreign server that contains this mapping.
umoptions	Text[]	-	User mapping options, as "keyword=value" strings.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.10.8 PG_USER_STATUS

PG_USER_STATUS stores the states of users who access databases. This system catalog is accessible only to users with the SYSADMIN permission.

Table 7-43 PG_USER_STATUS columns

Name	Type	Description
oid	OID	Row identifier (implicit attribute, which must be explicitly selected).
roloid	OID	Role ID.
failcount	Integer	Number of failed attempts.
locktime	Timestamp with time zone	Lock time. By default, this parameter displays the creation date of the role. If the role is locked by an administrator or due to exceeding the failed login attempt threshold, this parameter reflects the date when the role was locked.
rolstatus	Smallint	Role state. ● **0**: normal. ● **1**: locked for a specific period of time due to exceeding the failed login attempt threshold. ● **2**: locked by an administrator. ● **3**: not locked, despite exceeding the failed login attempt threshold.
permspace	Bigint	Size of the permanent table storage space used by the role.
tempspace	Bigint	Size of the temporary table storage space used by the role.
password expired	Smallint	Expiration status of its password. ● **0**: not expired. ● **1**: expired.
passwordlifetime	Smallint	Password validity period, in days. When left unset, it defaults to **0**, meaning that the password remains valid permanently.

7.2.11 Connection and Authentication

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.11.1 PG_AUTHID

PG_AUTHID stores information about database authentication identifiers (roles). The concept of "users" is encompassed within the "roles" framework, where a user is essentially a role with the **rolcanlogin** attribute set. Any role, whether its **rolcanlogin** is set or not, can include other roles as members.

There is only one PG_AUTHID within a cluster, not one per database. This system catalog is accessible only to users with the SYSADMIN permission.

Table 7-44 PG_AUTHID columns

Name	Type	Description
oid	OID	Row identifier (implicit attribute, which must be explicitly selected).
rolname	Name	Role name.
rolsuper	Boolean	Specifies whether the role is the initial system administrator with the highest permission. • **t** (true): yes. • **f** (false): no.
rolinherit	Boolean	Specifies whether the role automatically inherits permissions from its owning role. • **t** (true): yes. • **f** (false): no.
rolcreaterole	Boolean	Specifies whether the role can create additional roles. • **t** (true): yes. • **f** (false): no.
rolcreatedb	Boolean	Specifies whether the role can create databases. • **t** (true): yes. • **f** (false): no.
rolcatupdate	Boolean	Specifies whether the role can directly update system catalogs. Only the initial system administrator (**usesysid=10**) has this permission; other roles cannot acquire this permission. • **t** (true): yes. • **f** (false): no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
rolcanlogin	Boolean	Specifies whether the role can log in, that is, whether it can be assigned a session authorization identifier. • **t** (true): yes. • **f** (false): no.
rolreplication	Boolean	Specifies whether the role has the replication permission. • **t** (true): yes. • **f** (false): no.
rolauditadmin	Boolean	Specifies whether the role has the AUDITADMIN permission. • **t** (true): yes. • **f** (false): no.
rolsystemadmin	Boolean	Specifies whether the role has the SYSADMIN permission. • **t** (true): yes. • **f** (false): no.
rolconnlimit	Integer	Maximum number of concurrent connections that the role can establish (if the role can log in). A value of **-1** indicates no limit.
rolpassword	Text	Password ciphertext (**NULL** if no password is set).
rolvalidbegin	Timestamp with time zone	Start time of the account validity period (**NULL** if no start time is defined).
rolvaliduntil	Timestamp with time zone	End time of the account validity period (**NULL** if no end time is defined).
rolrespool	Name	Resource pool available to the user.
roluseft	Boolean	Specifies whether the role can manipulate foreign tables. • **t** (true): yes. • **f** (false): no.
rolparentid	OID	OID of the group to which the user belongs.
roltabspace	Text	Maximum space allowed for the user's data tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
rolkind	"Char"	Special user type. • **n**: common user. • **p**: permanent user.
rolnodegroup	OID	OID of the node group associated with a role. If no node group is associated, the value is **0**.
roltempspace	Text	Maximum space allowed for the user's temporary tables, in KB.
rolspillspace	Text	Maximum spill space allowed for job execution by the user, in KB.
rolexcpdata	Text	Query rules that can be configured by the user. (This column remains unused currently.)
rolmonitoradmin	Boolean	Specifies whether the role has the MONADMIN permission. • **t** (true): yes. • **f** (false): no.
roloperatoradmin	Boolean	Specifies whether the role has the OPRADMIN permission. • **t** (true): yes. • **f** (false): no.
rolpolicyadmin	Boolean	Specifies whether the role has the POLADMIN permission. • **t** (true): yes. • **f** (false): no.

7.2.11.2 PG_AUTH_HISTORY

PG_AUTH_HISTORY records the authentication history of roles. This system catalog is accessible only to users with the SYSADMIN permission.

Table 7-45 PG_AUTH_HISTORY columns

Name	Type	Description
oid	OID	Row identifier (implicit attribute, which must be explicitly selected).
roloid	OID	Role ID.
passwordtime	Timestamp with time zone	Time of password creation or modification.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
rolpassword	Text	Ciphertext of the role password. The encryption mode is determined by the GUC parameter **password_encryption_type**.

7.2.11.3 PG_AUTH_MEMBERS

PG_AUTH_MEMBERS stores and displays membership between roles.

Table 7-46 PG_AUTH_MEMBERS columns

Name	Type	Description
roleid	OID	ID of the parent role that owns a member.
member	OID	Role ID of the member included in the given parent role.
grantor	OID	ID of the role that grants this membership.
admin_option	Boolean	Specifies whether the member can grant the parent role to others.

7.2.12 Dynamic Data Masking

7.2.12.1 GS_MASKING_POLICY

GS_MASKING_POLICY records the main information about dynamic data masking policies. Each record corresponds to a masking policy. Only the SYSADMIN or POLADMIN can access this system catalog.

Table 7-47 GS_MASKING_POLICY columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
polname	name	Policy name, which must be unique.
polcomments	name	Policy description, which is represented by the COMMENTS keyword.
modifydate	timestamp without time zone	Latest timestamp when a policy is created or modified.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
polenabled	boolean	Specifies whether to enable a policy. • **t** (true): enabled. • **f** (false): disabled

7.2.12.2 GS_MASKING_POLICY_ACTIONS

GS_MASKING_POLICY_ACTIONS records the masking actions of a masking policy in the dynamic data masking policies. One masking policy corresponds to one or more rows of records in the catalog. Only the SYSADMIN or POLADMIN can access this system catalog.

Table 7-48 GS_MASKING_POLICY_ACTIONS columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
actiontype	name	Name of a masking function used by a masking policy
actparams	name	Parameter information specified in a masking function
actlabelname	name	Name of a masked label
policyoid	oid	OID of a masking policy to which a record belongs, corresponding to the OID in **GS_MASKING_POLICY**.
actmodifydate	timestamp without time zone	Latest timestamp when a record is created or modified

7.2.12.3 GS_MASKING_POLICY_FILTERS

GS_MASKING_POLICY_FILTERS records the user filtering criteria corresponding to the dynamic data masking policies. The corresponding masking policy takes effect only when the user information meets the filter criteria. Only the SYSADMIN or POLADMIN can access this system catalog.

Table 7-49 GS_MASKING_POLICY_FILTERS columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
filtertype	name	Filter type. Currently, the value is **logical_expr**.
filterlabelname	name	Filter range. Currently, the value is **logical_expr**.
policyoid	oid	OID of a masking policy to which a record belongs, corresponding to the OID in **GS_MASKING_POLICY**.
modifydate	timestamp without time zone	Latest timestamp when a user filter criterion is created or modified
logicaloperator	text	Polish notation of a filter criterion

7.2.13 DATABASE LINK

7.2.13.1 GS_DATABASE_LINK

GS_DATABASE_LINK stores database link information, mainly recording detailed information about database links. Only SYSADMIN can read this system catalog.

Table 7-50 GS_DATABASE_LINK columns

Name	Type	Description
oid	oid	Unique ID of the current database link object (hidden attribute, which must be specified).
dlname	name	Name of the current database link.
dlowner	oid	ID of the owner of the current database link. If the owner is **public**, the value is **0**.
dlfdw	oid	OID of the foreign data wrapper of the current database link.
dlcreater	oid	ID of the creator of the current database link.
options	text[]	Current database link connection information in the format of "keyword=value".
useroptions	text[]	User information used by the current database link to connect to the remote end, in the format of "keyword=value".
dlacl	aclitem[]	Current database link access permission.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.14 Materialized Views

7.2.14.1 GS_MATVIEW

GS_MATVIEW provides information about each materialized view in the database.

Table 7-51 GS_MATVIEW columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
matviewid	oid	OID of a materialized view.
mapid	oid	OID of a map table joined with a materialized view. Each map table corresponds to one materialized view. If a complete-refresh materialized view does not correspond to a map table, the value of this column is **0**.
ivm	boolean	Type of a materialized view. The value **t** indicates a fast-refresh materialized view, and the value **f** indicates a complete-refresh materialized view.
needrefresh	boolean	Reserved column.
refreshtime	timestamp without time zone	Last time when a materialized view was refreshed. If the materialized view is not refreshed, the value is null. This column is maintained only for fast-refresh materialized views on DNs. In other cases, the value is null.

7.2.14.2 GS_MATVIEW_DEPENDENCY

GS_MATVIEW_DEPENDENCY provides association information about each fast-refresh materialized view, base table, and Mlog table in the database. The Mlog table corresponding to the base table does not exist in the complete-refresh materialized view. Therefore, no record is written into the Mlog table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-52 GS_MATVIEW_DEPENDENCY columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
matviewid	oid	OID of a materialized view.
relid	oid	OID of a base table of a materialized view.
mlogid	oid	OID of a Mlog table which is the log table of a materialized view. Each Mlog table corresponds to one base table.
mxmin	integer	Reserved column.

7.2.15 Logical Decoding

In the logical decoding independent multi-version dictionary, the logical decoding dictionary catalog used is the system catalog. Logical decoding system catalogs are instance-level system catalogs. Only administrators can view and modify system catalogs.

7.2.15.1 GS_LOGICAL_ATTRDEF

GS_LOGICAL_ATTRDEF records the default values of storage columns required for logical decoding.

Table 7-53 GS_LOGICAL_ATTRDEF columns

Column	Type	Description
adid	oid	Identifier of an attribute descriptor.
adrelid	oid	OID of the table.
adnum	smallint	Attribute number.
csnmin	bigint	CSN when a tuple is inserted.
csnmax	bigint	CSN when a tuple is updated or deleted.
createtime	timestamp with time zone	Timestamp when a tuple is inserted.
originid	integer	Source identifier of a tuple.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
dbnode	oid	Database OID.
adbin	pg_node_tree	Expression of the default value.
adgencol	"char"	Specifies whether a column is a GENERATED column.

7.2.15.2 GS_LOGICAL_ATTRIBUTE

GS_LOGICAL_ATTRIBUTE records information about catalog columns required for logical decoding.

Table 7-54 GS_LOGICAL_ATTRIBUTE columns

Column	Type	Description
attrelid	oid	RELATION OID that contains the attribute.
attname	name	Attribute name.
atttypid	oid	OID of the data type of the attribute.
attlen	smallint	Attribute length.
attnum	smallint	Attribute ID, which uniquely identifies an attribute in a relationship.
atthasdef	boolean	Specifies whether an attribute has a default value.
csnmin	bigint	CSN when a tuple is inserted.
csnmax	bigint	CSN when a tuple is updated or deleted.
createtime	timestamp with time zone	Timestamp when a tuple is inserted.
originid	integer	Source identifier of a tuple.
dbnode	oid	Database OID.
attndims	integer	If the value is of the array type, the value is the declared dimension number. Otherwise, the value is zero.
attcacheoff	integer	Used for the byte offset of the function cache attribute in the heap tuple.
attbyval	boolean	Specifies whether an attribute is passed by value.
attstorage	"char"	Attribute storage mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
attalign	"char"	Alignment mode of an attribute.
attisdropped	boolean	Specifies whether the data is deleted (logically invisible).
attinitdefval	bytea	Default value expression of a storage attribute.
attcollation	oid	Copy of **attcollation** in **PG_ATTRIBUTE** of this column's type.
atttypmod	integer	Copy of **atttypmod** in **PG_ATTRIBUTE** of this column's type.

7.2.15.3 GS_LOGICAL_AUTHID

GS_LOGICAL_AUTHID records user information required for logical decoding.

Table 7-55 GS_LOGICAL_AUTHID columns

Column	Type	Description
rolid	oid	User ID.
rolname	name	Username.
csnmin	bigint	CSN when a tuple is inserted.
csnmax	bigint	CSN when a tuple is updated or deleted.
createtime	timestamp with time zone	Timestamp when a tuple is inserted.
originid	integer	Source identifier of a tuple.
rolkind	"char"	User type.
pdbid	oid	Tenant ID.

7.2.15.4 GS_LOGICAL_CLASS

GS_LOGICAL_CLASS records information about database objects required for logical decoding and the relationships between the database objects.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-56 GS_LOGICAL_CLASS columns

Column	Type	Description
relname	name	Table name.
relnamespace	oid	Namespace OID.
relpersistence	"char"	Catalog durability (for example, **p** indicates a permanent table).
relkind	"char"	Table type (for example, **r** indicates a relationship).
relnatts	smallint	Number of columns in the table.
relhaspkey	bool	Specifies whether the table has (or has ever had) a primary key index.
relcmprs	tinyint	Compression attribute of a row in a table.
relreplident	"char"	Replica identity replication mode.
spacenode	oid	Tablespace OID.
dbnode	oid	Database OID.
relnode	oid	Identifier of the physical storage file corresponding to the table.
createtime	timestamp with time zone	Timestamp when a tuple is inserted.
csnmin	bigint	CSN when a tuple is inserted.
csnmax	bigint	CSN when a tuple is updated or deleted.
originid	integer	Source identifier of a tuple.
relid	oid	Relationship identifier (table OID).
relowner	oid	Identifier of the table owner.
reltoastrelid	oid	OID of the TOAST table (if any).
relhasoids	boolean	Specifies whether the table has an object identifier (OID) column.
relam	oid	Identifier of the indexing method used by the table.
reloptions	text[]	Additional options for saving the table, stored in text format.
relbucket	oid	Identifier of the hash partitioning bucket to which the table belongs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
parttype	"char"	Specifies whether the table is a partitioned table. **'p'** indicates that it is a partitioned table, while **'n'** indicates that it is not.

7.2.15.5 GS_LOGICAL_CONSTRAINT

GS_LOGICAL_CONSTRAINT records CHECK, PRIMARY KEY, and UNIQUE constraints on tables required for logical decoding.

Table 7-57 GS_LOGICAL_CONSTRAINT columns

Name	Type	Description
conname	name	Constraint name (not necessarily unique).
connamespace	oid	OID of a namespace that contains the constraint.
contype	"char"	• **c**: CHECK constraint. • **p**: PRIMARY KEY constraint. • **u**: UNIQUE constraint. • **t**: trigger constraint. • **x**: mutual exclusion constraint. • **f**: FOREIGN KEY constraint. • **s**: clustering constraint. • **i**: invalid constraint.
condeferrable	boolean	Specifies whether the constraint is deferrable. • **true**: yes. • **false**: no.
condeferred	boolean	Specifies whether the constraint can be deferrable by default. • **true**: yes. • **false**: no.
convalidated	boolean	Specifies whether the constraint is valid. Currently, it can be set to **false** only for FOREIGN KEY and CHECK constraints. • **true**: valid. • **false**: invalid.
conrelid	oid	Table containing this constraint (**0** if it is not a table constraint).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
contypid	oid	Domain containing this constraint (**0** if it is not a domain constraint).
conindid	oid	ID of an index joined with the constraint.
confrelid	oid	Referenced table if this is a FOREIGN KEY constraint. Otherwise, the value is **0**.
confupdtype	"char"	Foreign key update action code. ● **a**: no action. ● **r**: restriction. ● **c**: cascading. ● **n**: The parameter is set to **null**. ● **d**: The default value is used.
confdeltype	"char"	Foreign key deletion action code. ● **a**: no action. ● **r**: restriction. ● **c**: cascading. ● **n**: The parameter is set to **null**. ● **d**: The default value is used.
confmatchtype	"char"	Foreign key match type. ● **f**: full match. ● **p**: partial match. ● **u**: unspecified (The **NULL** value can be matched if **f** is specified.)
conislocal	boolean	Specifies whether the constraint is defined locally for the relationship. ● **true**: yes ● **false**: no.
coninhcount	integer	Number of direct inheritance parent tables that this constraint has. When the value is not **0**, the constraint cannot be deleted or renamed.
connoinherit	boolean	Specifies whether the constraint can be inherited. ● **true**: yes. ● **false**: no.
consoft	boolean	Specifies whether the column indicates an informational constraint. ● **true**: yes ● **false**: no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
conopt	boolean	Specifies whether you can use the informational constraint to optimize the execution plan. • **true**: yes. • **false**: no.
csnmin	bigint	CSN when a tuple is inserted.
csnmax	bigint	CSN when a tuple is updated or deleted.
createtime	timestamp with time zone	Timestamp when a tuple is inserted.
originid	integer	Source identifier of a tuple.
dbnode	oid	Database OID.
constroid	oid	Row identifier.
conkey	smallint[]	Column list of the constrained control if this column is a table constraint.
confkey	smallint[]	List of referenced columns if this column is a foreign key.
conpfeqop	oid[]	ID list of the equality operators for PK = FK comparisons if this column is a foreign key.
conppeqop	oid[]	ID list of the equality operators for PK = PK comparisons if this column is a foreign key.
conffeqop	oid[]	ID list of the equality operators for FK = FK comparisons if this column is a foreign key.
conexclop	oid[]	ID list of the per-column exclusion operators if this column is an exclusion constraint.
consrc	text	Readable representation of the expression if this column is a CHECK constraint.
conincluding	smallint[]	Not for constraint, but will be included in the attribute column of **INDEX**.

7.2.15.6 GS_LOGICAL_DATABASE

GS_LOGICAL_DATABASE records information about available databases.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-58 GS_LOGICAL_DATABASE columns

Column	Type	Description
datname	name	Database name.
datid	oid	Database OID.
csnmin	bigint	CSN when a tuple is inserted.
csnmax	bigint	CSN when a tuple is updated or deleted.
createtime	timestamp with time zone	Timestamp when a tuple is inserted.
originid	integer	Source identifier of a tuple.
datdba	oid	Owner of the database, usually the user who created it.
encoding	integer	Character encoding of the database.
datcollate	name	Character type used by the database.
datctype	name	Sequence used by the database.
dattablespace	oid	Default tablespace of the database.
datcompatibility	name	Database compatibility mode. Currently, five compatibility modes are supported: ORA, MYSQL, TD, M, and PG, indicating compatibility with Oracle Database, MySQL, Teradata, M-compatible databases, and PostgreSQL, respectively.
dattimezone	name	Time zone of the database. The default is PRC.

7.2.15.7 GS_LOGICAL_DICTIONARY

GS_LOGICAL_DICTIONARY records logical decoding data dictionary information.

Table 7-59 GS_LOGICAL_DICTIONARY columns

Column	Type	Description
status	integer	Specifies whether the logical decoding data dictionary is valid.
restart_lsn	bigint	Reserved column.
restart_lsn_time	timestamp with time zone	Baseline completion time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
dictionary_csn_min	bigint	Minimum version number in use.
last_clean_time	timestamp with time zone	Time when the data dictionary table is cleared for the last time.
last_clean_result	boolean	Result of the last cleanup of the data dictionary table.
originid	integer	Source identifier of a tuple.
last_flush_csn	bigint	CSN of the last metadata information modification.
pdbid	oid	ID of the tenant corresponding to the record.

7.2.15.8 GS_LOGICAL_INDEX

GS_LOGICAL_INDEX stores some index information, and GS_LOGICAL_CLASS stores other information.

Table 7-60 GS_LOGICAL_INDEX columns

Column	Type	Description
indexrelid	Oid	OID of the index.
indrelid	Oid	Referenced RELATION OID.
indnatts	smallint	Total number of columns in an index.
indisunique	boolean	Specifies whether the index is a unique index.
indisprimary	boolean	Specifies whether the index is a primary key.
csnmin	bigint	CSN when a tuple is inserted.
csnmax	bigint	CSN when a tuple is updated or deleted.
createtime	timestamp with time zone	Timestamp when a tuple is inserted.
originid	integer	Source identifier of a tuple.
dbnode	oid	Database OID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
indimmediate	boolean	Specifies whether to execute the UNIQUE constraint immediately.
indisvalid	boolean	Specifies whether the index can be used for query.
indisready	boolean	Specifies whether this index is ready to accept inserts.
indisusable	boolean	Specifies whether this index can be used for insertion and selection.
indkey	int2vector	Index column number.
indcollation	oidvector	Sorting identifier.
indpred	pg_node_tree	Expression tree of the predicate if it is a partial index; otherwise, the value is **NULL**.
indisreplident	boolean	Specifies whether an index is identified as a replication identifier index.

7.2.15.9 GS_LOGICAL_NAMESPACE

GS_LOGICAL_NAMESPACE records namespace information.

Table 7-61 GS_LOGICAL_NAMESPACE columns

Column	Type	Description
nspid	oid	Namespace OID.
nspname	name	Namespace name.
nspowner	oid	OID of the namespace owner.
nsptimeline	bigint	Timeline identifier of the namespace.
csnmin	bigint	CSN when a tuple is inserted.
csnmax	bigint	CSN when a tuple is updated or deleted.
createtime	timestamp with time zone	Timestamp when a tuple is inserted.
originid	integer	Source identifier of a tuple.
dbnode	oid	Database OID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.15.10 GS_LOGICAL_PARTITION

GS_LOGICAL_PARTITION records partitioned table information.

Table 7-62 GS_LOGICAL_PARTITION columns

Column	Type	Description
partitionid	Oid	Partitioned table OID.
relname	name	Name of a partitioned table.
parttype	char	Partitioning type.
parentid	oid	OID of a parent table.
relfilenode	oid	Partitioned table relfilenode.
reltoastrelid	oid	OID of a TOAST table.
reltoastidxid	oid	OID of the TOAST index.
csnmin	bigint	CSN when a tuple is inserted.
csnmax	bigint	CSN when a tuple is updated or deleted.
originid	integer	Source identifier of a tuple.
createtime	timestamp with time zone	Timestamp when a tuple is inserted.
reltablespace	oid	Tablespace OID.
dbnode	oid	Database OID.

7.2.15.11 GS_LOGICAL_TYPE

GS_LOGICAL_TYPE stores information about data types.

Table 7-63 GS_LOGICAL_TYPE columns

Column	Type	Description
typname	name	Type name.
typnamespace	oid	OID of the namespace that contains this type.
typlen	smallint	Number of bytes used by a value of this type.
typbyval	boolean	Specifies whether the value of this type is passed by value or by reference.
typtype	"char"	Row type of the catalog.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
typisdefined	boolean	Specifies whether a type is completely defined.
typreceive	regproc	Receiving function in binary format.
typsend	regproc	Sending function.
typmodin	regproc	Input function for optional type modifiers.
typmodout	regproc	Output function.
typid	oid	Type OID.
dbnode	oid	Database OID.
originid	integer	Source identifier of a tuple.
createtime	timestamp with time zone	Timestamp when a tuple is inserted.
csnmin	bigint	CSN when a tuple is inserted.
csnmax	bigint	CSN when a tuple is updated or deleted.
typstorage	"char"	Data storage mode of a specified column. The options are as follows: ● **p**: plain storage. Numbers are always stored in a simple way. ● **e**: external storage. Data is stored outside the main table (for example, in the TOAST table). The main table stores references to data form external storage. ● **m**: main storage. Data can be stored compressed inline, which is similar to plain storage. This method is usually used to store types that require customized storage, for example, customized composite types. ● **x**: extended storage. Values may be compressed and/or moved into a level-2 relational table. **x** is usually the storage mode of the TOAST type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
typelem	oid	If **typelem** is not zero, it represents another row in the **gs_logical_type** table that defines the type (such as array) obtained by index. If **typsubscript** is zero, **typelem** should be zero. If the handler does not need **typelem** to determine the index type, **typelem** can also be zero when **typsubscript** is not zero. Note that **typelem** dependencies represent physical limitations on element types in this type; therefore, using DDL to modify a type may be restricted by the existence of this type.
typinput	regproc	Input conversion function (text format).
typoutput	regproc	Output conversion function (text format).
typrelid	oid	If this is a composite type (see **typtype**), then this column points to the GS_LOGICAL_CLASS entry that defines the corresponding table. For a composite type, the type of GS_LOGICAL_CLASS is not a table. For non-composite types, the value in this column is **0**.

7.2.15.12 GS_SQLAPPLY_SKIP

GS_SQLAPPLY_SKIP records rules that can be skipped during logical replay. This catalog is reserved in the current version.

Table 7-64 GS_SQLAPPLY_SKIP columns

Column	Type	Description
createtime	timestamp with time zone	Creation timestamp.
originid	integer	Replication source ID.
skiptype	integer	SKIP types: ● **1**: SKIP OBJ ● **2**: SKIP TXN ● **3**: SKIP ERR
dumpcontext	"char"	Dumps transactions to a file. ● **'a'**: The entire transaction is dumped. ● **'n'**: no dumping. ● **'s'**: Only changes skipped by the transaction are dumped.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
param1	text	For details, see the description in **Table 7-65** and **Table 7-66**.
param2	text	For details, see the description in **Table 7-65** and **Table 7-66**.
param3	text	For details, see **Table 7-65**.
param4	text	For details, see **Table 7-65**.
param5	text	Reserved column.

Table 7-65 Columns of param1–4 in SKIP OBJ or SKIP ERR

Column	Type	Description
db_name	text	Database name.
schema_name	text	Schema name.
object_name	text	Object name.
stmt_type	text	SQL statement type.

Table 7-66 Columns of param1–2 in SKIP TXN

Column	Type	Description
csn	text	Transaction SN.
xid	text	Transaction ID.

7.2.15.13 GS_TXN_LSN_TIME

GS_TXN_LSN_TIME records the mapping between logical decoding commit sequence numbers, log sequence numbers, and timestamps. In a distributed environment, it contains data only on DNs.

Table 7-67 GS_TXN_LSN_TIME columns

Column	Type	Description
create_csn	bigint	Commit sequence number.
create_lsn	bigint	Log sequence number.
snp_xmin	bigint	Minimum snapshot ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
create_time	timestamp with time zone	Timestamp when a tuple is inserted.
snp_snapshot	text	Serialized snapshot text.
barrier	text	Barrier information.

NOTE

- **create_lsn** is stored as a number in the system catalog. An LSN can be converted to the common LSN format (for reference only), that is, *XXXXXXXX*/*XXXXXXXX*, using the SQL syntax. The conversion syntax is as follows:

```
SELECT CONCAT(to_hex(({lsn_num}::bigint >> 32) & x'ffffffff'::bigint), '/', to_hex(({lsn_num}::bigint << 32 >> 32) & x'ffffffff'::bigint));
```

  When using the preceding SQL statement, convert **lsn_num** to the required number. The following uses **607906688** as an example:

```
gaussdb=# SELECT CONCAT(to_hex((607906688::bigint >> 32) & x'ffffffff'::bigint), '/',
to_hex((607906688::bigint << 32 >> 32) & x'ffffffff'::bigint));
concat
------------
0/243beb80
(1 row)
```

- An M-compatible database needs to use the following conversion statement:

```
SELECT CONCAT(hex(({lsn_num} >> 32) & 4294967295), '/', hex((({lsn_num}<< 32) >> 32) & 4294967295));
```

  When using the preceding SQL statement, convert **lsn_num** to the required number. The following uses **607906688** as an example:

```
gaussdb_m=# SELECT CONCAT(hex((607906688 >> 32) & 4294967295), '/', hex(((607906688 << 32) >> 32) & 4294967295));
  concat
------------
 0/243BEB80
(1 row)
```

7.2.15.14 GS_LOGICAL_SET

GS_LOGICAL_SET records metadata defined by the SET data type.

Table 7-68 GS_LOGICAL_SET columns

Column	Type	Description
setid	oid	Row identifier.
settypid	oid	OID of the SET data type.
setnum	tinyint	Number of members of the SET data type. A maximum of 64 members are supported.
setsortorder	tinyint	Sorting order of members when the SET data type is defined. The value starts from 0.
setlabel	text	Member name of the SET data type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Type	Description
dbnode	oid	Database OID.
originid	integer	Source identifier of a tuple.
createtime	timestamp with time zone	Timestamp when a tuple is inserted.
csnmin	bigint	CSN when a tuple is inserted.
csnmax	bigint	CSN when a tuple is updated or deleted.
numbers	integer	Number of members of the M-compatible SET and ENUM data types. This column is used only in M-compatible databases.
sortorder	integer	Sorting order of members when the M-compatible SET and ENUM data types are defined. This column is used only in M-compatible databases.

7.2.15.15 GS_LOGICAL_ENUM

GS_LOGICAL_ENUM contains entries showing the values and labels for each enumerated type.

Table 7-69 GS_LOGICAL_ENUM columns

Column	Type	Description
enumid	oid	Row identifier.
enumtypid	oid	OID of the **PG_TYPE** entry owning this enumerated value.
enumlabel	name	Textual label for this enumerated value.
enumsortorder	real	Sort position of this enumerated value within its enumerated type.
dbnode	oid	Database OID.
originid	integer	Source identifier of a tuple.
createtime	timestamp with time zone	Timestamp when a tuple is inserted.
csnmin	bigint	CSN when a tuple is inserted.
csnmax	bigint	CSN when a tuple is updated or deleted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.15.16 GS_SQLAPPLY_HISTORY

GS_SQLAPPLY_HISTORY records the decoding position of the start or end time of logical replay.

Table 7-70 GS_SQLAPPLY_HISTORY columns

Column	Parameter Type	Description
occurtime	timestamp with time zone	Start or end time of logical replay.
originid	integer	Decoding metadata ID of logical replay.
opkind	char	Start (**'b'**) or end (**'e'**) information of logical replay.
l_xid	bigint	Start or end transaction ID of logical replay in the local cluster.
l_lsn	bigint	Start or end log position of logical replay in the local cluster.
l_csn	bigint	Start or end transaction commit sequence of logical replay in the local cluster.
r_restartlsn	bigint	Start or end log stream position of logical decoding in the remote cluster.
r_confirmlsn	bigint	Earliest or latest commit log position of the transaction returned by logical decoding in the remote cluster.
r_confirmcsn	bigint	Earliest or latest commit sequence of the transaction returned by logical decoding in the remote cluster.
uuid	text	ID of the logical standby cluster.

7.2.16 DSL Rule Engine

7.2.16.1 GS_DSL_REWRITERULE

GS_DSL_REWRITERULE records DSL rule information. Only the initial user can modify this table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-71 GS_DSL_REWRITERULE columns

Name	Type	Description
rulename	name	Rule name, which is globally unique and is set when the rule is added. The name must start with a letter and can have up to 64 bytes, containing a mix of letters, digits, underscores (_), and hyphens (-). Chinese characters and other special characters are not allowed.
ruledesc	name	Brief description of the rule function. The value can contain a maximum of 64 bytes.
enabled	boolean	Specifies whether to enable this rule.
externalrule	boolean	The external rule is set to **TRUE**, and the built-in rule is set to **FALSE**.
costbased	boolean	Specifies whether to enable the rule based on the cost. This function is not supported currently.
priority	integer	Specifies the priority. The value is globally unique and ranges from 1 to 9999. A larger value indicates a lower priority.
dbcompatibility	tinyint	This column uses bit mapping to store the database compatibility mode of the rule. The bit is converted to a decimal number as follows: ● **01**: The database is ORA-compatible. ● **02**: The database is MySQL-compatible. ● **04**: The database is TD-compatible. ● **08**: The database is PG-compatible. ● **16**: The database is M-compatible. ● **31**: indicates that the database is compatible with all databases.
planmode	tinyint	Plan type supported by the rule. The options are **STREAM**, **REMOTEQUERY**, and **ANY**, with **ANY** being the default value. This column uses bit mapping to store the plan types supported by the rule. The bit IDs are as follows: ● **01**: The STREAM plan is supported. ● **02**: The REMOTEQUERY plan is supported. ● **03**: All plans are supported.
createtime	timestamp with time zone	Time when a rule is created for the first time, which is automatically generated by the system.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
lastupdatetime	timestamp with time zone	Time when the rule content is updated for the last time, which is automatically generated by the system.
rulehash	text	Hash value of the rule text, which is used to check the uniqueness of the text content.
ruletext	text	Rule text, which is used to store the definition of an external DSL rule. For a built-in DSL rule, this column is left blank.
conflictrule	text	Conflict rule. If a conflict rule that has been applied is detected, the rule is skipped. For example, if the value is **(rulea,ruleb,rulec)**, the rule is skipped when **rulea**, **ruleb**, or **rulec** has been rewritten.
rulegroup	name	Rule group, which is globally unique and is specified by the user when the rule is added. The name must start with a letter and can have up to 64 bytes, containing a mix of letters, digits, underscores (_), and hyphens (-). Chinese characters and other special characters are not allowed. This column is used only for built-in rules and is unavailable for external rules.
updateversion	integer	Version number of the updated rule. The system determines whether the rule takes effect based on the version number.

7.2.17 HTAP

7.2.17.1 GS_IMCV

GS_IMCV provides metadata about all IMCV catalogs.

Table 7-72 GS_IMCV columns

Name	Type	Description
reloid	oid	Table ID.
relname	name	Table name.
dbname	name	Database name.
username	name	Owner of the IMCV table.
parentoid	oid	Parent table ID. Specifies a parent table ID for a partition. **0** is displayed for non-partitions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
imcvispart	boolean	Specifies whether the table is a partitioned table.
imcvnattr	smallint	Number of columns to be loaded.
imcvkey	int2vector	Number of loaded columns.
priority	smallint	Loading priority. ● **'1'**: PRIORITY_NONE (no priority). ● **'2'**: PRIORITY_LOW (low priority). ● **'3'**: PRIORITY_HIGH (high priority).
childoids	oidvector_extend	If the table is a partitioned table, IDs of its subpartitions are recorded. Otherwise, it is empty.
dboid	oid	Database ID.
rootoid	oid	For an ordinary table, the value is the ID of the table. For a partitioned table, the value is the ID of the base table of the partitioned table.
schemaname	name	Schema name.

7.2.18 Other System Catalogs

7.2.18.1 GS_ASP

GS_ASP displays the persistent ACTIVE SESSION PROFILE samples. This system catalog can be queried only in the system library.

Table 7-73 GS_ASP columns

Name	Type	Description
sampleid	bigint	Sample ID.
sample_time	timestamp with time zone	Sampling time.
need_flush_sample	Boolean	Specifies whether the sample needs to be flushed to disks. ● **t** (true): yes ● **f** (false): no
databaseid	oid	Database ID.
thread_id	bigint	Thread ID.
sessionid	bigint	Session ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
start_time	timestamp with time zone	Start time of a session.
event	text	Event name. For details about key events in the kernel, see **Table 7-470**, **Table 7-471**, **Table 7-472**, and **Table 7-473**. For details about the impact of each transaction lock on services, see **LOCK**.
lwtid	integer	Lightweight thread ID of the current thread.
psessionid	bigint	Parent thread of the streaming thread.
tlevel	integer	Level of the streaming thread. The value corresponds to the level (ID) of the execution plan.
smpid	integer	Parallel thread ID in SMP execution mode.
userid	oid	ID of a session user.
application_name	text	Name of an application.
client_addr	inet	IP address of a client.
client_hostname	text	Name of a client.
client_port	integer	TCP port number used by a client to communicate with the backend.
query_id	bigint	ID of the unique SQL statement.
unique_query_id	bigint	ID of the normalized SQL statement.
user_id	oid	User ID in the key of the unique query.
cn_id	integer	CN from which the unique SQL statement comes. **cn_id** is in the key of the unique query.
unique_query	text	Standardized unique SQL text string.
locktag	text	Information of a lock that the session waits for, which can be parsed using **locktag_decode**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
lockmode	text	Mode of a lock that the session waits for. • **LW_EXCLUSIVE**: exclusive lock. • **LW_SHARED**: shared lock. • **LW_WAIT_UNTIL_FREE**: waits for **LW_EXCLUSIVE** to be available.
block_sessionid	bigint	Blocks a session from obtaining the session ID of a lock if the session is waiting for the lock.
wait_status	text	Provides more details about an event column.
global_sessionid	text	Global session ID. The structure is *nodeid*:*sessionid*#*seqid*. • *nodeid*: ID of the delivery node. • *sessionid*: session ID of the delivery node. • *seqid*: sequence number in ascending order. The ID is **0** on the delivery node. The *nodeid* and *sessionid* values of this column are the same for sessions delivered to other nodes in the same session.
xact_start_time	timestamp with time zone	Start time of the transaction.
query_start_time	timestamp with time zone	Time when the statement starts to be executed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
state	text	Current statement state. The options are as follows: • **active**: The background is executing a query. • **idle in transaction**: The background is in a transaction, but there is no statement being executed in the transaction. • **idle in transaction (aborted)**: The background is in a transaction, but there are statements failed in the transaction. • **fastpath function call**: The background is executing a fast-path function. • **disabled**: This state is reported if **track_activities** is disabled in this background.
event_start_time	timestamp with time zone	Start time of a wait event.
current_xid	xid	ID of the current transaction.
top_xid	xid	ID of the top-level transaction that is being executed.
instance_name	text	CN or DN to which the record belongs.
parent_unique_sql_id	bigint	Calls the **sql_id** of the parent statement of the current statement.
control_status	text	The value is **under_cpu_throttle** when CPU resources for abnormal SQL statements are controlled. Otherwise, the value is **normal**.
top_unique_query_id	bigint	This column displays the **unique_query_id** of the top-level SQL statement of the stored procedure.
top_unique_query	text	This column displays the **unique_query** of the top-level SQL statement of the stored procedure.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
sql_hash	bigint	Hash value of the current SQL statement, which is **0**.
plan_hash	bigint	Hash value of the execution plan of the current SQL statement, which is **0**.

7.2.18.2 GS_COPY_ERROR_OLD

GS_COPY_ERROR_OLD captures error rows related to the COPY FROM procedure for future reference.

This system catalog is used only for compatibility with earlier versions. That is, it may contain data in the database upgraded from an earlier version. In this version, this system catalog is invalid, and the COPY statement does not record any data in it.

Table 7-74 GS_COPY_ERROR_OLD columns

Name	Type	Description
relname	character varying	Table name, in the form of *Schema name*.*Table name*.
begintime	timestamp with time zone	Time when a data format error was reported.
filename	character varying	Name of the source data file where a data format error occurs.
lineno	bigint	Number of the row where a data format error occurs in a source data file.
rawrecord	text	Raw record of a data format error in the source data file.
detail	text	Error details.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- Currently, the pg_catalog.gs_copy_error_old table does not have an automatic clearing mechanism. The administrator needs to manually run the TRUNCATE statement to clear the table.
- The pg_catalog.gs_copy_error_old table does not have row-level security.

7.2.18.3 GS_COPY_SUMMARY_LOG

In a distributed deployment environment, this system catalog does not take effect, that is, no data is recorded.

Table 7-75 GS_COPY_SUMMARY_LOG columns

Name	Type	Description
relname	text	Name of the target table to be imported.
begintime	timestamp with time zone	Start time of an import task.
endtime	timestamp with time zone	End time of an import task.
id	bigint	ID of the import transaction.
pid	bigint	ID of the worker thread for the current import.
readrows	bigint	Total number of data rows read by the import task.
skiprows	bigint	Total number of data rows skipped in the import task.
loadrows	bigint	Number of data rows successfully imported in the current import task.
errorrows	bigint	Number of error data rows in the current import task.
whenrows	bigint	Number of data rows that violate the WHEN filter criterion in the current import task.
allnullrows	bigint	Number of data rows where all columns are empty.
detail	text	Summary of the import task, including the number of successfully imported rows, number of error data rows, number of rows that violate the WHEN condition, and number of blank rows.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
custom_id	text	Value of **copy_custom_id** in the COPY syntax, which indicates the information about the imported custom identifier.

NOTE

- Currently, the pg_catalog.gs_copy_summary_log table does not have an automatic clearing mechanism. The administrator needs to manually run the TRUNCATE statement to clear the table.
- The pg_catalog.gs_copy_summary_log table does not have row-level security.

7.2.18.4 GS_COPY_ERROR_LOG

GS_COPY_ERROR_LOG captures error rows related to the COPY FROM procedure for future reference.

- If the GUC parameter **enableSeparationOfDuty** is set to **on**, the user can be a common database user or an administrator.
- If the GUC parameter **enableSeparationOfDuty** is set to **off**, the user can be a common database user or an administrator. Common users require permissions granted by an administrator (by setting the **allow_create_sysobject** parameter to **on**.) An administrator can use it directly.

Table 7-76 GS_COPY_ERROR_LOG columns

Name	Type	Description
relname	text	Table name, in the form of *Schema name*.*Table name*.
begintime	timestamp with time zone	Time when a data format error was reported.
filename	text	Name of the source data file where a data format error occurs.
lineno	bigint	Number of the row where a data format error occurs in a source data file.
rawrecord	text	Raw record of a data format error in the source data file.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
detail	text	Error details.
custom_id	text	Value of **copy_custom_id** in the COPY syntax, which indicates the information about the imported custom identifier.

NOTE

- Currently, the pg_catalog.gs_copy_error_log table does not have an automatic clearing mechanism. The administrator needs to manually run the TRUNCATE statement to clear the table.
- The pg_catalog.gs_copy_error_log table does not have row-level security.
- After scale-in, the data in the pg_catalog.gs_copy_error_log table on the removed DN will be lost. Before scale-in, check whether the data in the table is required. If the data is required, back up the data in advance.
- In separation-of-duties scenarios, error logs are not recorded in pg_catalog.pgxc_copy_error_log or pg_catalog.gs_copy_error_log. If error information needs to be recorded for the **load_user** user, execute the following statement to create the error log in the schema with the same name as the user:

  ```
  CREATE TABLE load_user.pgxc_copy_error_log (relname varchar, begintime timestamptz, filename varchar, lineno int8, rawrecord text, detail text);
  CREATE INDEX copy_error_log_relname_idx ON load_user.pgxc_copy_error_log(relname);
  ```

  (Optional) Add the **custom_id** column to the table.

  ```
  ALTER TABLE load_user.pgxc_copy_error_log ADD COLUMN custom_id varchar(64);
  ```

7.2.18.5 GS_DEP_SOURCE

GS_DEP_SOURCE records metadata and partial compilation results of objects. Only system administrators can view the information.

Table 7-77 GS_DEP_SOURCE columns

Name	Type	Reference	Description
object_oid	oid	**object_oid** in PG_OBJECT	Object identifier.
object_type	"char"	-	Object type. **v** indicates a view.
source	text	-	Defined meta information.
compile	pg_node_tree	-	Partial compilation result.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

For the **source** column, the asterisk (*) in "SELECT *" in the view definition statement is expanded, and the schema used by the created view is appended. The following is an example:

```
gaussdb=# CREATE USER user01 WITH SYSADMIN PASSWORD 'xxxx';
gaussdb=# SET ROLE user01 PASSWORD 'xxxx';
-- Create table t1 under the user01 schema.
gaussdb=# CREATE TABLE t1(c1 INT, c2 INT);
-- Create view v1 under the user01 schema. The table queried by v1 is user01.t1.
gaussdb=# CREATE VIEW v1 AS SELECT * FROM t1;
-- Query the source column.
gaussdb=# SELECT source FROM gs_dep_source WHERE object_oid = xxx;
 source
--------
CREATE OR REPLACE VIEW user01.v1(c1, c2) AS SELECT t1.c1, t1.c2 FROM user01.t1;
(1 row)
```

7.2.18.6 GS_DEPENDENCIES

GS_DEPENDENCIES records object dependency information and has a one-to-many relationship with **GS_DEPENDENCIES_OBJ**.

Table 7-78 GS_DEPENDENCIES columns

Name	Type	Description
schemaname	name	Name of a namespace.
packagename	name	Name of a package.
refobjpos	integer	Position of the referenced object. • **1**: type • **2**: package header • **4**: function header • **8**: function body • **16**: package body • **32**: view
refobjoid	oid	OID of the referenced object.
objectname	text	Name of the dependent object.

7.2.18.7 GS_DEPENDENCIES_OBJ

GS_DEPENDENCIES_OBJ records the detailed information about referenced objects.

Table 7-79 GS_DEPENDENCIES_OBJ columns

Name	Type	Description
schemaname	name	Name of a namespace.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
packagename	name	Name of a package.
type	integer	Type of the referenced object. • **1**: unknown type • **2**: variable • **3**: type • **4**: function • **5**: view • **6**: function header
name	text	Name of the referenced object
objnode	pg_node_tree	Detailed information about the referenced object

7.2.18.8 GS_GLOBAL_CONFIG

GS_GLOBAL_CONFIG records the parameter values specified by users during cluster initialization. In addition, it also stores weak passwords set by users and status of the cluster (whether it is being scaled, which is specified by **redis_bucket_expansion**). Initial database users can write, modify, and delete parameters in system catalogs by using ALTER and DROP. Only the initial user and users with the SYSADMIN or CREATEROLE permission can access this system catalog. Other users do not have such permission.

Table 7-80 GS_GLOBAL_CONFIG columns

Name	Type	Description
name	name	Specifies the preset parameter name, weak password name, name of parameter for redistribution after scale-out, or parameter required by users during cluster initialization.
value	text	Specifies the preset parameter value, weak password name, parameter for redistribution after scale-out, or parameter value required by users during cluster initialization.

NOTICE

You are advised not to write, modify, or delete the **redis_bucket_expansion** parameter. The parameter name has been used to indicate that the cluster is being scaled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.18.9 GS_JOB_ATTRIBUTE

GS_JOB_ATTRIBUTE records attributes of DBE_SCHEDULER scheduled tasks, including basic attributes of scheduled tasks, scheduled task classes, certificates, authorization, programs, and schedules. Common users do not have the permission to access the newly installed database cluster.

Table 7-81 GS_JOB_ATTRIBUTE columns

Name	Type	Description
oid	oid	Row identifier (hidden column).
job_name	text	Names of scheduled tasks, scheduled task classes, certificates, programs, and schedules, and authorized user names.
attribute_name	text	Attribute names of scheduled tasks, scheduled task classes, certificates, programs, and schedules, and authorized content.
attribute_value	text	Attribute values of scheduled tasks, scheduled task classes, certificates, programs, and schedules.

7.2.18.10 GS_JOB_ARGUMENT

GS_JOB_ARGUMENT provides the parameter attributes of DBE_SCHEDULER scheduled tasks and programs.

Table 7-82 GS_JOB_ARGUMENT columns

Name	Type	Description
oid	oid	Row identifier (hidden column).
argument_position	integer	Location of a parameter of a scheduled task or program.
argument_type	name	Parameter type of a scheduled task or program.
job_name	text	Name of a scheduled task or program.
argument_name	text	Parameter name of a scheduled task or program. The scheduled task inherits the parameter name of the program. Therefore, this parameter is null.
argument_value	text	Parameter value of a scheduled task. (The program cannot bind a value.)
default_value	text	Default parameter value of a program.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.18.11 GS_PACKAGE

GS_PACKAGE records package information.

Table 7-83 GS_PACKAGE columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
pkgnamespace	oid	Schema to which a package belongs.
pkgowner	oid	Owner of a package.
pkgname	name	Name of a package.
pkgspecsrc	text	Package specification.
pkgbodydeclsrc	text	Package body.
pkgbodyinitsrc	text	Package initialization source.
pkgacl	aclitem[]	Access permission.
pkgsecdef	boolean	Specifies whether the package has the definer permission.

7.2.18.12 GS_PLAN_TRACE

GS_PLAN_TRACE is used to store plan traces. It records details about the plan generation process of DML statements. Only users with the SYSADMIN permission can use this system catalog. The plan trace feature is not supported in the distributed scenario. Therefore, no data is displayed in this view in the distributed scenario.

Table 7-84 GS_PLAN_TRACE columns

Name	Type	Description
query_id	text	Unique ID of the current request.
query	text	SQL statement of the current request. The value of this column cannot exceed the value of **track_activity_query_size**.
unique_sql_id	bigint	Unique ID of the SQL statement of the current request.
plan	text	Query plan text corresponding to the currently requested SQL statement. The value of this column cannot exceed 10 KB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
plan_trace	text	Details about the process of generating the query plan corresponding to the currently requested SQL statement. The value of this column cannot exceed 300 MB.
owner	oid	OID of the user who initiates the current SQL request.
modifydate	timestamp with time zone	Time when the current plan trace is updated (that is, time when the plan trace is created).

7.2.18.13 GS_POLICY_LABEL

GS_POLICY_LABEL records the resource label configuration information. One resource label corresponds to one or more records, and each record identifies the resource label to which a database resource belongs. Only the SYSADMIN or POLADMIN can access this system catalog.

Fully Qualified Domain Name (FQDN) identifies an absolute path of a database resource.

Table 7-85 GS_POLICY_LABEL columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
labelname	name	Specifies the resource label name.
labeltype	name	Resource label type. Currently, the value is **RESOURCE**.
fqdnnamespace	oid	OID of a namespace to which an identified database resource belongs.
fqdnid	oid	OID of an identified database resource. If the database resource is a column, this column is the OID of the catalog.
relcolumn	name	Column name. If the identified database resource is a column, this column indicates the column name. Otherwise, this column is empty.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
fqdntype	name	Type of the identified database resource, for example, schema, table, column, or view.

7.2.18.14 GS_RECYCLEBIN

GS_RECYCLEBIN describes details about objects in the recycle bin of the flashback feature.

Table 7-86 GS_RECYCLEBIN columns

Name	Type	Description
oid	oid	System column.
rcybaseid	oid	Base table object ID, which references **gs_recyclebin.oid**.
rcydbid	oid	OID of the database to which the current object belongs.
rcyrelid	oid	OID of the current object.
rcyname	name	Name of the object in the recycle bin. The format is BIN*$unique_id$oid$*0. *unique_id* indicates the unique identifier with a maximum of 16 characters, and *oid* indicates the object identifier.
rcyoriginname	name	Original object name.
rcyoperation	char	Operation type. • **d**: drop • **t**: truncate

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
rcytype	integer	Object type. ● **0**: table ● **1**: index. ● **2**: TOAST table. ● **3**: TOAST index. ● **4**: sequence, indicating the sequence object that is automatically associated with the serial, bigserial, smallserial, and largeserial types. ● **5**: partition. ● **6**: global index. ● **7**: materialized view. ● **8**: global secondary index.
rcyrecyclecsn	bigint	CSN when an object is dropped or truncated.
rcyrecycletime	timestamp with time zone	Time when an object is dropped or truncated.
rcycreatecsn	bigint	CSN when an object is created.
rcychangecsn	bigint	CSN when an object definition is modified.
rcynamespace	oid	OID of the namespace that contains this relationship.
rcyowner	oid	Owner of the relationship.
rcytablespace	oid	Tablespace in which this relationship is stored. If the value is **0**, the default tablespace of the database is used. This column is meaningless if the relationship has no on-disk file.
rcyrelfilenode	oid	File name of the recycle bin object on a disk, or **0** if none, which is used to restore the physical file when the TRUNCATE object is restored.
rcycanrestore	Boolean	Specifies whether flashback can be performed separately.
rcycanpurge	Boolean	Specifies whether the purge operation can be performed independently.
rcyfrozenxid	xid32	All transaction IDs before this one have been replaced with a permanent ("frozen") transaction ID in the table.
rcyfrozenxid64	xid	All transaction IDs before this one have been replaced with a permanent ("frozen") transaction ID in the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
rcybucket	oid	Bucket information in 12.2.15.42 PG_HASHBUCKET.

7.2.18.15 GS_SECURITY_LABEL

GS_SECURITY_LABEL is a shared system catalog that records information about security labels. Each record corresponds to a security label.

Table 7-87 GS_SECURITY_LABEL columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
label_name	name	Security label name.
label_content	text	Content of the security label.

7.2.18.16 GS_SQL_PATCH

GS_SQL_PATCH records the status information about SQL_PATCH on the current node.

Table 7-88 GS_SQL_PATCH columns

Name	Type	Description
patch_name	name	Patch name.
unique_sql_id	bigint	Global unique ID.
owner	oid	ID of the user who creates the patch.
enable	Boolean	Specifies whether the patch takes effect.
status	"char"	Specifies whether the SQL patch is created based on the unique SQL ID (value: **'d'**) or SQL hash value (value: **'h'**).
abort	Boolean	Specifies whether the patch is an abort hint.
hint_string	text	Hint text.
hint_node	pg_node_tree	Hint parsing and serialization result.
original_query	text	Original statement (reserved column).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
patched_query	text	Patched statement (reserved column).
original_query_tree	pg_node_tree	Original statement parsing result (reserved column).
patched_query_tree	pg_node_tree	Patched statement parsing result (reserved column).
description	text	Patch description.
parent_unique_sql_id	bigint	Globally unique ID of the SQL outer statement for which the patch takes effect. The value of this parameter is **0** for statements outside a stored procedure. For statements inside the stored procedure, the value of this parameter is the globally unique ID of the statement that calls the stored procedure.

7.2.18.17 GS_STATISTIC_EXT_HISTORY

GS_STATISTIC_EXT_HISTORY is a multi-column historical statistics management table. It stores historical extended statistics about tables in the database, including multi-column statistics and expression statistics (supported later).

Table 7-89 GS_STATISTIC_EXT_HISTORY columns

Name	Type	Description
starelid	oid	Table or index that the described column belongs to.
starelkind	"char"	Type of the object to which a table belongs. **'c'** indicates an ordinary table, and **'p'** indicates a partitioned table.
stainherit	Boolean	Specifies whether to collect statistics for objects that have inheritance relationship.
statimestamp	timestamp with time zone	Time when the statistics are collected.
stanullfrac	real	Percentage of column entries that are null.
stawidth	integer	Average stored width, in bytes, of non-null entries.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
stadistinct	real	Number of distinct, non-null data values in the column for database nodes. • A value greater than 0 is the actual number of distinct values. • A value less than 0 is the ratio of the **distinct** value to the total number of rows. For example, if **stadistinct** is **-0.5**, the actual **distinct** value is the total number of rows multiplied by 0.5. • The value **0** indicates that the number of distinct values is unknown.
stadndistinct	real	Number of unique non-null data values in the **DN1** column. • A value greater than 0 is the actual number of distinct values. • A value less than 0 is the ratio of the **distinct** value to the total number of rows. For example, if **stadndistinct** is **-0.5**, the actual **distinct** value is the total number of rows multiplied by 0.5. • The value **0** indicates that the number of distinct values is unknown.
standvfunc	"char"	Algorithm used to calculate the NDV based on the statistics. • **d**: The original DUJ1 algorithm is used for estimation. • **c**: The C19 algorithm is used for estimation.
staorigin	"char"	Source of the statistics collection mode. • **a**: The collection is triggered by AUTOANALYZE. • **m**: The collection is triggered by manual ANALYZE. • **g**: The gsstat thread is triggered to perform ANALYZE collection when a large amount of data is inserted.
stakindN	smallint	Code number stating that the type of statistics is stored in slot *N* of the *pg_statistic* row. The value of *N* ranges from 1 to 5.
staopN	oid	Operator used to generate the statistics stored in slot *N*. For example, a histogram slot shows the < operator that defines the sort order of the data. The value of *N* ranges from 1 to 5.
stakey	int2vector	Array of a column ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
stanumbersN	real[]	Numerical statistics of the appropriate type for slot *N*. The returned value is **NULL** if the slot does not involve numerical values. The value of *N* ranges from 1 to 5.
stavaluesN	anyarray	Column data values of the appropriate type for slot *N*. The returned value is **NULL** if the slot type does not store any data values. Each array's element values are actually of the specific column's data type so there is no way to define these columns' type more specifically than anyarray. The value of *N* ranges from 1 to 5.
staexprs	pg_node_tree	Expression corresponding to the extended statistics information.
stasource	"char"	Source of extended statistics: ● **'a'**: indicates that the statistics data is automatically created, which is controlled by the GUC parameter **auto_statistic_ext_columns**. ● **'m'**: indicates that a user manually creates the statistics data using analyze tablename ((column list)) or alter table tablename add statistics ((column list)).
stastatus	"char"	Status of extended statistics: ● **'a'**: active and available. ● **'d'**: disabled. Related information is not collected, and the optimizer does not use the data when generating a plan. You can use the alter table tablename disable/enable statistics((column list)) syntax to modify the status of extended statistics.
staextname	name	Alias of the multi-column group of multi-column statistics.

7.2.18.18 GS_STATISTIC_HISTORY

GS_STATISTIC_HISTORY is a single-column historical statistics management table that stores historical statistics about tables and index columns in a database.

Table 7-90 GS_STATISTIC_HISTORY columns

Name	Type	Description
starelid	oid	Table or index that the described column belongs to.
starelkind	"char"	Type of an object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
staattnum	smallint	Number of the described column in the table, starting from 1.
stainherit	Boolean	Specifies whether to collect statistics for objects that have inheritance relationship.
statimestamp	timestamp with time zone	Time when the statistics are collected.
stanullfrac	real	Percentage of column entries that are null.
stawidth	integer	Average stored width, in bytes, of non-null entries.
stadistinct	real	Number of distinct, non-null data values in the column for database nodes. ● A value greater than 0 is the actual number of distinct values. ● A value less than 0 is the ratio of the **distinct** value to the total number of rows. For example, if **stadistinct** is **-0.5**, the actual **distinct** value is the total number of rows multiplied by 0.5. ● The value **0** indicates that the number of distinct values is unknown.
standvfunc	"char"	Algorithm used to calculate the NDV based on the statistics. ● **d**: The original DUJ1 algorithm is used for estimation. ● **c**: The C19 algorithm is used for estimation. ● **u**: The algorithm that calculates the NDV cannot be obtained from the current CN. You can query the same timestamp in history tables of other CNs for the algorithm.
staorigin	"char"	Source of the statistics collection mode. ● **a**: The collection is triggered by AUTOANALYZE. ● **m**: The collection is triggered by manual ANALYZE. ● **g**: The gsstat thread is triggered to perform ANALYZE collection when a large amount of data is inserted.
stakindN	smallint	Code number stating that the type of statistics is stored in slot *N* of the *pg_statistic* row. The value of *N* ranges from 1 to 5.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
staopN	oid	Operator used to generate the statistics stored in slot *N*. For example, a histogram slot shows the < operator that defines the sort order of the data. The value of *N* ranges from 1 to 5.
stanumbersN	real[]	Numerical statistics of the appropriate type for slot *N*. The returned value is **NULL** if the slot does not involve numerical values. The value of *N* ranges from 1 to 5.
stavaluesN	anyarray	Column data values of the appropriate type for slot *N*. The returned value is **NULL** if the slot type does not store any data values. Each array's element values are actually of the specific column's data type so there is no way to define these columns' type more specifically than anyarray. The value of *N* ranges from 1 to 5.
stadndistinct	real	Number of unique non-null data values in the **DN1** column. ● A value greater than 0 is the actual number of distinct values. ● A value less than 0 is the ratio of the **distinct** value to the total number of rows. For example, if **stadndistinct** is **-0.5**, the actual **distinct** value is the total number of rows multiplied by 0.5. ● The value **0** indicates that the number of distinct values is unknown.
staextinfo	text	Information about extension statistics. Reserved column.

7.2.18.19 GS_SYSTEM_OBJECTS_VERSION

GS_SYSTEM_OBJECTS_VERSION records the version information of system objects. It is used to determine the changed and new system objects during the upgrade, and update the system object definitions or create system objects.

The system object types written into this system catalog include views, functions, tables, indexes, types, rules, and type casting relationships (CAST).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-91 GS_SYSTEM_OBJECTS_VERSION columns

Name	Type	Description
obj_id	oid	Identifier of a system object. For different types of system objects, the relationships are as follows: Unique identifier of a view, table, or index in **PG_CLASS**, and unique identifier of a type, function, rule, or CAST in **PG_TYPE**, **PG_PROC**, **PG_REWRITE**, or **PG_CAST**, respectively.
obj_type	char	Type of a system object. • **v**: view. • **f**: function. • **r**: ordinary table. • **i**: index. • **c**: composite type. • **d**: domain type. • **R**: rule. • **C**: type casting.
obj_version	int8	Version number of a system object. **0**: initial version number. **-1**: version number of a deleted object. Other version numbers are used for subsequent object changes.

7.2.18.20 GS_TABLESTATS_HISTORY

GS_TABLESTATS_HISTORY manages historical statistics at the table, index, and partition levels. It stores historical statistics about tables, indexes, and partitions in a database.

Table 7-92 GS_TABLESTATS_HISTORY columns

Name	Type	Description
relid	oid	Unique ID of a table, index, or partition in a 12.2.15.28 PG_CLASS or 12.2.1.1 PG_PARTITION.
relname	name	Name of a table, index, or partition.
relnamespace	oid	OID of the namespace that contains this object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
relkind	"char"	Object type. • **r**: ordinary table. • **I**: table-level index. • **i**: partitioned index. • **p**: partition. • **s**: subpartition.
reltimestamp	timestamp with time zone	Time when the statistics are collected.
relpages	double precision	Size of the table on disk in pages. This is only an estimate used by the optimizer.
reltuples	double precision	Number of rows in the table. This is only an estimate used by the optimizer.
relallvisible	integer	Number of pages marked as all visible in the table.

7.2.18.21 GS_TS_DICT_DEFINITION

GS_TS_DICT_DEFINITION stores the contents of user-defined dictionaries, used for creating and using user-defined dictionaries. In the current version, system catalogs can be viewed only.

Table 7-93 GS_TS_DICT_DEFINITION columns

Name	Type	Description
dict_oid	oid	OID of a dictionary to which the dictionary content belongs.
definition_type	"char"	Dictionary content type.
definition	text	Dictionary content definition.

7.2.18.22 GS_TXN_SNAPSHOT

GS_TXN_SNAPSHOT is a timestamp-CSN mapping table. It periodically samples and maintains an appropriate time range to estimate the CSN corresponding to the timestamp in the range.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-94 GS_TXN_SNAPSHOT columns

Name	Type	Description
snptime	timestamp with time zone	Snapshot time
snpxmin	bigint	Minimum snapshot ID
snpcsn	bigint	Snapshot CSN
snpsnapshot	text	Serialized snapshot text

7.2.18.23 GS_UID

GS_UID records the unique identification meta information of the **hasuids** attribute table in the database.

Table 7-95 GS_UID columns

Name	Type	Description
relid	oid	OID of a table
uid_backup	bigint	Largest unique identifier that can be assigned to a table

7.2.18.24 GS_WORKLOAD_RULE

GS_WORKLOAD_RULE records information about SQL concurrency control rules. There is no permission restriction on this system catalog. All users can query this system catalog.

Table 7-96 GS_WORKLOAD_RULE columns

Name	Type	Description
rule_id	bigint	Concurrency control rule ID, which is automatically generated by the system.
rule_name	name	Name of the concurrency control rule, which is used for search. The name may not be unique and can be **NULL**.
databases	name[]	List of databases on which the concurrency control rules take effect. If the value is **NULL**, the concurrency control rules take effect for all databases.
max_workload	bigint	Maximum number of concurrent rule settings.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
is_valid	boolean	Determines whether the concurrency control rules take effect. If the concurrency control rules time out, the value is set to **false**.
start_time	timestamp with time zone	Start time of the concurrency control rules. The value **NULL** indicates that the rules take effect from now on.
end_time	timestamp with time zone	End time of the concurrency control rules. The value **NULL** indicates that the rules are always effective.
rule_type	text	Concurrency control rule type. Currently, only **"sqlid"**, **"select"**, **"insert"**, **"update"**, **"delete"**, **"merge"**, **"resource"**, and **"client"** are supported. Other values are invalid.
option_val	text[]	Parameter values of a concurrency control rule, including the SQL IDs, keyword lists, resource limitations, client IP addresses, and application names. For details, see the **gs_add_workload_rule** API description.
node_names	text[]	List of nodes on which the concurrency control rules take effect. This parameter is reserved and does not take effect currently.
user_names	text[]	List of users for which the concurrency control rules take effect. This parameter is reserved and does not take effect currently.

NOTICE

For the client concurrency control rule, in the **option_val** column, **ApplicationName** retains and displays the original complete content entered by the user. During client concurrency control rule matching, the maximum valid length of **ApplicationName** is 64 bytes (including 63 available bytes and the terminator). If the original input length exceeds 63 bytes, the value is automatically truncated and only the first 63 bytes are used for rule matching.

7.2.18.25 PG_AGGREGATE

PG_AGGREGATE records information about aggregate functions. Each entry in PG_AGGREGATE is an extension of an entry in PG_PROC. The PG_PROC entry

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

carries the aggregate's name, input and output data types, and other information that is similar to ordinary functions.

Table 7-97 PG_AGGREGATE columns

Name	Type	Reference	Description
aggfnoid	regproc	**proname** in **PG_PROC**	**proname** of the aggregate function in **PG_PROC**.
aggtransfn	regproc	**proname** in **PG_PROC**	Transition function
aggcollectfn	regproc	**proname** in **PG_PROC**	Collect function
aggfinalfn	regproc	**proname** in **PG_PROC**	Final function (**0** if none)
aggsortop	oid	**oid** in **PG_OPERATOR**	Associated sort operator (**0** if none)
aggtranstype	oid	**oid** in **PG_TYPE**	Data type of the aggregate function's internal transition (state) data The possible values and their meanings are defined by the types in **pg_type.h**. The main two types are polymorphic (isPolymorphicType) and non-polymorphic.
agginitval	text	-	Initial value of the transition state. This is a text column containing the initial value in its external string representation. If this column is null, the transition state value starts from null.
agginitcollect	text	-	Initial value of the collection state. This is a text column containing the initial value in its external string representation. If this column is null, the collection state value starts from null.
aggkind	"char"	-	Type of the aggregate function: • **n**: normal aggregate • **o**: ordered set aggregate

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
aggnumdirectargs	smallint	-	Number of direct parameters (non-aggregation-related parameters) of the aggregate function of the ordered set aggregate type. For an aggregate function of the normal aggregate type, the value is **0**.
agginitfn	regproc	**proname** in PG_PROC	Initialization function.

7.2.18.26 PG_AM

PG_AM records information about index access methods. There is one row for each index access method supported by the system.

Table 7-98 PG_AM columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
amname	name	-	Name of the access method.
amstrategies	smallint	-	Number of operator strategies for the access method (**0** if the access method does not have a fixed set of operator strategies).
amsupport	smallint	-	Number of support routines for the access method.
amcanorder	boolean	-	Specifies whether the access method supports ordered scans sorted by the indexed column's value. • **t** (true): supported. • **f** (false): not supported.
amcanorderbyop	boolean	-	Specifies whether the access method supports ordered scans sorted by the result of an operator on the indexed column. • **t** (true): supported. • **f** (false): not supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
amcanbackward	boolean	-	Specifies whether the access method supports backward scanning. • **t** (true): supported. • **f** (false): not supported.
amcanunique	boolean	-	Specifies whether the access method supports unique indexes. • **t** (true): supported. • **f** (false): not supported.
amcanmulticol	boolean	-	Specifies whether the access method supports composite indexes. • **t** (true): supported. • **f** (false): not supported.
amoptionalkey	boolean	-	Specifies whether the access method supports scanning without any constraint for the first index column. • **t** (true): supported. • **f** (false): not supported.
amsearcharray	boolean	-	Specifies whether the access method supports **ScalarArrayOpExpr** searches. • **t** (true): supported. • **f** (false): not supported.
amsearchnulls	boolean	-	Specifies whether the access method supports **IS NULL/NOT NULL** searches. • **t** (true): supported. • **f** (false): not supported.
amstorage	boolean	-	Specifies whether the index storage data type can differ from the column data type. • **t** (true): allowed. • **f** (false): not allowed.
amclusterable	boolean	-	Specifies whether an index of this type can be clustered on. • **t** (true): allowed. • **f** (false): not allowed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
ampredlocks	boolean	-	Specifies whether an index of this type manages fine-grained predicate locks. • **t** (true): allowed. • **f (false)**: not allowed.
amkeytype	oid	**oid** in PG_TYPE	Type of data stored in index (**0** if it is not a fixed type).
aminsert	regproc	**proname** in PG_PROC	"Insert this tuple" function.
ambeginscan	regproc	**proname** in PG_PROC	"Prepare for index scan" function.
amgettuple	regproc	**proname** in PG_PROC	"Next valid tuple" function (**0** if none).
amgetbitmap	regproc	**proname** in PG_PROC	"Fetch all valid tuples" function (**0** if none).
amrescan	regproc	**proname** in PG_PROC	"(Re)start index scan" function.
amendscan	regproc	**proname** in PG_PROC	"Clean up after index scan" function.
ammarkpos	regproc	**proname** in PG_PROC	"Mark current scan position" function.
amrestrpos	regproc	**proname** in PG_PROC	"Restore marked scan position" function.
ammerge	regproc	**proname** in PG_PROC	"Merge multiple indexes" function.
ambuild	regproc	**proname** in PG_PROC	"Build new index" function.
ambuildempty	regproc	**proname** in PG_PROC	"Build empty index" function.
ambulkdelete	regproc	**proname** in PG_PROC	Bulk-delete function.
amvacuumcleanup	regproc	**proname** in PG_PROC	Post-VACUUM cleanup function.
amcanreturn	regproc	**proname** in PG_PROC	Function to check whether the index supports index-only scans (**0** if none).
amcostestimate	regproc	**proname** in PG_PROC	Function to estimate cost of an index scan.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
amoptions	regproc	**proname** in **PG_PROC**	Function to parse and validate **reloptions** for an index.

7.2.18.27 PG_AMOP

PG_AMOP records information about operators associated with access method operator families. There is one row for each operator that is a member of an operator family. A family member can be either a search operator or an ordering operator. An operator can appear in more than one family, but cannot appear in more than one search position nor more than one ordering position within a family.

Table 7-99 PG_AMOP columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
amopfamily	oid	**PG_OPFAMILY**.oid	Operator family of this entry.
amoplefttype	oid	**oid** in **PG_TYPE**	Left-hand input data type of the operator. For details about the values and their descriptions, see **Data Types**.
amoprighttype	oid	**oid** in **PG_TYPE**	Right-hand input data type of the operator. For details about the values and their descriptions, see **Data Types**.
amopstrategy	smallint	-	Number of operator strategies.
amoppurpose	"char"	-	Purpose of the operator. • **s**: search • **o**: order
amopopr	oid	**PG_OPERATOR**.oid	OID of the operator.
amopmethod	oid	**PG_AM**.oid	Operator family of the index access method.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
amopsortfamily	oid	PG_OPFAMILY.oid	The B-tree operator family according to which this entry sorts for an ordering operator (**0** for a search operator)

A search operator entry indicates that an index of this operator family can be searched to find all rows satisfying **WHERE indexed_column operator constant**. Such an operator must return a Boolean value, and its left-hand input type must match the index's column data type.

An ordering operator entry indicates that an index of this operator family can be scanned to return rows in the order represented by **ORDER BY indexed_column operator constant**. Such an operator could return any sortable data type, though again its left-hand input type must match the index's column data type. The exact semantics of the **ORDER BY** are specified by the **amopsortfamily** column, which must reference a btree operator family for the operator's result type.

7.2.18.28 PG_AMPROC

PG_AMPROC records information about the support procedures associated with the access method operator families. There is one row for each support procedure that belongs to an operator family.

Table 7-100 PG_AMPROC columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
amprocfamily	oid	PG_OPFAMILY.oid	Operator family of this entry.
amproclefttype	oid	**oid** in PG_TYPE	Left-hand input data type of the associated operator. For details about common data types, see **Data Types**.
amprocrighttype	oid	**oid** in PG_TYPE	Right-hand input data type of the associated operator. For details about common data types, see **Data Types**.
amprocnum	smallint	-	Support procedure number.
amproc	regproc	**proname** in PG_PROC	OID of the procedure.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The usual interpretation of the **amproclefttype** and **amprocrighttype** columns is that they identify the left and right input types of the operator(s) that a particular support procedure supports. For some access methods, these match the input data type(s) of the support procedure itself; for others not. There is a notion of "default" support procedures for an index, which are those with **amproclefttype** and **amprocrighttype** both equal to the index opclass's **opcintype**.

7.2.18.29 PG_ATTRDEF

PG_ATTRDEF records default values of columns.

Table 7-101 PG_ATTRDEF columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
adrelid	oid	Table to which a column belongs.
adnum	smallint	Number of columns.
adbin	pg_node_tree	Internal representation of the default value of the column.
adsrc	text	Internal representation of the readable default value.
adgencol	"char"	Specifies whether a column is a generated column. The value **'s'** indicates that the column is a generated column, and the value **'\0'** indicates that the column is an ordinary column. The default value is **'\0'**.
adbin_on_update	pg_node_tree	Internal representation of the attribute expression **on update current_timestamp** of the column.
adsrc_on_update	text	Internal representation of the readable attribute expression **on update current_timestamp**.
adbin_extend	pg_node_tree	Internal representation of a generated expression. In the scenario where a generated column references another generated column, the generated expression does not contain the generated column reference. The **adbin** column expands the referenced generated column to an equivalent form in the generated expression.

7.2.18.30 PG_ATTRIBUTE

PG_ATTRIBUTE records information about table columns.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-102 PG_ATTRIBUTE columns

Name	Type	Description
attrelid	oid	Table to which a column belongs.
attname	name	Column name. If this column indicates an index column, changing the column name in the original table does not update this column's information. To get the correct column name, use the pg_get_indexdef function.
atttypid	oid	Column type.
attstattarget	integer	Level of details of statistics collected for this column by ANALYZE. • The value **0** indicates that no statistics should be collected. • A negative value indicates that the system default statistic object is used. • The exact meaning of positive values is data type-dependent. For scalar data types, **attstattarget** is both the target number of "most common values" to collect, and the target number of histogram bins to create.
attlen	smallint	A copy of **typlen** in 12.2.15.78 PG_TYPE of this column's type.
attnum	smallint	Number of the column.
attndims	integer	Number of dimensions if the column is an array (**0** in other cases).
attcacheoff	integer	This column is always **-1** on disk. When it is loaded into a row descriptor in the memory, it may be updated to cache the offset of the columns in the row.
atttypmod	integer	Type-specific data supplied at the table creation time (for example, the maximum length of a **varchar** column). This column is used as the third parameter when passing to type-specific input functions and length coercion functions. The value will generally be **-1** for types that do not need ATTTYPMOD.
attbyval	boolean	A copy of **typbyval** in 12.2.15.78 PG_TYPE of this column's type.
attstorage	"char"	A copy of **typstorage** in 12.2.15.78 PG_TYPE of this column's type.
attalign	"char"	A copy of **typalign** in 12.2.15.78 PG_TYPE of this column's type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
attnotnull	boolean	A NOT NULL constraint. It is possible to change this column to enable or disable the constraint.
atthasdef	boolean	This column has a default value, in which case there will be a corresponding entry in the 12.2.15.25 PG_ATTRDEF system catalog that actually defines the value.
attisdropped	boolean	Indicates that this column has been dropped and is no longer valid. A dropped column is still physically present in the table but is ignored by the analyzer, so it cannot be accessed through SQL.
attislocal	boolean	Indicates that this column is locally defined in the relationship. Note that a column can be locally defined and inherited simultaneously.
attcmprmode	tinyint	Compressed modes for a specific column. The compressed mode includes: • **0**: not compressed (**ATT_CMPR_NOCOMPRESS**) • **1**: DELTA compression algorithm (**ATT_CMPR_DELTA**) • **2**: dictionary compression algorithm (**ATT_CMPR_DICTIONARY**) • **3**: prefix compression algorithm (**ATT_CMPR_PREFIX**) • **4**: digital string compression algorithm (**ATT_CMPR_NUMSTR**)
attinhcount	integer	Number of direct ancestors that this column has. A column with an ancestor cannot be dropped nor renamed.
attcollation	oid	Defined collation of a column.
attacl	aclitem[]	Permissions for column-level access.
attoptions	text[]	Column attribute. Currently, the following attributes are supported: • **n_distinct**: number of **distinct** values of a column (excluding subtables). • **n_distinct_inherited**: number of **distinct** values of a column (including subtables).
attfdwoptions	text[]	Column attribute of a foreign table. Currently, **dist_fdw**, **file_fdw**, and **log_fdw** do not use foreign table column attributes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
attinitdefval	bytea	**attinitdefval** stores the default value expression. ADD COLUMN in the row-store table must use this column.
attkvtype	tinyint	Key value type for a column. Value: **0**: default value (**ATT_KV_UNDEFINED**). **1**: dimension (**ATT_KV_TAG**). **2**: indicator (**ATT_KV_FIELD**). **3**: time column (**ATT_KV_TIMETAG**). **4**: hidden distribution key (**ATT_KV_HIDETAG**).
attidentity	"char"	Identity type of a column. Value: **'0'** or **'\0'**: The column is a non-IDENTITY column. **'a'**: The **IDENTITY** column attribute is of the **ALWAYS** type. **'d'**: The **IDENTITY** column attribute is of the **BY DEFAULT** type. **'n'**: The **IDENTITY** column attribute is of the **BY DEFAULT ON NULL** type.

7.2.18.31 PG_CAST

PG_CAST records the conversion relationship between data types.

Table 7-103 PG_CAST columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
castsource	oid	OID of the source data type.
casttarget	oid	OID of the target data type.
castfunc	oid	OID of the conversion function. The value **0** indicates that no conversion function is required.
castcontext	"char"	Conversion mode between the source and target data types. ● **'e'**: Only explicit conversion can be performed (using the CAST or :: syntax). ● **'i'**: Implicit conversion can be performed. ● **'a'**: Both implicit and explicit conversion can be performed when values are assigned to target columns.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
castmethod	"char"	Conversion method. • **'f'**: Conversion is performed using the specified function in the **castfunc** column. • **'b'**: Binary forcible conversion rather than the specified function in the **castfunc** column is performed between data types.

7.2.18.32 PG_CLASS

PG_CLASS records database objects and their relationship.

Table 7-104 PG_CLASS columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
relname	name	Name of an object, such as a table, index, or view.
relnamespace	oid	OID of the namespace that contains the relationship.
reltype	oid	Data type that corresponds to the table's row type. The index is 0 because the index does not have PG_TYPE records.
reloftype	oid	OID of the composite type (**0** for other types).
relowner	oid	Owner of the relationship.
relam	oid	Access method used, such as B-tree, if the row is an index.
relfilenode	oid	Name of the on-disk file of this relationship (**0** if such file does not exist).
reltablespace	oid	Tablespace in which this relationship is stored. If the value is **0**, the default tablespace in this database is used. This column is meaningless if the relationship has no on-disk file.
relpages	double precision	Size of the on-disk representation of the table in pages (of size **BLCKSZ**). This is only an estimate used by the optimizer.
reltuples	double precision	Number of rows in the table. This is only an estimate used by the optimizer.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
relallvisible	integer	Number of pages marked as all visible in the table. This column is used by the optimizer for optimizing SQL execution. It is updated by VACUUM, ANALYZE, and a few DDL statements such as CREATE INDEX.
reltoastrelid	oid	OID of the TOAST table associated with the table (**0** if no TOAST table exists). The TOAST table stores large columns "offline" in a secondary table.
reltoastidxid	oid	OID of the index for a TOAST table (**0** for a table other than a TOAST table)
reldeltarelid	oid	Not supported currently.
reldeltaidx	oid	Not supported currently.
relcudescrelid	oid	Not supported currently.
relcudescidx	oid	Not supported currently.
relhasindex	boolean	Its value is **true** if it is a table and has (or recently had) at least one index. It is set by CREATE INDEX but is not immediately cleared by DROP INDEX. If the VACUUM thread detects that a table has no index, it clears the **relhasindex** field and sets the value to **false**.
relisshared	boolean	Its value is **true** if the table is shared across all databases in the entire cluster. Otherwise, the value is **false**. Only certain system catalogs (such as PG_DATABASE) are shared.
relpersistence	"char"	• **p**: permanent table • **u**: non-log table • **t**: temporary table • **g**: global temporary table
relkind	"char"	• **r**: ordinary table. • **i**: index. • **G**: global secondary index. • **s**: sequence. • **v**: view. • **c**: composite type. • **t**: TOAST table. • **f**: foreign table. • **m**: materialized view. • **e**: STREAM object. • **o**: CONTVIEW object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
relnatts	smallint	Number of user columns in the relationship (excluding system columns). 12.2.15.26 PG_ATTRIBUTE has the same number of rows as the user columns.
relchecks	smallint	Number of CHECK constraints in the table. For details, see the system catalog 12.2.15.30 PG_CONSTRAINT.
relhasoids	boolean	Its value is **true** if an OID is generated for each row of the relationship. Otherwise, the value is **false**.
relhaspkey	boolean	Its value is **true** if the table has (or once had) a primary key. Otherwise, the value is **false**.
relhasrules	boolean	Its value is **true** if the table has rules. For details, see the system catalog 12.2.15.63 PG_REWRITE.
relhastriggers	boolean	The value is **true** if the table has (or once had) triggers. Triggers of the table and view are recorded in the system catalog 12.2.15.72 PG_TRIGGER.
relhassubclass	boolean	Its value is **true** if the table has (or once had) any inheritance child table. Otherwise, the value is **false**.
relcmprs	tinyint	Specifies whether the compression feature is enabled for the table. Note that only batch insertion triggers compression, so ordinary CRUD does not trigger compression. • **0**: Tables that do not support compression (primarily system catalogs, on which the compression attribute cannot be modified). • **1**: The compression feature of the table data is NOCOMPRESS or has no specified keyword. • **2**: The compression feature of the table data is COMPRESS.
relhasclusterkey	boolean	Not supported currently.
relrowmovement	boolean	Specifies whether row movement is allowed when the partitioned table is updated. • **true**: Row movement is allowed. • **false**: Row movement is not allowed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
parttype	"char"	Specifies whether the table or index has the property of a partitioned table. • **p**: The table or index has the property of a partitioned table. • **n**: The table or index does not have the property of a partitioned table. • **s**: The table is a subpartitioned table.
relfrozenxid	xid32	All transaction IDs before this one have been replaced with a permanent ("frozen") transaction ID in the table. This column is used to track whether the table needs to be vacuumed to prevent transaction ID wraparound (or to allow PG_CLOG to be shrunk). The value is **0** (**InvalidTransactionId**) if the relationship is not a table. To ensure compatibility with earlier versions, this column is reserved. The **relfrozenxid64** column is added to record the information.
relacl	aclitem[]	Access permissions. For details about the ACLItem type, see **ACLItem Type**. The command output of the query is as follows: *user1*=*privs*/*user2* indicates that the permission granted by user 2 to user 1 is **privs**. =*privs*/*user3* indicates that the permission granted to the **public** role by user 3 is **privs**. In the preceding command, user 1, user 2, and user 3 are the existing users or roles in the database, and **privs** indicates the permissions supported by the database. For details on permission descriptions, see **Table 7-105**.
reloptions	text[]	Method of accessing objects such as tables, indexes, and views, represented as a string in the "keyword=value" format.
relreplident	"char"	Identifier of a decoding column in logical decoding. • **d**: default (primary key, if any). • **n**: none. • **f**: all columns. • **i**: The indisreplident of the index is specified or the default index is used.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
relfrozenxid64	xid	All transaction IDs before this one have been replaced with a permanent ("frozen") transaction ID in the table. This column is used to track whether the table needs to be vacuumed to prevent transaction ID wraparound (or to allow PG_CLOG to be shrunk). The value is **0** (**InvalidTransactionId**) if the relationship is not a table. For a global temporary table, this column is meaningless. You can use the pg_catalog.gs_global_list_gtt_relfrozenxids function to view **relfrozenxid64** of the global temporary table of each session.
relbucket	oid	Specifies whether the current catalog contains hash bucket shards. A valid OID points to the specific shard information recorded in the PG_HASHBUCKET catalog. **NULL** indicates that hash bucket shards are not included.
relbucketkey	int2vector	Hash partition key information. **NULL** indicates that the column information is not included.
relminmxid	xid	All multi-transaction IDs before this one have been replaced with a transaction ID in the table. This is used to track whether the table needs to be vacuumed in order to prevent multi-transaction IDs wraparound or to allow PG_CLOG to be shrunk. The value is **0** (**InvalidTransactionId**) if the relationship is not a table.

Table 7-105 Description of permissions

Parameter	Description
r	SELECT (read)
w	UPDATE (write)
a	INSERT (insert)
d	DELETE
D	TRUNCATE
x	REFERENCES
t	TRIGGER
X	EXECUTE
U	USAGE

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Description
C	CREATE
c	CONNECT
T	TEMPORARY
A	ALTER
P	DROP
m	COMMENT
i	INDEX
v	VACUUM
*	Authorization options for preceding permissions

7.2.18.33 PG_COLLATION

PG_COLLATION describes available collations, which are essentially mappings from an SQL name to OS locale categories.

Table 7-106 PG_COLLATION columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
collname	name	-	Collation name (unique per namespace and encoding).
collnamespace	oid	**oid** in PG_NAMESPACE	OID of the namespace that contains this collation.
collowner	oid	**oid** in PG_AUTHID	Owner of the collation.
collencoding	integer	-	Encoding in which the collation is applicable, or **–1** if it works for any encoding. It is compatible with PostgreSQL.
collcollate	name	-	**LC_COLLATE** for this collation object.
collctype	name	-	**LC_CTYPE** for this collation object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
collpadattr	text	-	Collation padding attribute. • **NULL**: not applicable. • **NO PAD**: no padding. • **PAD SPACE**: blank spaces padded at the end.
collisdef	boolean	-	Determines whether the collation is the default collation of the character set.
id	integer	-	Not supported in the current version.

7.2.18.34 PG_CONSTRAINT

PG_CONSTRAINT records CHECK, PRIMARY KEY, UNIQUE, and FOREIGN KEY constraints on tables.

Table 7-107 PG_CONSTRAINT columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
conname	name	Constraint name (not necessarily unique).
connamespace	oid	OID of the namespace that contains the constraint.
contype	"char"	• **c**: CHECK constraint. • **p**: PRIMARY KEY constraint. • **u**: UNIQUE constraint. • **t**: trigger constraint. • **x**: mutual exclusion constraint. • **f**: FOREIGN KEY constraint. • **s**: clustering constraint. • **i**: invalid constraint.
condeferrable	Boolean	Specifies whether the constraint is deferrable. • **true**: yes. • **false**: no.
condeferred	Boolean	Specifies whether the constraint can be deferrable by default. • **true**: yes. • **false**: no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
convalidated	Boolean	Specifies whether the constraint is valid. Currently, it can be set to **false** only for FOREIGN KEY and CHECK constraints. • **true**: valid • **false**: invalid.
conrelid	oid	Table containing this constraint (**0** if it is not a table constraint).
contypid	oid	Domain containing this constraint (**0** if it is not a domain constraint).
conindid	oid	ID of the index associated with the constraint.
confrelid	oid	Referenced table if this constraint is a foreign key. Otherwise, the value is **0**.
confupdtype	"char"	Foreign key update action code. • **a**: no action. • **r**: restriction. • **c**: cascading. • **n**: The parameter is set to **null**. • **d**: The default value is used.
confdeltype	"char"	Foreign key deletion action code. • **a**: no action. • **r**: restriction. • **c**: cascading. • **n**: The parameter is set to **null**. • **d**: The default value is used.
confmatchtype	"char"	Foreign key match type. • **f**: full match. • **p**: partial match. • **u**: unspecified (The NULL value can be matched if **f** is specified.)
conislocal	Boolean	Specifies whether the constraint is defined locally for the relationship. • **true**: yes. • **false**: no.
coninhcount	integer	Number of direct inheritance parent tables that this constraint has. When the value is not **0**, the constraint cannot be deleted or renamed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
connoinherit	Boolean	Specifies whether the constraint can be inherited. • **true**: yes. • **false**: no.
consoft	Boolean	Specifies whether the column indicates an informational constraint. • **true**: yes. • **false**: no.
conopt	Boolean	Specifies whether you can use the informational constraint to optimize the execution plan. • **true**: yes. • **false**: no.
conkey	smallint[]	Column list of the constrained control if this column is a table constraint.
confkey	smallint[]	List of referenced columns if this column is a foreign key.
conpfeqop	oid[]	ID list of the equality operators for PK = FK comparisons if this column is a foreign key.
conppeqop	oid[]	ID list of the equality operators for PK = PK comparisons if this column is a foreign key.
conffeqop	oid[]	ID list of the equality operators for FK = FK comparisons if this column is a foreign key.
conexclop	oid[]	ID list of the per-column exclusion operators if this column is an exclusion constraint.
conbin	pg_node_tree	Internal representation of the expression if this column is a CHECK constraint.
consrc	text	Readable representation of the expression if this column is a CHECK constraint.
conincluding	smallint[]	Not for constraint, but will be included in the attribute column of **INDEX**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- **consrc** is not updated when referenced objects change and does not track new column names. You are advised to use pg_get_constraintdef() to extract the definition of a CHECK constraint.
- The **relchecks** column of **PG_CLASS** must be consistent with the number of CHECK constraints found for the given relationship on the table.

7.2.18.35 PG_CONVERSION

PG_CONVERSION describes encoding conversion information.

Table 7-108 PG_CONVERSION columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
conname	name	-	Conversion name (unique within a namespace).
connamespace	oid	**oid** in **PG_NAMESPACE**	OID of the namespace that contains this conversion.
conowner	oid	**oid** in **PG_AUTHID**	Owner of the conversion.
conforencoding	integer	-	Source encoding ID.
contoencoding	integer	-	Destination encoding ID.
conproc	regproc	**proname** in **PG_PROC**	Conversion procedure.
condefault	boolean	-	If this is the default conversion, the value is **true**. Otherwise, the value is **false**.

7.2.18.36 PG_DATABASE

PG_DATABASE records information about available databases.

Table 7-109 PG_DATABASE columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
datname	name	Database name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
datdba	oid	Owner of the database, usually the user who created it.
encoding	integer	Character encoding for the database.
datcollate	name	Sequence used by the database.
datctype	name	Character type used by the database.
datistemplate	boolean	Specifies whether the database can be used as a template database. ● **true**: allowed. ● **false**: not allowed.
datallowconn	boolean	This column is used to protect the **template0/templatea** database from being altered. ● **true**: Users can connect to the database. ● **false**: No user can connect to the database.
datconnlimit	integer	Maximum number of concurrent connections allowed on this database. The value **-1** indicates no limit.
datlastsysoid	oid	Last system OID in the database.
datfrozenxid	xid32	Tracks whether the database needs to be vacuumed to prevent transaction ID wraparound. This column is discarded in the current version. To ensure compatibility with earlier versions, this column is reserved. The **datfrozenxid64** column is added to record the information.
dattablespace	oid	Default tablespace of the database.
datcompatibility	name	Database compatibility mode Currently, four compatible modes are supported: PG, ORA, MYSQL, and TD.
datacl	aclitem[]	Access permission
datfrozenxid64	xid	Tracks whether the database needs to be vacuumed to prevent transaction ID wraparound.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
datminmxid	xid	All multi-transaction IDs before this one have been replaced with a transaction ID in the database. This is used to track whether the database needs to be vacuumed in order to prevent transaction IDs wraparound or to allow PG_CLOG to be shrunk. It is the minimum value of **relminmxid** in 12.2.15.28 PG_CLASS of all tables in the database.
dattimezone	name	Database time zone. The default is PRC.
dattype	"char"	Database type. **"D"** indicates an ordinary database.

7.2.18.37 PG_DEPEND

PG_DEPEND records the dependency between database objects. This information allows **DROP** commands to find which other objects must be dropped by **DROP CASCADE** or prevent dropping in the **DROP RESTRICT** case.

See also **PG_SHDEPEND**, which performs a similar functionality for dependencies involving objects that are shared across a database cluster.

Table 7-110 PG_DEPEND columns

Name	Type	Reference	Description
classid	oid	**oid** in PG_CLASS	OID of the system catalog where a dependent object resides.
objid	oid	Any OID column	OID of the dependent object.
objsubid	integer	-	Column number for a table column (**objid** and **classid** refer to the table itself); **0** for all other object types.
refclassid	oid	**oid** in PG_CLASS	OID of the system catalog where a referenced object resides.
refobjid	oid	Any OID column	OID of the referenced object.
refobjsubid	integer	-	Column number for a table column (**refobjid** and **refclassid** refer to the table itself); **0** for all other object types.
deptype	"char"	-	A code defining the specific semantics of this dependency.
depsrc	text	-	Name of the referenced object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

In all cases, a PG_DEPEND entry indicates that the referenced object cannot be dropped without also dropping the dependent object. However, there are several subflavors identified by **deptype**:

- DEPENDENCY_NORMAL (n): A normal relationship between separately created objects. The dependent object can be dropped without affecting the referenced object. The referenced object can only be dropped by specifying CASCADE, in which case the dependent object is dropped too. Example: a table column has a normal dependency on its data type.
- DEPENDENCY_AUTO (a): The dependent object can be dropped separately from the referenced object, and should be automatically dropped (regardless of RESTRICT or CASCADE mode) if the referenced object is dropped. Example: a named constraint on a table is made autodependent on the table, so that it will go away if the table is dropped.
- DEPENDENCY_INTERNAL (i): The dependent object was created as part of creation of the referenced object, and is only a part of its internal implementation. A DROP of the dependent object will be disallowed outright (We'll tell the user to issue a DROP against the referenced object, instead). A DROP of the referenced object will be propagated through to drop the dependent object whether CASCADE is specified or not. Example: A trigger created to enforce a FOREIGN KEY constraint is made internally dependent on the constraint's **PG_CONSTRAINT** entry.
- DEPENDENCY_EXTENSION (e): The dependent object is a member of the extension of the referenced object (see **PG_EXTENSION**). The dependent object can be dropped only via DROP EXTENSION on the referenced object. Functionally this dependency type acts the same as an internal dependency, but it is kept separate for clarity and to simplify GS_DUMP.
- DEPENDENCY_PIN (p): There is no dependent object; this type of entry is a signal that the system itself depends on the referenced object, and so that object must never be dropped. Entries of this type are created only by **initdb**. The columns for the dependent object contain zeroes.
- DEPENDENCY_COMPILE (c): When the dependent object is a view, this records the dependency relationships with referenced objects during compilation. For example, if a table column is used in the view definition, the compilation dependency of the view on the table is recorded during view creation and recompilation.

7.2.18.38 PG_DESCRIPTION

PG_DESCRIPTION records optional descriptions (comments) for each database object. Descriptions of many built-in system objects are provided in the initial contents of PG_DESCRIPTION.

See also **PG_SHDESCRIPTION**, which provides a similar functionality for descriptions involving objects that are shared within the entire cluster.

Table 7-111 PG_DESCRIPTION columns

Name	Type	Reference	Description
objoid	oid	Any OID column	OID of the object that this description pertains to.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
classoid	oid	**oid** in PG_CLASS	OID of the system catalog where the object appears.
objsubid	integer	-	Column number for a comment on a table column (**objoid** and **classoid** refer to the table itself); **0** for all other object types.
description	text	-	Arbitrary text that serves as the description of the object.

7.2.18.39 PG_DIRECTORY

PG_DIRECTORY stores directory objects added by users. You can execute the **CREATE DIRECTORY** statement to add records to this system catalog. When **enable_access_server_directory** is set to **off**, only the initial user can create directory objects. When **enable_access_server_directory** is set to **on**, users with the SYSADMIN permission and users inheriting the permissions of the built-in role gs_role_directory_create can create directory objects. Common users can access this system catalog only after being authorized.

Table 7-112 PG_DIRECTORY columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
dirname	name	Name of a directory object.
owner	oid	Owner of a directory object.
dirpath	text	Directory path.
diracl	aclitem[]	Access permissions.

7.2.18.40 PG_ENUM

PG_ENUM contains entries showing the values and labels for each enumerated type. The internal representation of a given enumerated value is actually the OID of its associated row in PG_ENUM.

Table 7-113 PG_ENUM columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
enumtypid	oid	**oid** in PG_TYPE	OID of the PG_TYPE entry owning this enumerated value.
enumsortorder	real	-	Sort position of this enumerated value within its enumerated type.
enumlabel	name	-	Textual label for this enumerated value.

The OIDs for PG_ENUM rows follow a special rule: even-numbered OIDs are guaranteed to be ordered in the same way as the sort ordering of their enumerated type. If two even OIDs belong to the same enumerated type, the smaller OID must have the smaller **enumsortorder** value. Odd-numbered OID values need bear no relationship to the sort order. This rule allows the enumerated comparison routines to avoid catalog lookups in many common cases. The routines that create and alter enumerated types attempt to assign even OIDs to enumerated values whenever possible.

When an enumerated type is created, its members are assigned sort-order positions from 1 to *n*. However, members added later might be given negative or fractional values of **enumsortorder**. The only requirement on these values is that they be correctly ordered and unique within each enumerated type.

7.2.18.41 PG_EXTENSION

PG_EXTENSION records information about the installed extensions. By default, GaussDB has the following extensions: plpgsql, DIST_FDW, FILE_FDW, GC_FDW, DBLINK_FDW, PACKAGES, ROACH_API, STREAMING, TSDB, GSREDISTRIBUTE, SECURITY_PLUGIN, GSSTAT_PLUGIN, NUMERIC_ENHANCE, PKG_DBE_RAW, PKG_DBE_OUTPUT, PKG_DBE_UTILITY, PKG_DBE_XML, PKG_DBE_OBFUSCATION_TOOLKIT, PKG_ILM, PKG_DBE_XMLGEN, PKG_DBE_STATS, PKG_DBE_DESCRIBE, and PKG_DBE_LICENSE.

Table 7-114 PG_EXTENSION

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
extname	name	Extension name.
extowner	oid	Owner of the extension.
extnamespace	oid	Namespace containing the extension's exported objects.
extrelocatable	Boolean	Specifies whether the extension can be relocated to another namespace. The value can be **true** (yes) or **false** (no).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
extversion	text	Version number of the extension.
extconfig	oid[]	Configuration information about the extension.
extcondition	text[]	Filter conditions for the extension's configuration information.

7.2.18.42 PG_EXTENSION_DATA_SOURCE

PG_EXTENSION_DATA_SOURCE records information about external data sources. An external data source contains information about an external database, such as its password encoding. It is mainly used with Extension Connector. By default, only system administrators can query this system catalog. Due to specification changes, the current version no longer supports this feature. Do not use it.

Table 7-115 PG_EXTENSION_DATA_SOURCE columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
srcname	name	-	Name of an external data source.
srcowner	oid	PG_AUTHID.oid	Owner of a foreign data source.
srctype	text	-	Type of an external data source. It is **NULL** by default.
srcversion	text	-	Version of an external data source. It is **NULL** by default.
srcacl	aclitem[]	-	Access permissions.
srcoptions	text[]	-	Option used for foreign data sources, expressed in a string in the format of *keyword*=*value*.

7.2.18.43 PG_FOREIGN_DATA_WRAPPER

PG_FOREIGN_DATA_WRAPPER records foreign data wrapper definitions. A foreign data wrapper is the mechanism by which external data, residing on foreign servers, is accessed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-116 PG_FOREIGN_DATA_WRAPPER columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
fdwname	name	-	Name of a foreign data wrapper.
fdwowner	oid	**oid** in **PG_AUTHID**	Owner of a foreign data wrapper.
fdwhandler	oid	**oid** in **PG_PROC**	References a handler function that is responsible for supplying execution routines for a foreign data wrapper. The value is **0** if no handler is provided.
fdwvalidator	oid	**oid** in **PG_PROC**	References a validator function that is responsible for checking the validity of the options given to the foreign data wrapper, as well as options for foreign servers and user mappings using the foreign data wrapper. The value is **0** if no validator is provided.
fdwacl	aclitem[]	-	Access permissions.
fdwoptions	text[]	-	Foreign data wrapper specific option, expressed in a string in the format of keyword=value.

7.2.18.44 PG_FOREIGN_SERVER

PG_FOREIGN_SERVER records foreign server definitions. A foreign server describes a source of external data, such as a remote server. Foreign servers are accessed via foreign data wrappers.

Table 7-117 PG_FOREIGN_SERVER columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
srvname	name	-	Name of a foreign server.
srvowner	oid	**oid** in **PG_AUTHID**	Owner of the foreign server.
srvfdw	oid	**oid** in **PG_FOREIGN_DATA_WRAPPER**	OID of the foreign data wrapper of the foreign server.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
srvtype	text	-	Type of the server (optional).
srvversion	text	-	Version of the server (optional).
srvacl	aclitem[]	-	Access permissions.
srvoptions	text[]	-	Option used for foreign servers, expressed in a string in the format of keyword=value.

7.2.18.45 PG_FOREIGN_TABLE

PG_FOREIGN_TABLE records auxiliary information about foreign tables.

Table 7-118 PG_FOREIGN_TABLE columns

Name	Type	Description
ftrelid	oid	ID of a foreign table.
ftserver	oid	Server where the foreign table is located.
ftwriteonly	boolean	Specifies whether the foreign table is writable. Values are as follows: • **t** (true): yes. • **f** (false): no.
ftoptions	text[]	Options of the foreign table. For details, see the **7.13.8.22 CREATE FOREIGN TABLE** syntax description.

7.2.18.46 PG_HASHBUCKET

PG_HASHBUCKET records hash bucket information.

Table 7-119 PG_HASHBUCKET columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
bucketid	oid	Node group bound to the database where the PG_HASHBUCKET system catalog resides on the CN. This column is empty on the DN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
bucketcnt	integer	This parameter is not used on the CN. It indicates the number of buckets owned by the current DN on the DN.
bucketmapsize	integer	This parameter is not used in the current version.
bucketref	integer	Reserved column with **1** as its default value.
bucketvector	oidvector_extend	This parameter is not used on the CN. It indicates the list of buckets owned by the current DN on the DN.
bucketmap	text	Stores the mapping between logical buckets and physical buckets, that is, the mapping between 16384 and 1024.
bucketversion	oidvector_extend	Records the version number of the information that changes during subsequent hash bucket scale-out.
bucketcsn	text	Maximum CSN of each bucket on the source node before hash bucket redistribution, which is used to determine the visibility of the new node.
bucketxid	text	**next_xid** set for bringing a new node online during hash bucket scaling. This parameter is used to check whether the value is within the threshold range.
bucketclogmode	smallint	Clog directory type used by a bucket table. • **Empty**: The **pg_clog/**<*bucketid*> directory is used. • **1**: The **pg_clog/base/**<*dbid*>**/**<*bucketid*> directory is used.
cbibuckets	oidvector_extend	Records the list of buckets that are being rebuilt online by the CBI.

7.2.18.47 PG_INDEX

PG_INDEX records part of index information. The rest is mostly recorded in PG_CLASS.

Table 7-120 PG_INDEX columns

Name	Type	Description
indexrelid	oid	OID of the **12.2.15.28 PG_CLASS** entry for the index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
indrelid	oid	OID of the 12.2.15.28 PG_CLASS entry for the table that uses the index.
indnatts	smallint	Number of columns in the index.
indisunique	Boolean	Specifies whether the index is unique. • **true**: The index is unique. • **false**: The index is not unique.
indisprimary	Boolean	Specifies whether the index is the primary key of the table. • **true**: The index is the primary key of the table. **indisunique** should always be **true** when the value of this column is **true**. • **false**: The index is not the primary key of the table.
indisexclusion	Boolean	Specifies whether the index supports exclusive constraints. • **true**: The index supports exclusive constraints. • **false**: The index does not support exclusive constraints.
indimmediate	Boolean	Specifies whether to check the uniqueness of data to be inserted. • **true**: The uniqueness check is performed immediately when data is inserted. • **false**: The uniqueness check is not performed when data is inserted.
indisclustered	Boolean	Specifies whether the table is clustered on the index. • **true**: The table is clustered on the index. • **false**: The table is not clustered on the index.
indisusable	Boolean	Specifies whether the index is available for insert and select operations. • **true**: The index is available for insert and select operations. • **false**: The index is unavailable for insert and select operations.
indisvalid	Boolean	• **true**: The index can be used for query. • **false**: The index is possibly incomplete and must still be modified by INSERT or UPDATE operations, but it cannot be securely used for queries. If it is a unique index, the uniqueness property is also not **true**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
indcheckxmin	Boolean	• **true**: Queries must not use the index until the xmin of this row in PG_INDEX is lower than their **TransactionXmin**, because the table may contain broken HOT chains with incompatible rows that they can see. • **false**: Indexes can be used for query.
indisready	Boolean	• **true**: The index is available for inserting data. • **false**: The index is ignored when data is inserted or modified.
indkey	int2vector	This is an array of **indnatts** values indicating that this index creates table columns. For example, a value of **1 3** indicates that the first and the third columns make up the index key. The value **0** in this array indicates that the corresponding index attribute is an expression over the table columns, rather than a simple column reference.
indcollation	oidvector	OID of the collation corresponding to each index column. For details, see **PG_COLLATION**.
indclass	oidvector	For each column in the index key, this contains the OID of the operator class to use. For details, see **12.2.15.53 PG_OPCLASS**.
indoption	int2vector	Array of values that store per-column flag bits. The meaning of the bits is defined by the index's access method.
indexprs	pg_node_tree	Expression trees (in **nodeToString()** representation) for index attributes that are not simple column references. It is a list with one element for each zero entry in indkey. The value is null if all index attributes are simple references.
indpred	pg_node_tree	Expression tree (in **nodeToString()** representation) for partial index predicate. If the index is not a partial index, this column is an empty string.
indisreplident	Boolean	Specifies whether the column of this index is a decoded column of logical decoding. • **true**: The column of this index is a decoded column of logical decoding. • **false**: The column of this index is not a decoded column of logical decoding.
indnkeyatts	smallint	Total number of columns in the index. The columns that exceed the value of **indnatts** are not involved in the index query.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
indcctmpid	oid	OID of the temporary table when the Ustore builds indexes online.
indisvisible	Boolean	• **true**: The index status is visible, that is, the optimizer can use the index. • **false**: The index is invisible. If the value of **enable_invisible_indexes** is **off**, the optimizer cannot use the index. If the value of **enable_invisible_indexes** is **on**, the optimizer can use the index.

7.2.18.48 PG_INHERITS

PG_INHERITS records information about table inheritance hierarchies. There is one entry for each direct child table in the database. Indirect inheritance can be determined by following chains of entries.

Table 7-121 PG_INHERITS columns

Name	Type	Reference	Description
inhrelid	oid	**oid** in **PG_CLASS**	OID of a child table
inhparent	oid	**oid** in **PG_CLASS**	OID of a parent table.
inhseqno	integer	-	If there is more than one direct parent for a child table (multiple inheritances), this number tells the order in which the inherited columns are to be arranged. The count starts at 1.

7.2.18.49 PG_JOB

PG_JOB records detailed information about jobs created by users. Dedicated threads poll the system catalog PG_JOB and trigger jobs based on scheduled job execution time, and update job status in PG_JOB. This system catalog belongs to the Shared Relation category. All job records are visible to all databases. Common users can access this system catalog only after being authorized.

Table 7-122 PG_JOB columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
job_id	bigint	Job ID, which is the primary key and is unique (with a unique index).
current_postgres_pid	bigint	If the current job has been executed, the thread ID of this job is recorded. The default value is **-1**, indicating that the job has not yet been executed.
log_user	name	Username of the creator.
priv_user	name	Username of the job executor.
dbname	name	Name of the database in which the job will be executed.
node_name	name	CN on which the job will be executed.
job_status	"char"	Execution status of the current task. The default value is **'s'**. The options are as follows: ● **'r'**: running ● **'s'**: successfully finished ● **'f'**: job failed ● **'d'**: disable If a job fails to be executed for 16 consecutive times, **job_status** is automatically set to **'d'**, and no more attempt will be made on this job. Note: When you disable a scheduled job (by setting **job_queue_processes** to **0**), the thread that monitors the job execution is not started, and the job status will not be updated. You can ignore this status. Only when the scheduled job function is enabled (**job_queue_processes** is not set to **0**), the system updates the value of this column based on the real-time job status.
start_date	timestamp without time zone	Start time of the first job execution, accurate to millisecond.
next_run_date	timestamp without time zone	Time when a scheduled task is executed next time. The time is accurate to milliseconds.
failure_count	smallint	Number of times the job has started and failed. If a job fails to be executed for 16 consecutive times, no more attempt will be made on it.
interval	text	Job execution interval.
last_start_date	timestamp without time zone	Start time of the last job execution, accurate to millisecond.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
last_end_date	timestamp without time zone	End time of the last job execution, accurate to millisecond.
last_suc_date	timestamp without time zone	Start time of the last successful job execution, accurate to millisecond.
this_run_date	timestamp without time zone	Start time of the ongoing job execution, accurate to millisecond.
nspname	name	Name of the schema used for job execution.
job_name	text	Name of the DBE_SCHEDULER scheduled task.
end_date	timestamp without time zone	Expiration time of the DBE_SCHEDULER scheduled task, accurate to millisecond.
enable	Boolean	Enabling status of the DBE_SCHEDULER scheduled job. The options are as follows: • **true**: enabled • **false**: disabled
failure_msg	text	Error information about the latest task execution.

7.2.18.50 PG_JOB_PROC

PG_JOB_PROC records the content of each job in **PG_JOB**, including the PL/SQL code blocks and anonymous blocks. Storing such information in the system catalog PG_JOB and loading it to the shared memory will result in excessive memory usage. Therefore, such information is stored in a separate table and is retrieved when needed. Common users can access this system catalog only after being authorized.

Table 7-123 PG_JOB_PROC columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
job_id	integer	Foreign key, which is joined with **job_id** in the system catalog **PG_JOB**.
what	text	Job content, which is the program content in the DBE_SCHEDULER scheduled job.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
job_name	text	Name of the DBE_SCHEDULER scheduled job or program.

7.2.18.51 PG_LANGUAGE

PG_LANGUAGE registers programming languages. You can use them and APIs to write functions or stored procedures.

Table 7-124 PG_LANGUAGE columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
lanname	name	-	Name of the language.
lanowner	oid	**oid** in **PG_AUTHID**	Owner of the language.
lanispl	boolean	-	• **true**: user-defined language. • **false**: internal language, for example, SQL. Currently, gs_dump still uses this column to determine which languages need to be dumped, but this might be replaced by a different mechanism in the future.
lanpltrusted	boolean	-	• **true**: The language is trusted, which means that it is believed not to grant access to anything outside the normal SQL execution environment. • **false**: The language is untrusted. Only the initial user can create functions in untrusted languages.
lanplcallfoid	oid	**oid** in **PG_PROC**	For non-internal languages, this column references the language handler, which is a special function responsible for executing all functions that are written in the particular language.
laninline	oid	**oid** in **PG_PROC**	This column references a function responsible for executing "inline" anonymous code blocks (DO blocks). The value is **0** if inline blocks are not supported.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
lanvalidator	oid	**oid** in PG_PROC	This column references a language validator function responsible for checking the syntax and validity of new functions when they are created. The value is **0** if no validator is provided.
lanacl	aclitem[]	-	Access permission.

7.2.18.52 PG_NAMESPACE

PG_NAMESPACE records namespaces, that is, schema-related information. If the database object isolation attribute is enabled, users can view only the schema information that they have the permission to access.

Table 7-125 PG_NAMESPACE columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
nspname	name	Name of a namespace.
nspowner	oid	Owner of a namespace.
nsptimeline	bigint	Timeline when the namespace is created on the DN. This column is for internal use and valid only on the DN.
nspacl	aclitem[]	Access permission. For details, see **GRANT** and **REVOKE**.
in_redistribution	"char"	Specifies whether the content is in the redistribution state.
nspblockchain	Boolean	Specifies whether the mode is the tamper-proof mode. • **true**: The mode is the tamper-proof mode. • **false**: The mode is not the tamper-proof mode.
nspcollation	oid	Default collation of the namespace (a value may exist only when **sql_compatibility** is set to **'MYSQL'**).
nspgroupoid	oid	Node group. This is a single-node attribute. When a single-node table is implicitly created in this schema, it is created in the node group specified by this column by default.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.18.53 PG_OBJECT

PG_OBJECT records the creator, creation time, and last modification time of objects of specified types (ordinary tables, indexes, sequences, views, stored procedures, and functions).

Table 7-126 PG_OBJECT columns

Name	Type	Description
object_oid	oid	Object identifier.
object_type	"char"	Object type. • **r**: ordinary table • **i**: index • **s**: sequence • **v**: view • **p**: stored procedure and function • **S**: package header • **B**: package body
creator	oid	Object owner.
ctime	timestamp with time zone	Creation time of an object.
mtime	timestamp with time zone	Last modification time of an object. The modification operations include **ALTER**, **GRANT**, and **REVOKE**.
createcsn	bigint	CSN when an object is created.
changecsn	bigint	CSN when DDL operations are performed on a table or an index.
valid	Boolean	Validity of an object. **t** indicates valid, and **f** indicates invalid.
relchangecsn	bigint	CSN when the table definition is changed or the TRUNCATE DDL operation is performed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

- Objects created or modified during database initialization (initdb) cannot be recorded. **PG_OBJECT** does not contain these object records.
- A database upgraded to this version cannot record objects created before the upgrade. **PG_OBJECT** does not contain these object records.
- When the preceding two types of objects are modified again, the modification time (**mtime**) is recorded. Because the creation time of the objects cannot be obtained, **ctime** is empty.
- When an object created before the upgrade is modified again, the modification time (specified by **mtime**) is recorded. When DDL operations are performed on a table or an index, the transaction commit sequence number (specified by **changecsn**) of the transaction to which the table or index belongs is recorded. Because the creation time of the object cannot be obtained, **ctime** and **createcsn** are empty.
- The time recorded by **ctime** and **mtime** is the start time of the transaction to which the current operation belongs.
- The time of object modification due to capacity expansion is also recorded.
- **createcsn** and **changecsn** record the transaction commit sequence number of the transaction to which the current operation belongs.
- If the statement for creating an object has an undefined object, or the referenced object is modified or deleted, the object to be created will be invalid.

7.2.18.54 PG_OPCLASS

PG_OPCLASS defines index access method operator classes.

Each operator class defines semantics for index columns of a particular data type and a particular index access method. An operator class essentially specifies that a particular operator family is applicable to a particular indexable column data type. The set of operators from the family that are actually usable with the indexed column are data types that accept the left-hand column.

Table 7-127 PG_OPCLASS columns

Name	Type	Reference	Description
OID	OID	-	Row identifier (hidden attribute, which must be specified).
opcmethod	OID	**oid** in **PG_AM**	Index access method operator class served by an operator class.
opcname	name	-	Name of the operator class.
opcnamespace	OID	**oid** in **PG_NAMESPACE**	Namespace of the operator class.
opcowner	OID	**oid** in **PG_AUTHID**	Owner of the operator class.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
opcfamily	OID	**oid** in PG_OPFAMILY	Operator family containing the operator class.
opcintype	OID	**oid** in PG_TYPE	Data type that the operator class indexes.
opcdefault	Boolean	-	The value is **t** (true) if the operator class is the default for **opcintype**. Otherwise, the value is **f** (false).
opckeytype	OID	**oid** in PG_TYPE	Type of data stored in an index, or zero if same as **opcintype**.

An operator class's **opcmethod** must match the **opfmethod** of its containing operator family.

7.2.18.55 PG_OPERATOR

PG_OPERATOR records information about operators.

Table 7-128 PG_OPERATOR columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
oprname	name	-	Name of an operator.
oprnamespace	oid	**oid** in PG_NAMESPACE	OID of the namespace that contains the operator.
oprowner	oid	**oid** in PG_AUTHID	Owner of the operator.
oprkind	"char"	-	• **b**: infix (both sides) • **l**: prefix (left side) • **r**: suffix (right side)
oprcanmerge	boolean	-	Specifies whether the operator supports merge joins. • **t (true)**: yes • **f (false)**: no

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
oprcanhash	boolean	-	Specifies whether the operator supports hash joins. • **t (true)**: yes • **f (false)**: no
oprleft	oid	**oid** in **PG_TYPE**	Type of the left operand.
oprright	oid	**oid** in **PG_TYPE**	Type of the right operand.
oprresult	oid	**oid** in **PG_TYPE**	Type of the result.
oprcom	oid	**oid** in **PG_OPERATOR**	If it exists, the value is the exchange character of this operator. If it does not exist, the value is **0**.
oprnegate	oid	**oid** in **PG_OPERATOR**	If it exists, the value is the invertor of this operator. If it does not exist, the value is **0**.
oprcode	regproc	**proname** in **PG_PROC**	Function that implements the operator.
oprrest	regproc	**proname** in **PG_PROC**	Restriction selectivity estimation function for the operator.
oprjoin	regproc	**proname** in **PG_PROC**	Join selectivity estimation function for the operator.

7.2.18.56 PG_OPFAMILY

PG_OPFAMILY defines operator families.

Each operator family is a collection of operators and associated support routines that implement semantics specified for a particular index access method. Furthermore, the operators in a family are all compatible, in a way that is specified by the access method. The operator family allows cross-data-type operators to be used with indexes and to be reasoned about using knowledge of access method semantics.

Table 7-129 PG_OPFAMILY columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
opfmethod	oid	**oid** in PG_AM	Index access method used by an operator family.
opfname	name	-	Name of the operator family.
opfnamespace	oid	**oid** in PG_NAMESPACE	Namespace of the operator family.
opfowner	oid	**oid** in PG_AUTHID	Owner of the operator family.

The majority of the information defining an operator family is not in its PG_OPFAMILY row, but in the associated rows in PG_AMOP, PG_AMPROC, and PG_OPCLASS.

7.2.18.57 PG_PLTEMPLATE

PG_PLTEMPLATE records template information for procedural languages.

Table 7-130 PG_PLTEMPLATE columns

Name	Type	Description
tmplname	name	Name of the language for which this template is used.
tmpltrusted	Boolean	The value is **true** if the language is considered trusted. Otherwise, the value is **false**.
tmpldbacreate	Boolean	The value is **true** if the language is created by the owner of the database. Otherwise, the value is **false**.
tmplhandler	text	Name of the call handler function.
tmplinline	text	Name of the anonymous block handler (**NULL** if no name of the block handler exists).
tmplvalidator	text	Name of the verification function (**NULL** if no verification function is available).
tmpllibrary	text	Path of the shared library that implements languages.
tmplacl	aclitem[]	Access permissions for template (not yet used).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.18.58 PG_PROC

PG_PROC stores information about functions or stored procedures.

Table 7-131 PG_PROC columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
proname	name	Function name.
pronamespace	oid	OID of the namespace that contains the function.
proowner	oid	Owner of the function.
prolang	oid	Implementation language or call API of the function.
procost	real	Estimated execution cost.
prorows	real	Estimated number of rows that are influenced.
provariadic	oid	Data type of parameter element.
protransform	regproc	Simplified call method for the function.
proisagg	Boolean	Specifies whether the function is an aggregate function. ● **t** (true): yes ● **f** (false): no
proiswindow	Boolean	Specifies whether the function is a window function. ● **t** (true): yes ● **f** (false): no
prosecdef	Boolean	Specifies whether the function is a security definer (or a setuid function). ● **t** (true): yes ● **f** (false): no
proleakproof	Boolean	Specifies whether the function has side effects. If no leakproof treatment is provided for parameters, the function throws errors. ● **t** (true): There is no side effect. ● **f** (false): There are side effects.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
proisstrict	Boolean	• **t** (true): When this function is used, the function does not call the function and returns null if the input parameter is null. • **f** (false): When this function is used, it is called even if the input parameter is null. Therefore, this function must process null input.
proretset	Boolean	Specifies whether the return value of the function is a set (that is, multiple values of the specified data type). • **true**: The return value of the function is a set. • **false**: The return value of the function is not a set.
provolatile	"char"	Specifies whether the function's result depends only on its input parameters, or is affected by outside factors. • **i**: immutable functions, which always deliver the same result for the same inputs. • **s**: stable functions, whose results (for fixed inputs) do not change within a scan. • **v**: volatile functions, whose results may change at any time. It is also used for functions with side effects, so that the engine cannot get optimized if volatile functions are called.
pronargs	smallint	Number of parameters.
pronargdefaults	smallint	Number of parameters that have default values.
prorettype	oid	Data type of return values.
proargtypes	oidvector	Array that stores the data types of function parameters. This array includes only input parameters (including **INOUT** parameters), and indicates the call signature (API) of the function.
proallargtypes	oid[]	Array that contains the data types of function parameters. This array includes all parameter types (including **OUT** and **INOUT** parameters); however, if all the parameters are **IN** parameters, this column is null. Note that array indexing is 1-based, whereas for historical reasons, **proargtypes** is indexed from 0.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
proargmodes	"char"[]	Array with the modes of the function parameters, encoded as follows: • **i** indicates the **IN** parameter. • **o** indicates the **OUT** parameter. • **b** indicates the **INOUT** parameter. • **v** indicates the **VARIADIC** parameter. If all the parameters are **IN** parameters, this column is null. Note that indexes correspond to positions of **proallargtypes**, not **proargtypes**.
proargnames	text[]	Array that stores the names of the function parameters. Parameters without a name are set to empty strings in the array. If none of the parameters have a name, this column is null. Note that indexes correspond to positions of **proallargtypes**, not **proargtypes**.
proargdefaults	pg_node_tree	Expression tree of the default value. This is the list of **pronargdefaults** elements.
prosrc	text	A definition that describes a function or stored procedure. In an interpreting language, it is the function source code, a connection symbol, a file name, or any body content specified when a function or stored procedure is created, depending on how a language or call is used.
probin	text	Additional information about how to call the function. Again, the interpretation is language-specific.
proconfig	text[]	Function's local settings for run-time configuration variables.
proacl	aclitem[]	Access permissions. For details, see **GRANT** and **REVOKE**.
prodefaultargpos	int2vector	Position of the input parameter of a function with a default value.
fencedmode	Boolean	Execution mode of a function, indicating whether the function is executed in fence or not fence mode. • **true**: fence mode. In this mode, the function is executed in the fork process. • **false**: not fence mode. For built-in functions, **fencedmode** is set to **false**, indicating the not fence mode.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
proshippable	Boolean	Specifies whether the function can be pushed down to DNs for execution. The default value is **false**. ● Functions of the **IMMUTABLE** type can always be pushed down to DNs. ● Functions of the **STABLE** or **VOLATILE** type can be pushed down to DNs only if their attribute is **SHIPPABLE**.
propackage	Boolean	Specifies whether the function supports overloading. The default value is **false**. ● **t** (true): supported. ● **f** (false): not supported.
prokind	"char"	Specifies whether the object is a function or a stored procedure. ● **'f'**: The object is a function. ● **'p'**: The object is a stored procedure.
proargsrc	text	Describes the parameter input strings of functions or stored procedures that are compatible with ORA syntax, including parameter comments. The default value is **NULL**.
proisprivate	Boolean	Specifies whether a function is a private function in the package. The default value is **false**.
propackageid	oid	OID of the package to which the function belongs. If the function is not in the package, the value is **0**.
proargtypesext	oidvector_extend	Data type array used to store function parameters when there are a large number of function parameters. This array includes only input parameters (including **INOUT** parameters), and indicates the call signature (API) of the function.
prodefaultargposext	int2vector_extend	Position of the input parameter with a default value when the function has a large number of parameters.
allargtypes	oidvector	Array for storing the data types of stored procedure parameters, including all parameters (input parameters, output parameters, and **INOUT** parameters) of the stored procedure.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
allargtypesext	oidvector_extend	Array for storing the data types of stored procedure parameters when the number of function parameters is greater than 666. The array contains all parameters (including input parameters, output parameters, and **INOUT** parameters).

NOTE

When a function is created, data is inserted into the PG_PROC catalog to update the index. When there are a large number of input and output parameters, the index length may exceed one-third of the page length. As a result, the error "Index row size xxx exceeds maximum xxx for index 'pg_proc_proname_all_args_nsp_index'" may be reported as expected. You can reduce the number of parameters to avoid this error.

7.2.18.59 PG_RANGE

PG_RANGE records information about range types, except for records of types in **PG_TYPE**.

Table 7-132 PG_RANGE columns

Name	Type	Reference	Description
rngtypid	oid	**oid** in **PG_TYPE**	OID of the range type.
rngsubtype	oid	**oid** in **PG_TYPE**	OID of the element type (subtype) of this range type.
rngcollation	oid	**oid** in **PG_COLLATION**	OID of the collation used for range comparisons (**0** if none).
rngsubopc	oid	**oid** in **PG_OPCLASS**	OID of the subtype's operator class used for range comparisons.
rngcanonical	regproc	**proname** in **PG_PROC**	Name of the function to convert a range value into canonical form (**0** if none).
rngsubdiff	regproc	**proname** in **PG_PROC**	Name of the function used to calculate the difference between two elements in a range. The return value of the function is of the double-precision type. If the function does not exist, the value of **rngsubdiff** is **0**.

If the element type is discrete, **rngcanonical** determines the collation used for the range type. If the element type is not collatable, **rngsubopc** determines the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

collation used for the range type. If the element type is collatable, **rngsubopc** and **rngcollation** determine the collation used for the range type.

7.2.18.60 PG_REPLICATION_ORIGIN

PG_REPLICATION_ORIGIN contains all created replication sources and is shared among all databases in a cluster. Each cluster has only one copy of PG_REPLICATION_ORIGIN, not one copy per database instance.

Table 7-133 PG_REPLICATION_ORIGIN columns

Name	Type	Description
roident	oid	Unique replication source identifier within a cluster.
roname	text	External user-defined replication source name.

7.2.18.61 PG_RESOURCE_POOL

PG_RESOURCE_POOL provides information about database resource pools.

Table 7-134 PG_RESOURCE_POOL columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
respool_name	name	Name of the resource pool.
mem_percent	integer	Percentage of the memory configuration.
cpu_affinity	bigint	Value of cores bound to the CPU.
control_group	name	Name of the Cgroup where the resource pool is located.
active_statements	integer	Reserved column.
max_dop	integer	Maximum scanning concurrency during data redistribution. This column is used only for scaling.
memory_limit	name	Maximum memory of the resource pool.
parentid	oid	OID of the parent resource pool.
io_limits	integer	Upper limit of IOPS. The unit is 10 thousand per second.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
io_priority	name	Specifies the I/O priority set for jobs that consume many I/O resources. It takes effect when the I/O usage reaches 90%.
nodegroup	name	Indicates the name of the node group to which the resource pool belongs.
is_foreign	Boolean	Specifies whether the resource pool can be used for users outside the logical database. • **true**: The resource pool controls the resources of common users who do not belong to the current resource pool. • **false**: The resources of common users who do not belong to the current resource pool are not controlled.
max_worker	integer	Concurrency in a table during data redistribution. This column is used only for scaling.
max_connections	integer	Maximum number of connections that can be used by a resource pool.
max_dynamic_memory	name	Maximum dynamic memory that can be used by a resource pool.
max_shared_memory	name	Maximum shared memory that can be used by a resource pool.
max_concurrency	integer	Maximum number of concurrent requests that can be used by a resource pool.

7.2.18.62 PG_REWRITE

PG_REWRITE records rewrite rules defined for tables and views.

Table 7-135 PG_REWRITE columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
rulename	name	Rule name.
ev_class	oid	Name of the table that uses the rule.
ev_attr	smallint	Column to which this rule applies (always **0** to indicate the entire table).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
ev_type	"char"	Event type for the rule. • **1**: SELECT • **2**: UPDATE • **3**: INSERT • **4**: DELETE
ev_enabled	"char"	Controls the mode in which the rule is triggered. • **O**: The rule is triggered in origin and local modes. • **D**: The rule is disabled. • **R**: The rule is triggered in replica mode. • **A**: The rule is always triggered.
is_instead	Boolean	The value is **true** if the rule is of the **INSTEAD** type. Otherwise, the value is **false**.
ev_qual	pg_node_tree	Expression tree (in the form of a nodeToString() representation) for the rule's qualifying condition.
ev_action	pg_node_tree	Query tree (in the form of a nodeToString() representation) for the rule's action.
ev_action_def	text	Saves the original SELECT statement when the view is defined.

7.2.18.63 PG_SET

PG_SET records metadata defined by the SET data type.

Currently, this feature is not supported in the distributed mode.

7.2.18.64 PG_SHDEPEND

PG_SHDEPEND records the dependency between database objects and shared objects, such as roles. Based on this information, GaussDB can ensure that those objects are unreferenced before attempting to delete them.

See also **PG_DEPEND**, which provides a similar functionality for dependencies involving objects within a single database.

PG_SHDEPEND is shared among all databases of cluster. That is, there is only one PG_SHDEPEND for each cluster, not for each database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-136 PG_SHDEPEND columns

Name	Type	Reference	Description
dbid	oid	**oid** in **PG_DATABASE**	OID of the database where a dependent object is (**0** for a shared object).
classid	oid	**oid** in **PG_CLASS**	OID of the system catalog where a dependent object resides.
objid	oid	Any OID column	OID of the dependent object.
objsubid	integer	-	Column number of a table column (the table to which the column points can be determined based on **objid** and **classid**). The value is **0** for all other object types.
refclassid	oid	**oid** in **PG_CLASS**	OID of the system catalog the referenced object is in (must be a shared catalog).
refobjid	oid	Any OID column	OID of the referenced object.
deptype	"char"	-	Code segment defining the specific semantics of this dependency relationship. The value can be **o**, **a**, **p**, **d**, or **l**.
objfile	text	-	Path of the user-defined C function library file.

In all cases, a PG_SHDEPEND entry indicates that the referenced object cannot be dropped without also dropping the dependent object. However, there are several subflavors identified by **deptype**:

- SHARED_DEPENDENCY_OWNER (o)

  The referenced object (which must be a role) is the owner of the dependent object.

- SHARED_DEPENDENCY_ACL (a)

  The referenced object (which must be a role) is mentioned in the access control list (ACL) of the dependent object. SHARED_DEPENDENCY_ACL does not add a record to the owner of the object because the owner will have a SHARED_DEPENDENCY_OWNER record.

- SHARED_DEPENDENCY_PIN (p)

  This type of record indicates that the system itself depends on the depended object. Therefore, such an object cannot be deleted. Entries of this type are created only by **initdb**. The columns for the dependent object contain zeroes.

- SHARED_DEPENDENCY_ DBPRIV(d)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The referenced object (must be a role) has the ANY permission on the dependent object (the specified OID of the dependent object corresponds to a row in the **12.2.10.1 GS_DB_PRIVILEGE** system catalog).

- SHARED_DEPENDENCY_ SECLABEL(l)

  The referenced object (must be a security label) is applied to the dependent object.

7.2.18.65 PG_SHDESCRIPTION

PG_SHDESCRIPTION records optional comments for shared database objects. Descriptions can be manipulated with the **COMMENT** command and viewed with the **\d** command.

See also PG_DESCRIPTION, which provides a similar functionality for descriptions involving objects within a single database.

PG_SHDESCRIPTION is shared among all databases of cluster. That is, there is only one PG_SHDESCRIPTION for each cluster, not for each database.

Table 7-137 PG_SHDESCRIPTION columns

Name	Type	Reference	Description
objoid	oid	Any OID column	OID of the described object.
classoid	oid	**oid** in **PG_CLASS**	OID of the system catalog where the object appears.
description	text	-	Description of the object.

7.2.18.66 PG_STATISTIC

PG_STATISTIC stores statistics about tables and index columns in a database. By default, only users with the SYSADMIN permission can access the system catalog. Common users can access this system catalog only after being authorized.

Table 7-138 PG_STATISTIC columns

Name	Type	Description
starelid	oid	Table or index that the described column belongs to.
starelkind	"char"	Type of an object.
staattnum	smallint	Number of the described column in the table, starting from 1.
stainherit	boolean	Specifies whether to collect statistics for objects that have inheritance relationship.
stanullfrac	real	Percentage of column entries that are null.
stawidth	integer	Average stored width, in bytes, of non-null entries.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
stadistinct	real	Number of distinct, non-null data values in the column for all DNs. ● A value greater than 0 is the actual number of distinct values. ● A value less than 0 is the ratio of the **distinct** value to the total number of rows. For example, if **stadistinct** is **-0.5**, the actual **distinct** value is the total number of rows multiplied by 0.5. ● The value **0** indicates that the number of distinct values is unknown.
stakindN	smallint	**stakindN** records the type of statistics. N is a code, indicating that the statistics of this type are stored in the *N*th slot of the **pg_statistic** row. Value range of *N* ranges from 1 to 5, for example, stakind1, stakind2, or stakind3. For **stakindN**, the numbers indicate the following information: ● **1**: MCV ● **2**: histogram ● **3**: correlation ● **4**: MCELEM (common element, used for arrays) ● **5**: histogram of the number of different elements (used for arrays)
staopN	oid	Operator used to generate the statistics stored in slot *N*. For example, a histogram slot shows the < operator that defines the sort order of the data. The value of *N* ranges from 1 to 5.
stanumbersN	real[]	Numerical statistics of the appropriate type for slot *N*. The value is **NULL** if the slot does not involve numerical values. The value of *N* ranges from 1 to 5.
stavaluesN	anyarray	Column data values of the appropriate type for slot *N*. The value is **NULL** if the slot type does not store any data values. Each array's element values are actually of the specific column's data type, so there is no way to define these columns' type more specifically than anyarray. The value of *N* ranges from 1 to 5.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
stadndistinct	real	Number of unique non-null data values in the **DN1** column. • A value greater than 0 is the actual number of distinct values. • A value less than 0 is the ratio of the **distinct** value to the total number of rows. For example, if **stadndistinct** is **-0.5**, the actual **distinct** value is the total number of rows multiplied by 0.5. • The value **0** indicates that the number of distinct values is unknown.
staextinfo	text	Information about extension statistics. Reserved column.
stastate	"char"	Specifies whether the statistics are locked. If the statistics are locked, they cannot be updated. Currently, this parameter is meaningless in distributed mode. • **l**: locked • **u**: unlocked

NOTICE

PG_STATISTIC stores sensitive information about statistical objects, such as MCVs. System administrators and authorized users can access the PG_STATISTIC system catalog to query the sensitive information about the statistical objects.

7.2.18.67 PG_STATISTIC_EXT

PG_STATISTIC_EXT displays extended statistics of tables in a database, such as statistics of multiple columns. Statistics of expressions will be supported later. You can specify the extended statistics to be collected. This system catalog is accessible only to system administrators.

Table 7-139 PG_STATISTIC_EXT columns

Name	Type	Description
starelid	oid	Table or index that the described column belongs to.
starelkind	char	Type of an object. **'c'** indicates a table, and **'p'** indicates a partition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
stainherit	boolean	Specifies whether to collect statistics for objects that have inheritance relationship. ● **t** (true): yes ● **f** (false): no
stanullfrac	real	Percentage of column entries that are null.
stawidth	integer	Average stored width, in bytes, of non-null entries.
stadistinct	real	Number of distinct, non-null data values in the column for all DNs. ● A value greater than 0 is the actual number of distinct values. ● A value less than 0 is the ratio of the **distinct** value to the total number of rows. For example, if **stadistinct** is **-0.5**, the actual **distinct** value is the total number of rows multiplied by 0.5. ● The value **0** indicates that the number of distinct values is unknown.
stadndistinct	real	Number of unique non-null data values in the **DN1** column. ● A value greater than 0 is the actual number of distinct values. ● A value less than 0 is the ratio of the **distinct** value to the total number of rows. For example, if **stadndistinct** is **-0.5**, the actual **distinct** value is the total number of rows multiplied by 0.5. ● The value **0** indicates that the number of distinct values is unknown.
stakindN	smallint	**stakindN** records the type of statistics. N is a code, indicating that the statistics of this type are stored in the *N*th slot of the **pg_statistic** row. Value range of *N* ranges from 1 to 5, for example, stakind1, stakind2, or stakind3. For **stakindN**, the numbers indicate the following information: ● **1**: MCV ● **6**: MCV with a null value ● **8**: Bayesian network
staopN	oid	Operator used to generate the statistics stored in slot *N*. For example, a histogram slot shows the < operator that defines the sort order of the data. The value of *N* ranges from 1 to 5.
stakey	int2vector	Array of a column ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
stanumbersN	real[]	Numerical statistics of the appropriate type for slot *N*. The value is **NULL** if the slot does not involve numerical values. The value of *N* ranges from 1 to 5.
stavaluesN	anyarray	Column data values of the appropriate type for slot *N*. The value is **NULL** if the slot type does not store any data values. Each array's element values are actually of the specific column's data type, so the best practice is to define these columns type as anyarray. The value of *N* ranges from 1 to 5.
staexprs	pg_node_tree	Expression corresponding to the extended statistics information.
stasource	char	Source of extended statistics: ● **'a'**: indicates automatic creation. For details, see the GUC parameter **auto_statistic_ext_columns**. ● **'m'**: indicates that a user manually creates the statistics data using analyze tablename ((column list)) or alter table tablename add statistics ((column list)).
stastatus	char	Status of extended statistics: ● **'a'**: active and available. ● **'d'**: disabled. Related information is not collected, and the optimizer does not use the data when generating a plan. You can use the alter table tablename disable/enable statistics((column list)) syntax to modify the status of extended statistics.
stastate	char	Specifies whether the statistics are locked. If the statistics are locked, they cannot be updated. Currently, this parameter is meaningless in distributed mode. ● **l**: locked ● **u**: unlocked
staextname	name	Alias of the multi-column group of multi-column statistics.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

PG_STATISTIC_EXT stores sensitive information about statistical objects, such as MCVs. System administrators and authorized users can access the PG_STATISTIC_EXT system catalog to query the sensitive information about the statistical objects.

7.2.18.68 PG_SYNONYM

PG_SYNONYM records the mapping between synonym object names and other database object names.

Table 7-140 PG_SYNONYM columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
synname	name	Synonym name.
synnamespace	oid	OID of the namespace that contains a synonym. The OID of the PUBLIC synonym namespace is 0.
synowner	oid	Owner of a synonym, usually the OID of the user who created it. The owner of the PUBLIC synonym is 0.
synobjschema	name	Schema name specified by an associated object.
synobjname	name	Name of an associated object.
syndblinkname	name	Database link name.

7.2.18.69 PG_TABLESPACE

PG_TABLESPACE records tablespace information.

Table 7-141 PG_TABLESPACE columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
spcname	name	Tablespace name.
spcowner	oid	Owner of the tablespace, usually the user who created it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
spcacl	aclitem[]	Access permissions. For details, see **GRANT** and **REVOKE**.
spcoptions	text[]	Options of the tablespace.
spcmaxsize	text	Maximum size of the available disk space, in bytes.
relative	Boolean	Specifies whether the storage path specified by the tablespace is a relative path. • **t (true)**: yes. • **f (false)**: no.

7.2.18.70 PG_TRIGGER

PG_TRIGGER records trigger information.

Table 7-142 PG_TRIGGER columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
tgrelid	oid	OID of the table where the trigger is located.
tgname	name	Trigger name.
tgfoid	oid	Function to be called by the trigger.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
tgtype	smallint	Trigger type. • **4**: AFTER INSERT FOR EACH STATEMENT. • **5**: AFTER INSERT FOR EACH ROW. • **6**: BEFORE INSERT FOR EACH STATEMENT. • **7**: BEFORE INSERT FOR EACH ROW. • **8**: AFTER DELETE FOR EACH STATEMENT. • **9**: AFTER DELETE FOR EACH ROW. • **10**: BEFORE DELETE FOR EACH STATEMENT. • **11**: BEFORE DELETE FOR EACH ROW. • **16**: AFTER UPDATE FOR EACH STATEMENT. • **17**: AFTER UPDATE FOR EACH ROW. • **18**: BEFORE UPDATE FOR EACH STATEMENT. • **19**: BEFORE UPDATE FOR EACH ROW. • **32**: AFTER TRUNCATE FOR EACH STATEMENT. • **34**: BEFORE TRUNCATE FOR EACH STATEMENT. • **69**: INSTEAD OF INSERT FOR EACH ROW. • **73**: INSTEAD OF DELETE FOR EACH ROW. • **81**: INSTEAD OF UPDATE FOR EACH ROW.
tgenabled	"char"	• **O**: The trigger is triggered in origin and local modes. • **D**: The trigger is disabled. • **R**: The trigger is triggered in replica mode. • **A**: The trigger is always triggered.
tgisinternal	Boolean	Internal trigger ID. If the value is **true**, it indicates an internal trigger.
tgconstrrelid	oid	Table referenced by the integrity constraint.
tgconstrindid	oid	Index of the integrity constraint.
tgconstraint	oid	OID of the constraint trigger in **PG_CONSTRAINT**.
tgdeferrable	Boolean	Specifies whether the constraint trigger is of the DEFERRABLE type. • **t** (true): yes • **f** (false): no

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
tginitdeferred	Boolean	Specifies whether the trigger is of the INITIALLY DEFERRED type. • **t** (true): yes • **f** (false): no
tgnargs	smallint	Number of input parameters of the trigger function.
tgattr	int2vector	Column ID specified by the trigger. If no column is specified, an empty array is used.
tgargs	bytea	Parameter specified in the trigger.
tgqual	pg_node_tree	WHEN condition of the trigger (**null** if the WHEN condition does not exist).
tgowner	oid	Trigger owner.

7.2.18.71 PG_TS_CONFIG

PG_TS_CONFIG contains entries representing text search configurations. Each of these configurations contains a list of specific text search parsers and dictionary mappings.

The parser is shown in the PG_TS_CONFIG entry, but the token-to-dictionary mapping is defined by subsidiary entries in **PG_TS_CONFIG_MAP**.

Table 7-143 PG_TS_CONFIG columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
cfgname	name	-	Text search configuration name.
cfgnamespace	oid	**oid** in **PG_NAMESPACE**	OID of the namespace that contains the configuration.
cfgowner	oid	**oid** in **PG_AUTHID**	Owner of the configuration.
cfgparser	oid	**oid** in **PG_TS_PARSER**	OID of the text search parser for this configuration.
cfoptions	text[]	-	Configuration options.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.2.18.72 PG_TS_CONFIG_MAP

PG_TS_CONFIG_MAP contains parser symbol mappings for each text search configuration.

Table 7-144 PG_TS_CONFIG_MAP columns

Name	Type	Reference	Description
mapcfg	oid	**oid** in PG_TS_CONFIG	OID of the PG_TS_CONFIG entry owning this map entry.
maptokentype	integer	-	Token type generated by the configuration's parser.
mapseqno	integer	-	Sequence number of a token type when the values of **mapcfg** or **maptokentype** are the same.
mapdict	oid	**oid** in PG_TS_DICT	OID of the text search dictionary to consult.

7.2.18.73 PG_TS_DICT

PG_TS_DICT contains entries that define text search dictionaries. A dictionary depends on a text search template, which specifies all the implementation functions needed; the dictionary itself provides values for the user-settable parameters supported by the template.

This division of labor allows dictionaries to be created by unprivileged users. The parameters are specified by a text string **dictinitoption**, whose format and meaning vary depending on the template.

Table 7-145 PG_TS_DICT columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
dictname	name	-	Text search dictionary name
dictnamespace	oid	**oid** in PG_NAMESPACE	OID of the namespace that contains the dictionary.
dictowner	oid	**oid** in PG_AUTHID	Owner of the dictionary.
dicttemplate	oid	**oid** in PG_TS_TEMPLATE	OID of the text search template for the dictionary.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
dictinitoption	text	-	Initialization option string for the template.

7.2.18.74 PG_TS_PARSER

PG_TS_PARSER contains definition of text search parsers. A parser is responsible for splitting input text into lexemes and assigning a token type to each lexeme. Because the parser must be implemented through functions at the C language level, the new parser must be created by a database system administrator.

Table 7-146 PG_TS_PARSER columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
prsname	name	-	Text search parser name.
prsnamespace	oid	**oid** in **PG_NAMESPACE**	OID of the namespace that contains the parser.
prsstart	regproc	**proname** in **PG_PROC**	Name of the parser's startup function.
prstoken	regproc	**proname** in **PG_PROC**	Name of the parser's next-token function.
prsend	regproc	**proname** in **PG_PROC**	Name of the parser's shutdown function.
prsheadline	regproc	**proname** in **PG_PROC**	Name of the parser's headline function.
prslextype	regproc	**proname** in **PG_PROC**	Name of the parser's lextype function.

7.2.18.75 PG_TS_TEMPLATE

PG_TS_TEMPLATE contains entries defining text search templates. A template provides a framework for text search dictionaries. Since a template must be implemented by C-language-level functions, templates can be created only by database administrators.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-147 PG_TS_TEMPLATE columns

Name	Type	Reference	Description
oid	oid	-	Row identifier (hidden attribute, which must be specified).
tmplname	name	-	Text search template name
tmplnamespace	oid	**oid** in **PG_NAMESPACE**	OID of the namespace that contains the template
tmplinit	regproc	**proname** in **PG_PROC**	Name of the template's initialization function.
tmpllexize	regproc	**proname** in **PG_PROC**	Name of the template's lexize function.

7.2.18.76 PG_TYPE

PG_TYPE stores information about data types.

Table 7-148 PG_TYPE columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
typname	name	Data type name.
typnamespace	oid	OID of the namespace that contains the type.
typowner	oid	Owner of the type.
typlen	smallint	Number of bytes in the internal representation of the type for a fixed-size type. It is a negative number for a variable-length type. • **-1**: a "varlena" type (one that has a length word). • **-2**: a null-terminated C string.
typbyval	Boolean	• **true**: A value of this type is specified by value internally. • **false**: A value of this type is specified by reference internally. If **typlen** of this type is not **1**, **2**, **4**, or **8**, you are advised to specify a reference value for **typbyval**. You can also specify a value for **typbyval**. Variable-length types are usually passed by reference or by value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
typtype	"char"	• **b**: basic type. • **c**: composite type (for example, the row type of a table). • **d**: domain type. • **p**: pseudo type. • **r**: range type. • **e**: enumeration type. • **u**: undefined type. • **o**: set type. For details, see **typrelid** and **typbasetype**.
typcategory	"char"	Fuzzy classification of data types, which can be used by parsers as the basis for data conversion.
typispreferred	Boolean	• **true**: Data is converted when it complies with the conversion rule specified by **typcategory**. • **false**: Data is not converted.
typisdefined	Boolean	• **true**: The type has been defined. • **false**: The placeholder of an undefined type is used. In this case, there is no reliable information except the name, namespace, and OID of the type.
typdelim	"char"	Character that separates two values of this type when parsing an array input. Note that the delimiter is associated with the array element data type, not the array data type.
typrelid	oid	If this is a composite type (see **typtype**), then this column points to the **PG_CLASS** entry that defines the corresponding table. For a free-standing composite type, the PG_CLASS entry does not represent a table, but it is required for the type's **PG_ATTRIBUTE** entries to link to. It is **0** for non-composite type.
typelem	oid	If **typelem** is not **0**, it identifies another row in PG_TYPE. The current type can be described as an array yielding values of type **typelem**. A "true" array type has a variable length (**typlen** = **-1**), but some fixed-length types (**typlen** > **0**) also have non-zero **typelem**, for example **name** and **point**. If a fixed-length type has a **typelem**, its internal representation must be a number of values of the **typelem** data type with no other data. Variable-length array types have a header defined by the array subroutines.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
typarray	oid	If the value is not **0**, the corresponding type record is available in PG_TYPE.
typinput	regproc	Input conversion function (text format).
typoutput	regproc	Output conversion function (text format).
typreceive	regproc	Input conversion function (binary format); **0** for non-input conversion function.
typsend	regproc	output conversion function (binary format); **0** for non-output conversion function.
typmodin	regproc	Input type modifier function; **0** if none.
typmodout	regproc	Output type modifier function; **0** if none.
typanalyze	regproc	Custom ANALYZE function; **0** if the standard function is used.
typalign	"char"	Alignment required when storing a value of this type. It applies to storage on disks as well as most representations of the value. When multiple values are stored consecutively, such as in the representation of a complete row on disk, padding is inserted before a data of this type so that it begins on the specified boundary. The alignment reference is the beginning of the first datum in the sequence. Possible values are: ● **c**: char alignment, that is, no alignment required ● **s**: short alignment (2 bytes on most machines) ● **i**: integer alignment (4 bytes on most machines) ● **d**: double alignment (8 bytes on many machines, but by no means all) **NOTICE** For types used in system catalogs, the size and alignment defined in PG_TYPE must agree with the way that the compiler lays out the column in a structure representing a table row.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
typstorage	"char"	Data storage mode of a specified column, including the following types: • **p**: plain storage. Values are always stored in plain mode. • **e**: external storage. Data is stored outside the main table (for example, in the TOAST table). The main table stores references to data form external storage. • **m**: main storage. Data can be stored compressed inline, which is similar to plain storage. This method is usually used to store types that require customized storage, for example, customized composite types. • **x**: extended storage. Values may be compressed and/or moved into a level-2 relational table. **x** is usually the storage mode of the TOAST type.
typnotnull	Boolean	Specifies whether the type has a NOT NULL constraint. Currently, it is used for domains only.
typbasetype	oid	If this is a domain (see **typtype**), then **typbasetype** identifies the type that this one is based on. The value is **0** if this type is not a derived type.
typtypmod	integer	Records the **typtypmod** to be applied to domains' base types by domains (the value is **–1** if the base type does not use **typmod**). This is **-1** if this type is not a domain. If the type is array, **typtypmod** indicates the maximum capacity of the array type. If the type is index-by table collection, **typtypmod** indicates the maximum index length of the collection type.
typndims	integer	If a domain is an array, **typndims** is the number of array dimensions. This is **0** for types other than domains over array types.
typcollation	oid	Sorting rule of a specified type. For details about the values, see the **PG_COLLATION** system catalog. (**0** if sequencing is not supported)
typdefaultbin	pg_node_tree	**nodeToString()** representation of a default expression for the type if the value is non-null. Currently, this column is only used for domains.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
typdefault	text	The value is **NULL** if a type has no associated default value. If **typdefaultbin** is not set to **NULL**, **typdefault** must contain a default expression represented by **typdefaultbin**. If **typdefaultbin** is **NULL** and **typdefault** is not, then **typdefault** is the external representation of the type's default value, which can be fed to the type's input converter to produce a constant.
typacl	aclitem[]	Access permission.
typelemmod	integer	• **-1**: **typmod** is not required for the element type corresponding to the set and array types. • Value greater than or equal to 0: **typmod** of the element type corresponding to the set and array types is used. • **NULL**: **typmod** of the element type corresponding to the set and array types is unknown. **typmod** usually refers to the maximum length of a type.

7.2.18.77 PGXC_CLASS

PGXC_CLASS records replicated or distributed information for each table.

Table 7-149 PGXC_CLASS columns

Name	Type	Description
pcrelid	oid	OID of the table.
pclocatortype	"char"	Locator type. • **H**: Hash • **G**: Range • **L**: List • **M**: Modulo • **N**: Round Robin • **R**: Replication • **U**: MurmurHash
pchashalgorithm	smallint	Distributed tuple using the hash algorithm. • **1**: default hash algorithm. • **2**: MURMURHASH algorithm.
pchashbuckets	smallint	Value of a harsh container.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
pgroup	name	Name of the node.
redistributed	"char"	Indicates that a table has been redistributed.
redis_order	integer	Redistribution sequence. Tables whose values are **0** will not be redistributed in this round of redistribution.
pcattnum	int2vector	Column number used as a distributed key.
nodeoids	oidvector_extend	List of distributed table node OIDs.
options	text	Extension status information. This is a reserved column in the system.
diskey	text[]	Character string of the distribution key expression in the brackets of the DISTRIBUTE BY clause in the SQL table creation statement. Currently, only MURMURHASH distribution is supported.
diskeyexprs	pg_node_tree	Character string information obtained by serializing diskey by using the pack_node_to_string() function.

7.2.18.78 PGXC_GROUP

PGXC_GROUP records information about node groups.

Table 7-150 PGXC_GROUP columns

Name	Type	Description
oid	oid	Row identifier (hidden attribute, which must be specified).
group_name	name	Name of a node group
in_redistribution	"char"	Specifies whether redistribution is required. Valid value: • **n**: The node group is not redistributed. • **y**: The source node group is in redistribution. • **t**: The destination node group is in redistribution.
group_members	oidvector_extend	Node OID list of the node group.
group_buckets	text	Distributed data bucket group.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
is_installation	Boolean	Specifies whether to install a sub-cluster. • **t** (true): yes • **f** (false): no
group_acl	aclitem[]	Access permission.
group_kind	"char"	Node group type. The options are as follows: • **i**: installation node group. • **n**: node group in a common cluster. • **e**: elastic cluster. • **s**: system built-in single-DN cluster.
group_parent	oid	For a child node group, this column indicates the OID of the parent node group. For a parent node group, this column is left blank.
bucket_map	text	Mapping between physical buckets and logical buckets.

7.2.18.79 PGXC_SLICE

PGXC_SLICE is a system catalog created for recording range distribution and list distribution details. Currently, range interval cannot be used to automatically scale out shards. It is reserved in the system catalog.

Table 7-151 PGXC_SLICE columns

Name	Type	Description
relname	name	Table name or shard name, which is distinguished by **type**.
type	"char"	• **'t'**: **relname** is the table name. • **'s'**: **relname** is the shard name.
strategy	"char"	• **'r'**: range distributed table. • **'l'**: list distributed table. This value will be extended for subsequent interval shards.
relid	oid	OID of the distributed table to which the tuple belongs.
referenceoid	oid	OID of the referenced distributed table, which is used for slice reference table creation syntax.
sindex	integer	Position of the current boundary in a shard when the table is a list distributed table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
interval	text[]	Reserved column.
transitboundary	text[]	Reserved column.
transitno	integer	Reserved column.
nodeoid	oid	When **relname** is set to a shard name, **nodeoid** indicates the OID of the DN where the shard data is stored.
boundaries	text[]	When **relname** is set to a shard name, this parameter indicates the boundary value of the shard.
specified	boolean	Specifies whether the DN corresponding to the current shard is explicitly specified in the DDL.
sliceorder	integer	User-defined shard sequence.

7.2.18.80 PLAN_TABLE_DATA

PLAN_TABLE_DATA stores plan information collected by EXPLAIN PLAN. Different from the PLAN_TABLE view, the system catalog PLAN_TABLE_DATA stores EXPLAIN PLAN information collected by all sessions and users.

Table 7-152 PLAN_TABLE_DATA columns

Name	Type	Description
session_id	text	Session that inserts the data. Its value consists of a worker thread start timestamp and a worker thread ID. Values are constrained by NOT NULL.
user_id	oid	User who inserts the data. Values are constrained by **NOT NULL**.
statement_id	character varying(30)	Query tag specified by a user.
plan_id	bigint	Query plan ID. The ID is automatically generated in the plan generation phase and is used by kernel engineers for debugging.
id	integer	Node ID in a plan.
operation	character varying(30)	Operation description.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
options	character varying(255)	Operation action.
object_name	name	Name of an operated object. It is defined by users.
object_type	character varying(30)	Object type.
object_owner	name	Schema to which the object belongs. It is defined by users.
projection	character varying(4000)	Returned column information.
cost	double precision	Execution cost estimated by the optimizer for an operator.
cardinality	double precision	Number of rows estimated by the optimizer for an operator.

NOTE

- PLAN_TABLE_DATA records data of all users and sessions on the current node. Only administrators can access all the data. Common users can view their own data in the **PLAN_TABLE** view.
- Data of inactive (exited) sessions is cleaned from PLAN_TABLE_DATA by gs_clean after being stored in this system catalog for a certain period of time (5 minutes by default). You can also run **gs_clean -C** to manually delete data of inactive sessions.
- Data is automatically inserted into PLAN_TABLE_DATA after **EXPLAIN PLAN** is executed. Therefore, do not manually insert data into or update data in PLAN_TABLE_DATA. Otherwise, data in PLAN_TABLE_DATA may be disordered. To delete data from the table, you are advised to use the **PLAN_TABLE** view.
- Information in the **statement_id**, **object_name**, **object_owner**, and **projection** columns is stored in letter cases specified by users and information in other columns is stored in uppercase.

7.2.18.81 STATEMENT_HISTORY

STATEMENT_HISTORY displays information about statements executed on the current node. To query this system catalog, you must have the SYSADMIN permission. The result can be queried only in the system database but cannot be queried in the user database.

The constraints on the query of this system catalog are as follows:

- Data must be queried in the Postgres database. No data exists in other databases.
- This system catalog is controlled by **track_stmt_stat_level**. The default value is **OFF,L0**, where the first part controls full SQL statements, and the second part controls slow SQL statements. For details about the record level of each

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

column, see the following table. To ensure system performance, you are advised to use the SET statement to change the value of this parameter so that the parameter takes effect only for the current session.

- If the values in **track_stmt_stat_level** for controlling full SQL statements and slow SQL statements are not **OFF**, the **is_slow_sql** field is recorded as **true** for an SQL statement that exceeds the **log_min_duration_statement** threshold. Other fields are recorded based on the levels in **track_stmt_stat_level**.

Table 7-153 STATEMENT_HISTORY columns

Name	Type	Description	Lowest Record Level
db_name	name	Database name.	L0
schema_name	name	Schema name.	L0
origin_node	integer	Node name.	L0
user_name	name	Username.	L0
application_name	text	Name of the application that sends a request.	L0
client_addr	text	IP address of the client that sends a request.	L0
client_port	integer	Port number of the client that sends a request.	L0
unique_query_id	bigint	ID of the normalized SQL statement.	L0
debug_query_id	bigint	ID of the unique SQL statement. Some statements are not unique. For example, the value of **debug_query_id** in the Parse packet, DCL statements, and TCL statements is **0**. Statements in a stored procedure use the same ID.	L0
query	text	Normalized SQL (available only on CNs). When **track_stmt_parameter** is enabled, complete SQL statements are displayed.	L0
start_time	timestamp with time zone	Time when a statement starts.	L0
finish_time	timestamp with time zone	Time when a statement ends.	L0

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description	Lowest Record Level
slow_sql_threshold	bigint	Standard for slow SQL statement execution.	L0
transaction_id	bigint	Transaction ID.	L0
thread_id	bigint	ID of an execution thread.	L0
session_id	bigint	Session ID of a user.	L0
n_soft_parse	bigint	Number of soft parsing times. The value of **n_soft_parse** plus the value of **n_hard_parse** may be greater than the value of **n_calls** because the number of subqueries is not counted in the value of **n_calls**. (This is meaningless for non-DML statements.)	L0
n_hard_parse	bigint	Number of hard parsing times. The value of **n_soft_parse** plus the value of **n_hard_parse** may be greater than the value of **n_calls** because the number of subqueries is not counted in the value of **n_calls**. The parsing of complex data types, for example, smalldatetime, also increases the number of parsing times. (This is meaningless for non-DML statements.)	L0
query_plan	text	Statement execution plan.	L1(Full SQL) L0(Slow SQL)
n_returned_rows	bigint	Number of rows in the result set returned by the **SELECT** statement.	L0

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description	Lowest Record Level
n_tuples_fetched	bigint	Number of rows randomly scanned, that is, number of times that TABLE ACCESS BY INDEX ROWID is performed. The process of TABLE ACCESS BY INDEX ROWID is as follows: When you use indexes to query data, if the columns to be queried are not in the indexes, the database engine needs to access the main table again through the primary key or unique identifier in the indexes to obtain the required complete data rows.	L0
n_tuples_returned	bigint	Number of rows sequentially scanned, that is, the sum of the number of live rows obtained by sequential scans and the number of index entries returned by index scans.	L0
n_tuples_inserted	bigint	Number of rows inserted.	L0
n_tuples_updated	bigint	Number of rows updated.	L0
n_tuples_deleted	bigint	Number of rows deleted.	L0
n_blocks_fetched	bigint	Number of buffer block accesses, indicating the total number of blocks read from the buffer and disk.	L0
n_blocks_hit	bigint	Number of buffer block hits, indicating the number of blocks read from the buffer.	L0
db_time	bigint	Valid DB time, which is accumulated if multiple threads are involved (unit: μs).	L0
cpu_time	bigint	Time consumed on the CPU (unit: μs).	L0
execution_time	bigint	Execution time in the executor (unit: μs).	L0
parse_time	bigint	SQL parsing time (unit: μs). (If an exception occurs during parsing, **parse_time** is not recorded.)	L0

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description	Lowest Record Level
plan_time	bigint	SQL plan generation time (unit: μs). (If an exception occurs during plan generation, **plan_time** is not recorded.)	L0
rewrite_time	bigint	SQL rewriting time (unit: μs). (If an exception occurs during rewriting, **rewrite_time** is not recorded.)	L0
pl_execution_time	bigint	Execution time of PL/pgSQL (unit: μs).	L0
pl_compilation_time	bigint	Compilation time of PL/pgSQL (unit: μs).	L0
data_io_time	bigint	I/O time (unit: μs).	L0
net_send_info	text	Network status of messages sent through a physical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, CNs communicate with each other, CNs communicate with clients, and CNs communicate with DNs through physical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: **{"time":***xxx***, "n_calls":***xxx***, "size":***xxx***}**.	L0
net_recv_info	text	Network status of messages received through a physical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, CNs communicate with each other, CNs communicate with clients, and CNs communicate with DNs through physical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: **{"time":***xxx***, "n_calls":***xxx***, "size":***xxx***}**.	L0

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description	Lowest Record Level
net_stream_send_info	text	Network status of messages sent through a logical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, DNs of different shards communicate with each other through logical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: **{"time":***xxx***, "n_calls":***xxx***, "size":***xxx***}**.	L0
net_stream_recv_info	text	Network status of messages received through a logical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, DNs of different shards communicate with each other through logical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: **{"time":***xxx***, "n_calls":***xxx***, "size":***xxx***}**.	L0
lock_count	bigint	Number of locks.	L0
lock_time	bigint	Time required for locking.	L1
lock_wait_count	bigint	Number of lock waits.	L0
lock_wait_time	bigint	Time required for lock waiting.	L1
lock_max_count	bigint	Maximum number of locks.	L0
lwlock_count	bigint	Number of lightweight locks (reserved).	L0
lwlock_wait_count	bigint	Number of lightweight lock waits.	L0
lwlock_time	bigint	Time required for lightweight locking (reserved).	L1
lwlock_wait_time	bigint	Time required for lightweight lock wait.	L1

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description	Lowest Record Level
details	bytea	List of wait times, statement lock events, and GSTrace function running time. When the value of the record level is greater than or equal to L0, the list of wait events starts to be recorded. It displays statistics about wait events on the current node. For details about key events in the kernel, see **Table 7-470**, **Table 7-471**, **Table 7-472**, and **Table 7-473**. You can also view the list of all events in the system in the wait_event_info view. For details about the impact of each transaction lock on services, see **LOCK**. In the GStrace full-link tracing function, the **function** field on the GStrace panel indicates the function name, and the parent-child relationship of the function is expressed in a tree structure. The **duration** field on the GStrace panel indicates the function running time, in microseconds. When the record level is **L2**, the list of statement lock events starts to be recorded in chronological order. The number of recorded lock events is determined by the **track_stmt_details_size** parameter. This column is in binary format and needs to be read using the parsing function pg_catalog.statement_detail_decode. For details, see **Table 1-204**. Events include: • LOCK_START: Start locking. • LOCK_END: Complete locking. • LOCK_WAIT_START: Start lock waiting. • LOCK_WAIT_END: Complete lock waiting. • LOCK_RELEASE: Release the lock. • LWLOCK_WAIT_START: Start lightweight lock waiting.	L0/L2

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description	Lowest Record Level
		• LWLOCK_WAIT_END: Complete lightweight lock waiting.	
is_slow_sql	Boolean	Specifies whether the SQL statement is a slow SQL statement. • **t** (true): yes. • **f** (false): no.	L0
trace_id	text	Driver-specific trace ID, which is associated with an application request.	L0
advise	text	Risk information that may cause the SQL statement to be a slow SQL statement. **NOTE** • If the statement is executed in SQLBypass mode (you can run the **explain** command to view the plan), there is no risk of slow SQL statements and the information in this column may be missing. • This column can contain a maximum of 256 characters and the excess part will be truncated. Optimize the SQL statement based on the existing prompt information. • In **advise**, table aliases are displayed as table aliases, and column aliases are displayed as original column names.	L0
parent_unique_sql_id	bigint	Normalized SQL ID of the outer SQL statement. For statements executed in a stored procedure, the value is the normalized SQL ID of the statement that calls the stored procedure. For statements outside the stored procedure, the value is **0**.	L0

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description	Lowest Record Level
finish_status	text	Statement completion status. • **normal** (default) • **cancelled**: cancellation or termination. • **error**: The operation ends due to other exceptions except **cancelled**. **NOTE** If the statement completion status is not **normal**, the statement execution is abnormal. Other columns have no reference value.	L0
used_memory	integer	Memory usage of a single statement (unit: MB).	L0
lock_max_local_count	bigint	Maximum number of locks that can be held by the locallock mechanism.	L0
lock_max_fastpath_count	bigint	Maximum number of locks that can be held by the fastpath mechanism.	L0
lock_max_global_count	bigint	Maximum number of locks that can be held by the primary lock table.	L0
sql_hash	bigint	Hash value of the current SQL statement.	L0
plan_hash	bigint	Hash value of the current SQL plan.	L0
plan_hash_prev	bigint	For a generic plan, the value is the hash value of the previous plan.	L0
driver_start_time	timestamp with time zone	Time when the JDBC driver starts the execution.	L0
driver_wait_response	timestamp with time zone	Time when the JDBC driver starts to wait for the data returned by the database after sending a packet.	L0
driver_finish_time	timestamp with time zone	Time when the JDBC driver receives the data returned by the database.	L0

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description	Lowest Record Level
driver_info	json	The default value of L0 is **{}**. Starting from L1, the time when the JDBC driver sends and receives packets is recorded. After the **track_stmt_stat_level** parameter is modified, the modification takes effect only for new sessions.	L1
kernel_info	json	Time when database packets are sent and received.	L0
adaptive_join_states	text	Specifies the chosen connection mode of the adaptive connection operator. The format is **[plan_node_id, use_hashjoin]**. The first digit indicates the corresponding **plan_node_id**, and the second digit specifies whether to use HashJoin. The value **1** indicates yes, and the value **0** indicates no.	L0
aplan_count	bigint	Number of Aplan execution times.	L0
aplan_parse_time	bigint	Aplan parsing time.	L0
aplan_execution_time	bigint	Aplan execution time.	L0
relids	text	pg_class.oid involving tables in the statement. Separate multiple tables with commas (,).	L0

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- The record level of the **query_plan** column is L1 for full SQL statements and L0 for slow SQL statements.
- For the **db_time** time model, the time statistics of each dimension in **statement_history** meet the requirement that **db_time** is greater than or equal to **max(cpu_time, parse_time, plan_time, rewrite_time, data_io_time, net_send_info.time, net_recv_info.time, net_stream_send_info.time, net_stream_recv_info.time)**.
- The **query** column in statement_history is obtained from the unique SQL hash table. This process creates the LWLock wait event UniqueSQLMappingLock. By then, **db_time** has already ended, so it does not include this time. Therefore, if **instr_unique_sql_count** is set to a large value, **lwlock_wait_time** may be greater than **db_time**.
- For the **db_time** time model formula, the following dimensions are not included in the statistical range:
  - **execution_time**
  - **pl_execution_time**
  - **pl_compilation_time**
- The maximum nesting depth of wait events is **20**. If the nesting depth exceeds 20, a new wait event overwrites the last wait event.
- Adaptive join does not support distributed mode. Therefore, **adaptive_join_states** does not take effect in distributed mode.
- SQL plan management does not support the distributed mode. Therefore, **sql_hash**, **plan_hash**, and **plan_hash_prev** do not take effect in the distributed mode.
- Statistics about statements in a stored procedure executed on a DN are not aggregated.
- The **CPU_TIME** value of the statement in the stored procedure is **0**.
- Lock-related metrics of sub-statements in a distributed stored procedure are not aggregated to the top-level statements.
- If a query plan fails to be recorded, the cause of the failure is displayed in the **query_plan** column. The following are some examples:
  - In the LightProxy scenario, statements are pushed down, and the plan is recorded on DNs instead of CNs. Message: "The current plan is lightproxy. Please check the query_plan on the DN node."
  - In the Stream scenario, the plan is pushed down and recorded on CNs instead of DNs. Message: "The current plan is streaming. Please check the query_plan on the CN node."
  - For COMMIT statements that are automatically committed by DML statements, related plans are not recorded. Message: "The current record is a transaction commit of a DML statement. Please check the plan of this DML statement."
  - Plans are not recorded for non-DML/DQL statements. Message: "Do not record plans for non-DML/DQL statements." For DDL statements, in a cluster with multiple CNs, query_plan records information about interactions between nodes.
  - Plans are not recorded for substatements that are not adapted or have no plan in a stored procedure, such as simple expressions, OPEN CURSOR, CLOSE, and BYPASS path statements. Plans for substatements executed in a stored procedure are not recorded on DNs. The message "Statement has no plan or is unsupported for plan recording." is displayed.
  - For SQL statements that have entered the PARSE phase, no plan has been generated and therefore the statements are not recorded. Message: "This statement is a PARSE statement and has not yet generated a query plan."
  - When the system checks whether **query_plan** needs to be reported, the SQL statement execution is not slow, but the overall execution of the SQL statements is slow due to subsequent tasks such as cleanup. Message: "Check the reported query_plan time is 2026-01-01 08:05:01.735619+08, at this time the SQL is not slow."

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- If the statement execution status is abnormal, the plan may fail to be recorded. In this case, check the logs to analyze the cause. Message: "An error occurred during statement execution. Please check the log for analysis."
- For other scenarios where the recording fails, check the logs to analyze the cause. Message: "The query plan failed to be recorded. Please check the log analysis."

7.2.18.82 STREAMING_STREAM

STREAMING_STREAM records the metadata of all STREAM objects.

Table 7-154 STREAMING_STREAM column

Name	Type	Description
relid	oid	STREAM object ID.
queries	bytea	Bitmap mapping of the CONTVIEW corresponding to the STREAM.

7.2.18.83 STREAMING_CONT_QUERY

STREAMING_CONT_QUERY records the metadata of all CONTVIEW objects.

Table 7-155 STREAMING_CONT_QUERY columns

Name	Type	Description
id	integer	Unique identifier of the CONTVIEW object.
type	"char"	CONTVIEW type. ● **'r'**: CONTVIEW is based on the row-store model.
relid	oid	CONTVIEW object ID.
defrelid	oid	ID of the continuous computing rule view corresponding to CONTVIEW.
active	Boolean	Specifies whether the CONTVIEW is in the continuous computing state. ● **t** (true): yes. ● **f** (false): no.
streamrelid	oid	ID of STREAM corresponding to CONTVIEW.
matrelid	oid	ID of the materialized table corresponding to CONTVIEW.
lookupidxid	oid	ID of GROUP LOOK UP INDEX corresponding to CONTVIEW. This column is for internal use and is available only in row-store tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
step_factor	smallint	CONTVIEW step mode. The main values are **0** (no overlapping window) and **1** (sliding window, with one step).
ttl	integer	Value of **ttl_interval** set by CONTVIEW.
ttl_attno	smallint	Number of a time column corresponding to the TTL function set by CONTVIEW.
dictrelid	oid	ID of the dictionary table corresponding to CONTVIEW.
grpnum	smallint	Number of dimension columns in the CONTVIEW continuous computing rule. This column is for internal use.
grpidx	int2vector	Index of the dimension column in TARGET LIST in the CONTVIEW continuous computing rule. This column is for internal use.

7.3 System Views

7.3.1 Partitioned Tables

7.3.1.1 ADM_IND_PARTITIONS

ADM_IND_PARTITIONS displays partition information about local indexes of partitioned tables in the database. Each local index partition (if any) of a partitioned table in the database has a corresponding row in ADM_IND_PARTITIONS. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-156 ADM_IND_PARTITIONS columns

Name	Type	Description
index_owner	character varying(64)	Owner name of the partitioned table index to which an index partition belongs.
index_name	character varying(64)	Name of the partitioned table index to which an index partition belongs.
partition_name	character varying(64)	Name of an index partition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
def_tablespace_name	name	Tablespace name of an index partition.
high_value	text	Upper limit of the partition corresponding to the index partition. • For range partitioning, the upper limit of each partition is displayed. • For list partitioning, the value list of each partition is displayed. • For hash partitioning, the number of each partition is displayed.
index_partition_usable	boolean	Specifies whether an index partition is available. • **t** (true): yes. • **f** (false): no.
schema	character varying(64)	Schema of the partitioned table index to which an index partition belongs.
high_value_length	integer	Character length of the limit of the partition corresponding to an index partition.
composite	character varying(3)	Specifies whether the index is a local index on the subpartitioned table. This table does not store subpartition information. Therefore, the value is **NO**.
subpartition_count	numeric	Number of subpartitions in a partition. This table does not store subpartition information. Therefore, the value is **0**.
partition_position	numeric	Position of an index partition in the index.
status	character varying(8)	Specifies whether an index partition is available.
tablespace_name	name	Name of the tablespace that contains a partition.
pct_free	numeric	Minimum percentage of available space in a block.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
ini_trans	numeric	Initial number of transactions. The default value is **4**. The value is **NULL** for non-Ustore partitioned tables.
max_trans	numeric	Maximum number of transactions. The default value is **128**. The value is **NULL** for non-Ustore partitioned tables.
initial_extent	numeric	Not supported. The value is **NULL**.
next_extent	numeric	Not supported. The value is **NULL**.
min_extent	numeric	Not supported. The value is **NULL**.
max_extent	numeric	Not supported. The value is **NULL**.
max_size	numeric	Not supported. The value is **NULL**.
pct_increase	numeric	Not supported. The value is **NULL**.
freelists	numeric	Not supported. The value is **NULL**.
freelist_groups	numeric	Not supported. The value is **NULL**.
logging	character varying(7)	Specifies whether the changes made to an index are logged.
compression	character varying(13)	Specifies whether an index compression is enabled for a partitioned index.
blevel	numeric	Not supported. The value is **NULL**.
leaf_blocks	numeric	Not supported. The value is **NULL**.
distinct_keys	numeric	Not supported. The value is **NULL**.
avg_leaf_blocks_per_key	numeric	Not supported. The value is **NULL**.
avg_data_blocks_per_key	numeric	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
clustering_factor	numeric	Sequence of a row in the table based on the value of the index. You need to run the **analyze** command to collect statistics.
num_rows	numeric	Not supported. The value is **NULL**.
sample_size	numeric	Not supported. The value is **NULL**.
last_analyzed	timestamp with time zone	Date when a partition was last analyzed.
buffer_pool	character varying(7)	Actual buffer pool of a partition.
flash_cache	character varying(7)	Not supported. The value is **NULL**.
cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
pct_direct_access	numeric	Not supported. The value is **NULL**.
global_stats	character varying(3)	Not supported. The value is **NULL**.
domidx_opstatus	character varying(6)	Not supported. The value is **NULL**.
parameters	character varying(1000)	Not supported. The value is **NULL**.
interval	character varying(3)	Specifies whether this partition is an interval partitioned table. The distributed system does not support interval partitioning. Therefore, the value of this parameter is **NO**.
segment_created	character varying(3)	Specifies whether an index partition segment has been created.
orphaned_entries	character varying(3)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.1.2 ADM_IND_SUBPARTITIONS

ADM_IND_SUBPARTITIONS displays partition information about local indexes (excluding global indexes of partitioned tables) of subpartitioned tables in the database. Each local index partition (if any) of a subpartitioned table in the database has a corresponding row in ADM_IND_SUBPARTITIONS. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-157 ADM_IND_SUBPARTITIONS columns

Name	Type	Description
index_owner	character varying(64)	Owner name of the partitioned table index to which an index partition belongs.
index_name	character varying(64)	Name of the partitioned table index to which an index partition belongs.
partition_name	character varying(64)	Name of the partition where an index is located.
subpartition_name	character varying(64)	Name of the subpartition where an index is located.
def_tablespace_name	name	Tablespace name of an index partition.
high_value	text	Limit of the partition corresponding to an index partition. • For range partitioning, the upper limit of each partition is displayed. • For list partitioning, the value list of each partition is displayed. • For hash partitioning, the number of each partition is displayed.
index_partition_usable	boolean	Specifies whether an index partition is available. • **t** (true): yes. • **f** (false): no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
schema	character varying(64)	Schema of the partitioned table index to which an index partition belongs.
high_value_length	integer	Character length of the limit of the partition corresponding to an index partition.
partition_position	numeric	Position of an index partition in the index.
subpartition_position	numeric	Position of the subpartition in a partition.
status	character varying(8)	Specifies whether an index partition is available.
tablespace_name	name	Tablespace name of an index partition.
pct_free	numeric	Minimum percentage of available space in a block.
ini_trans	numeric	Initial number of transactions. The default value is **4**. The value is **NULL** for non-Ustore partitioned tables.
max_trans	numeric	Maximum number of transactions. The default value is **128**. The value is **NULL** for non-Ustore partitioned tables.
initial_extent	numeric	Not supported. The value is **NULL**.
next_extent	numeric	Not supported. The value is **NULL**.
min_extent	numeric	Not supported. The value is **NULL**.
max_extent	numeric	Not supported. The value is **NULL**.
max_size	numeric	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
pct_increase	numeric	Not supported. The value is **NULL**.
freelists	numeric	Not supported. The value is **NULL**.
freelist_groups	numeric	Not supported. The value is **NULL**.
logging	character varying(7)	Specifies whether the changes made to an index are logged.
compression	character varying(13)	Compression type used for subpartitions.
blevel	numeric	Not supported. The value is **NULL**.
leaf_blocks	numeric	Not supported. The value is **NULL**.
distinct_keys	numeric	Not supported. The value is **NULL**.
avg_leaf_blocks_per_key	numeric	Not supported. The value is **NULL**.
avg_data_blocks_per_key	numeric	Not supported. The value is **NULL**.
clustering_factor	numeric	Sequence of a row in the table based on the value of the index. You need to run the **analyze** command to collect statistics.
num_rows	numeric	Number of rows in a subpartition. You need to run the vacuum command to collect statistics.
sample_size	numeric	Not supported. The value is **NULL**.
last_analyzed	timestamp with time zone	Date when a partition was last analyzed.
buffer_pool	character varying(7)	Buffer pool of a subpartition.
flash_cache	character varying(7)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
global_stats	character varying(3)	Not supported. The value is **NULL**.
interval	character varying(3)	Specifies whether this partition is an interval partitioned table. The distributed system does not support interval partitioning. Therefore, the value of this parameter is **NO**.
segment_created	character varying(3)	Specifies whether an index partition segment has been created.
domidx_opstatus	character varying(6)	Not supported. The value is **NULL**.
parameters	character varying(1000)	Not supported. The value is **NULL**.

7.3.1.3 ADM_PART_COL_STATISTICS

ADM_PART_COL_STATISTICS displays column statistics and histogram information about all table partitions in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-158 ADM_PART_COL_STATISTICS columns

Name	Type	Description
owner	character varying(128)	Owner of a partitioned table.
table_name	character varying(128)	Table name.
partition_name	character varying(128)	Table partition name.
column_name	character varying(4000)	Column name.
num_distinct	numeric	Number of different values in a column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
low_value	raw	Low value in a column.
high_value	raw	High value in a column.
density	numeric	• If there is a histogram on the column, this column displays the selectivity of values in the histogram that span less than two endpoints. It does not represent the selectivity of values that span two or more endpoints. • If no histogram is available on the column, the value of this column is **1/ NUM_DISTINCT**.
num_nulls	numeric	Number of empty values in a column.
num_buckets	numeric	Number of buckets in the histogram of a column.
sample_size	numeric	Sample size used to analyze a column.
last_analyzed	date	Date when the column was last analyzed. Database restart is not supported. Otherwise, data loss will occur.
global_stats	character varying(3)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
notes	character varying(63)	Not supported. The value is **NULL**.
avg_col_len	numeric	Average length of a column, in bytes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
histogram	character varying(15)	Specifies whether the histogram exists and the type of the histogram. • **NONE**: no histogram. • **FREQUENCY**: frequency histogram. • **EQUI-WIDTH**: equal-width histogram.
schema	character varying(64)	Name of the namespace to which a column belongs.

NOTE

In ORA-compatible mode, if the GUC parameter **mapping_date_to_datea** is set to **on** during database initialization, the type of the **last_analyzed** column in the view is changed to datea. If the GUC parameter **mapping_date_to_datea** is set to **off**, the type of the **last_analyzed** column in the view is changed to timestamp(0) without time zone.

7.3.1.4 ADM_PART_INDEXES

ADM_PART_INDEXES displays information about all partitioned table indexes in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-159 ADM_PART_INDEXES columns

Name	Type	Description
def_tablespace_name	name	Tablespace name of a partitioned table index.
index_owner	character varying(64)	Owner name of a partitioned table index.
index_name	character varying(64)	Name of a partitioned table index.
partition_count	bigint	Number of index partitions of a partitioned table index.
partitioning_key_count	integer	Number of partition keys of a partitioned table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
partitioning_type	text	Partitioning policy of a partitioned table. **NOTE** For details about the supported partitioning policies of the current partitioned table, see **CREATE TABLE PARTITION**.
schema	character varying(64)	Name of the schema to which a partitioned table index belongs.
table_name	character varying(64)	Name of the partitioned table to which a partitioned table index belongs.
subpartitioning_type	text	Partitioning policy of a subpartitioned table. For a partitioned table, **NONE** is displayed. **NOTE** For details about the supported partitioning policies of the current subpartitioned table, see **CREATE TABLE SUBPARTITION**.
def_subpartition_count	integer	Default number of subpartitions to be created. The value is **1** for a subpartitioned table and **0** for a partitioned table.
subpartitioning_key_count	integer	Number of subpartition keys of a partitioned table.

7.3.1.5 ADM_PART_TABLES

ADM_PART_TABLES displays information about all partitioned tables in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-160 ADM_PART_TABLES columns

Name	Type	Description
table_owner	character varying(64)	Owner name of a partitioned table.
table_name	character varying(64)	Name of a partitioned table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
partitioning_type	text	Partitioning policy of a partitioned table. **NOTE** For details about the supported partitioning policies of the current partitioned table, see **CREATE TABLE PARTITION**.
partition_count	bigint	Number of partitions of a partitioned table.
partitioning_key_count	integer	Number of partition keys of a partitioned table.
def_tablespace_name	name	Tablespace name of a partitioned table.
schema	character varying(64)	Schema of a partitioned table.
subpartitioning_type	text	Partitioning policy of a subpartitioned table. For a partitioned table, **NONE** is displayed.
def_subpartition_count	integer	Default number of subpartitions to be created. The value is **1** for a subpartitioned table and **0** for a partitioned table.
subpartitioning_key_count	integer	Number of subpartition keys of a partitioned table.
status	character varying(8)	Not supported. The value is **valid**.
def_pct_free	numeric	Default value of **PCTFREE** used when a partition is added.
def_pct_used	numeric	Not supported. The value is **NULL**.
def_ini_trans	numeric	Default value of **INITRANS** used when a partition is added.
def_max_trans	numeric	Default value of **MAXTRANS** used when a partition is added.
def_initial_extent	character varying(40)	Not supported. The value is **NULL**.
def_next_extent	character varying(40)	Not supported. The value is **NULL**.
def_min_extents	character varying(40)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
def_max_extents	character varying(40)	Not supported. The value is **NULL**.
def_max_size	character varying(40)	Not supported. The value is **NULL**.
def_pct_increase	character varying(40)	Not supported. The value is **NULL**.
def_freelists	numeric	Not supported. The value is **NULL**.
def_freelist_groups	numeric	Not supported. The value is **NULL**.
def_logging	character varying(7)	Not supported. The value is **NULL**.
def_compression	character varying(8)	Default compression mode used when a partition is added. • NONE • ENABLED • DISABLED
def_compress_for	character varying(30)	Default compression mode used when a partition is added. **NOTE** For available compression modes and compression levels, see **WITH ({ storage_parameter = value } [, ...])**.
def_buffer_pool	character varying(7)	Not supported. The value is **DEFAULT**.
def_flash_cache	character varying(7)	Not supported. The value is **NULL**.
def_cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
ref_ptn_constraint_name	character varying(128)	Not supported. The value is **NULL**.
interval	character varying(1000)	Interval.
autolist	character varying(3)	Not supported. The value is **NO**.
interval_subpartition	character varying(1000)	Not supported. The value is **NULL**.
autolist_subpartition	character varying(3)	Not supported. The value is **NO**.
is_nested	character varying(3)	Not supported. The value is **NO**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
def_segment_creation	character varying(4)	Currently, the segment page mode is not supported. When the segment page mode is enabled, the value is **YES**.
def_indexing	character varying(3)	Not supported. The value is **ON**.
def_inmemory	character varying(8)	Not supported. The value is **NONE**.
def_inmemory_priority	character varying(8)	Not supported. The value is **NULL**.
def_inmemory_distribute	character varying(15)	Not supported. The value is **NULL**.
def_inmemory_compression	character varying(17)	Not supported. The value is **NULL**.
def_inmemory_duplicate	character varying(13)	Not supported. The value is **NULL**.
def_read_only	character varying(3)	Not supported. The value is **NO**.
def_cellmemory	character varying(24)	Not supported. The value is **NULL**.
def_inmemory_service	character varying(12)	Not supported. The value is **NULL**.
def_inmemory_service_name	character varying(1000)	Not supported. The value is **NULL**.

7.3.1.6 ADM_SUBPART_KEY_COLUMNS

ADM_SUBPART_KEY_COLUMNS displays information about the partition key columns of subpartitioned tables or partitioned indexes in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas. The distributed system does not support subpartitioned tables. All columns in this view are set to **NULL**.

Table 7-161 ADM_SUBPART_KEY_COLUMNS columns

Name	Type	Description
owner	character varying(128)	Owner of a subpartitioned table or index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
name	character varying(128)	Name of a subpartitioned table or index.
object_type	character(5)	Object type. • Its value is **table** for a subpartitioned table. • Its value is **index** for a subpartitioned index.
column_name	character varying(4000)	Subpartition key column name of a subpartitioned table.
column_position	numeric	Position of a column in a partition.
collated_column_id	numeric	Not supported. The value is **NULL**.

7.3.1.7 ADM_TAB_PARTITIONS

ADM_TAB_PARTITIONS displays information about partitions (including subpartitioned tables) in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-162 ADM_TAB_PARTITIONS columns

Name	Type	Description
table_owner	character varying(64)	Table owner.
table_name	character varying(64)	Relational table name
partition_name	character varying(64)	Partition name.
high_value	text	Limit of the partition. • For range partitioning, the upper limit of each partition is displayed. • For list partitioning, the value list of each partition is displayed. • For hash partitioning, the number of each partition is displayed.
tablespace_name	name	Tablespace name of a partitioned table.
schema	character varying(64)	Name of a namespace.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
composite	character varying(3)	Specifies whether the table is a subpartitioned table.
subpartition_count	bigint	Number of subpartitions.
high_value_length	integer	Length of the partition boundary value expression.
partition_position	numeric	Position of the partition in the table.
pct_free	numeric	Minimum percentage of available space in a block.
pct_used	numeric	Not supported. The value is **NULL**.
ini_trans	numeric	Initial number of transactions. The default value is **4**. The value is **NULL** for non-Ustore partitioned tables.
max_trans	numeric	Maximum number of transactions. The default value is **128**. The value is **NULL** for non-Ustore partitioned tables.
initial_extent	numeric	Not supported. The value is **NULL**.
next_extent	numeric	Not supported. The value is **NULL**.
min_extent	numeric	Not supported. The value is **NULL**.
max_extent	numeric	Not supported. The value is **NULL**.
max_size	numeric	Not supported. The value is **NULL**.
pct_increase	numeric	Not supported. The value is **NULL**.
freelists	numeric	Not supported. The value is **NULL**.
freelist_groups	numeric	Not supported. The value is **NULL**.
logging	character varying(7)	Specifies whether the changes made to a table are logged.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
compression	character varying(8)	Actual compression attribute of a partitioned table.
compress_for	character varying(30)	Not supported. The value is **NULL**.
num_rows	numeric	Number of rows in a partition.
blocks	numeric	Not supported. The value is **NULL**.
empty_blocks	numeric	Not supported. The value is **NULL**.
avg_space	numeric	Not supported. The value is **NULL**.
chain_cnt	numeric	Not supported. The value is **NULL**.
avg_row_len	numeric	Not supported. The value is **NULL**.
sample_size	numeric	Not supported. The value is **NULL**.
last_analyzed	timestamp with time zone	Date when a partition was last analyzed.
buffer_pool	character varying(7)	Buffer pool allocated to a partitioned block.
flash_cache	character varying(7)	Not supported. The value is **NULL**.
cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
global_stats	character varying(3)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
is_nested	character varying(3)	Specifies whether this partitioned table is a nested partitioned table.
parent_table_partition	character varying(128)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
interval	character varying(3)	Specifies whether this partition is an interval partitioned table. The distributed system does not support interval partitioning. Therefore, the value of this parameter is **NO**.
segment_created	character varying(4)	Specifies whether a table partition has segments created.
indexing	character varying(4)	Not supported. The value is **NULL**.
read_only	character varying(4)	Not supported. The value is **NULL**.
inmemory	character varying(8)	Not supported. The value is **NULL**.
inmemory_priority	character varying(8)	Not supported. The value is **NULL**.
inmemory_distribute	character varying(15)	Not supported. The value is **NULL**.
inmemory_compression	character varying(17)	Not supported. The value is **NULL**.
inmemory_duplicate	character varying(13)	Not supported. The value is **NULL**.
cellmemory	character varying(24)	Not supported. The value is **NULL**.
inmemory_service	character varying(12)	Not supported. The value is **NULL**.
inmemory_service_name	character varying(100)	Not supported. The value is **NULL**.
memoptimize_read	character varying(8)	Not supported. The value is **NULL**.
memoptimize_write	character varying(8)	Not supported. The value is **NULL**.

7.3.1.8 ADM_TAB_SUBPARTITIONS

ADM_TAB_SUBPARTITIONS displays information about all subpartitions in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-163 ADM_TAB_SUBPARTITIONS columns

Name	Type	Description
table_owner	character varying(64)	Table owner.
table_name	character varying(64)	Relational table name
partition_name	character varying(64)	Partition name.
subpartition_name	character varying(64)	Subpartition name.
high_value	text	Limit of a subpartition. • For range partitioning, the upper limit of each partition is displayed. • For list partitioning, the value list of each partition is displayed. • For hash partitioning, the number of each partition is displayed.
tablespace_name	name	Tablespace name of a subpartitioned table.
schema	character varying(64)	Name of a namespace.
high_value_length	integer	Character length of the limit of a subpartition.

7.3.1.9 DB_IND_PARTITIONS

DB_IND_PARTITIONS displays partition information about local indexes (excluding global indexes of partitioned tables) of partitioned tables accessible to the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-164 DB_IND_PARTITIONS columns

Name	Type	Description
index_owner	character varying(64)	Owner name of the partitioned table index to which an index partition belongs.
index_name	character varying(64)	Name of the partitioned table index to which an index partition belongs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
partition_name	character varying(64)	Name of an index partition.
def_tablespace_name	name	Tablespace name of an index partition.
high_value	text	Upper limit of the partition corresponding to the index partition. • For range partitioning, the upper limit of each partition is displayed. • For list partitioning, the value list of each partition is displayed. • For hash partitioning, the number of each partition is displayed.
index_partition_usable	boolean	Specifies whether an index partition is available. • **t** (true): yes. • **f** (false): no.
schema	character varying(64)	Schema of the partitioned table index to which an index partition belongs.
high_value_length	integer	Character length of the limit of the partition corresponding to an index partition.
composite	character varying(3)	Specifies whether the index is a local index on the subpartitioned table. This table does not store subpartition information. Therefore, the value is **NO**.
subpartition_count	numeric	Number of subpartitions in a partition. This table does not store subpartition information. Therefore, the value is **0**.
partition_position	numeric	Position of an index partition in the index.
status	character varying(8)	Specifies whether an index partition is available.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
tablespace_name	name	Name of the tablespace that contains a partition.
pct_free	numeric	Minimum percentage of available space in a block.
ini_trans	numeric	Initial number of transactions. The default value is **4**. The value is **NULL** for non-Ustore partitioned tables.
max_trans	numeric	Maximum number of transactions. The default value is **128**. The value is **NULL** for non-Ustore partitioned tables.
initial_extent	numeric	Not supported. The value is **NULL**.
next_extent	numeric	Not supported. The value is **NULL**.
min_extent	numeric	Not supported. The value is **NULL**.
max_extent	numeric	Not supported. The value is **NULL**.
max_size	numeric	Not supported. The value is **NULL**.
pct_increase	numeric	Not supported. The value is **NULL**.
freelists	numeric	Not supported. The value is **NULL**.
freelist_groups	numeric	Not supported. The value is **NULL**.
logging	character varying(7)	Specifies whether the changes made to an index are logged.
compression	character varying(13)	Specifies whether an index compression is enabled for a partitioned index.
blevel	numeric	Not supported. The value is **NULL**.
leaf_blocks	numeric	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
distinct_keys	numeric	Not supported. The value is **NULL**.
avg_leaf_blocks_per_key	numeric	Not supported. The value is **NULL**.
avg_data_blocks_per_key	numeric	Not supported. The value is **NULL**.
clustering_factor	numeric	Sequence of a row in the table based on the value of the index. You need to run the **analyze** command to collect statistics.
num_rows	numeric	Not supported. The value is **NULL**.
sample_size	numeric	Not supported. The value is **NULL**.
last_analyzed	timestamp with time zone	Date when a partition was last analyzed.
buffer_pool	character varying(7)	Actual buffer pool of a partition.
flash_cache	character varying(7)	Not supported. The value is **NULL**.
cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
pct_direct_access	numeric	Not supported. The value is **NULL**.
global_stats	character varying(3)	Not supported. The value is **NULL**.
domidx_opstatus	character varying(6)	Not supported. The value is **NULL**.
parameters	character varying(1000)	Not supported. The value is **NULL**.
interval	character varying(3)	Specifies whether this partition is an interval partitioned table. The distributed system does not support interval partitioning. Therefore, the value of this parameter is **NO**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
segment_created	character varying(3)	Specifies whether an index partition segment has been created.
orphaned_entries	character varying(3)	Not supported. The value is **NULL**.

7.3.1.10 DB_IND_SUBPARTITIONS

DB_IND_SUBPARTITIONS displays partition information about local indexes (excluding global indexes of partitioned tables) of subpartitioned tables accessible to the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-165 DB_IND_SUBPARTITIONS columns

Name	Type	Description
index_owner	character varying(64)	Owner name of the partitioned table index to which an index partition belongs.
index_name	character varying(64)	Name of the partitioned table index to which an index partition belongs.
partition_name	character varying(64)	Name of the partition where an index is located.
subpartition_name	character varying(64)	Name of the subpartition where an index is located.
def_tablespace_name	name	Tablespace name of an index partition.
high_value	text	Limit of the partition corresponding to an index partition.
index_partition_usable	boolean	Specifies whether an index partition is available. • **t** (true): yes. • **f** (false): no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
schema	character varying(64)	Schema of the partitioned table index to which an index partition belongs.
high_value_length	integer	Character length of the limit of the partition corresponding to an index partition.
partition_position	numeric	Position of an index partition in the index.
subpartition_position	numeric	Position of the subpartition in a partition.
status	character varying(8)	Specifies whether an index partition is available.
tablespace_name	name	Tablespace name of an index partition.
pct_free	numeric	Minimum percentage of available space in a block.
ini_trans	numeric	Initial number of transactions. The default value is **4**. The value is **NULL** for non-Ustore partitioned tables.
max_trans	numeric	Maximum number of transactions. The default value is **128**. The value is **NULL** for non-Ustore partitioned tables.
initial_extent	numeric	Not supported. The value is **NULL**.
next_extent	numeric	Not supported. The value is **NULL**.
min_extent	numeric	Not supported. The value is **NULL**.
max_extent	numeric	Not supported. The value is **NULL**.
max_size	numeric	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
pct_increase	numeric	Not supported. The value is **NULL**.
freelists	numeric	Not supported. The value is **NULL**.
freelist_groups	numeric	Not supported. The value is **NULL**.
logging	character varying(7)	Specifies whether the changes made to an index are logged.
compression	character varying(13)	Compression type used for subpartitions.
blevel	numeric	Not supported. The value is **NULL**.
leaf_blocks	numeric	Not supported. The value is **NULL**.
distinct_keys	numeric	Not supported. The value is **NULL**.
avg_leaf_blocks_per_key	numeric	Not supported. The value is **NULL**.
avg_data_blocks_per_key	numeric	Not supported. The value is **NULL**.
clustering_factor	numeric	Sequence of a row in the table based on the value of the index. You need to run the **analyze** command to collect statistics.
num_rows	numeric	Number of rows in a subpartition. You need to run the vacuum command to collect statistics.
sample_size	numeric	Not supported. The value is **NULL**.
last_analyzed	timestamp with time zone	Date when a partition was last analyzed.
buffer_pool	character varying(7)	Buffer pool of a subpartition.
flash_cache	character varying(7)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
global_stats	character varying(3)	Not supported. The value is **NULL**.
interval	character varying(3)	Specifies whether this partition is an interval partitioned table. The distributed system does not support interval partitioning. Therefore, the value of this parameter is **NO**.
segment_created	character varying(3)	Specifies whether an index partition segment has been created.
domidx_opstatus	character varying(6)	Not supported. The value is **NULL**.
parameters	character varying(1000)	Not supported. The value is **NULL**.

7.3.1.11 DB_PART_COL_STATISTICS

DB_PART_COL_STATISTICS displays column statistics and histogram information about table partitions accessible to the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-166 DB_PART_COL_STATISTICS columns

Name	Type	Description
owner	character varying(128)	Owner of a partitioned table.
table_name	character varying(128)	Table name.
partition_name	character varying(128)	Table partition name.
column_name	character varying(4000)	Column name.
num_distinct	numeric	Number of different values in a column.
low_value	raw	Low value in a column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
high_value	raw	High value in a column.
density	numeric	• If there is a histogram on the column, this column displays the selectivity of values in the histogram that span less than two endpoints. It does not represent the selectivity of values that span two or more endpoints. • If no histogram is available on the column, the value of this column is **1/ NUM_DISTINCT**.
num_nulls	numeric	Number of empty values in a column.
num_buckets	numeric	Number of buckets in the histogram of a column.
sample_size	numeric	Sample size used to analyze a column.
last_analyzed	date	Date when a column was last analyzed. Database restart is not supported. Otherwise, data loss will occur.
global_stats	character varying(3)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
notes	character varying(63)	Not supported. The value is **NULL**.
avg_col_len	numeric	Average length of a column, in bytes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
histogram	character varying(15)	Specifies whether the histogram exists and the type of the histogram. • **NONE**: no histogram. • **FREQUENCY**: frequency histogram. • **EQUI-WIDTH**: equal-width histogram.
schema	character varying(64)	Name of the namespace to which a column belongs.

NOTE

In ORA-compatible mode, if the GUC parameter **mapping_date_to_datea** is set to **on** during database initialization, the type of the **last_analyzed** column in the view is changed to datea. If the GUC parameter **mapping_date_to_datea** is set to **off**, the type of the **last_analyzed** column in the view is changed to timestamp(0) without time zone.

7.3.1.12 DB_PART_INDEXES

DB_PART_INDEXES displays information about partitioned table indexes (excluding global indexes of partitioned tables) accessible to the current user. This view exists in both PG_CATALOG and SYS schemas.

Table 7-167 DB_PART_INDEXES columns

Name	Type	Description
def_tablespace_name	name	Tablespace name of a partitioned table index.
index_owner	character varying(64)	Owner name of a partitioned table index.
index_name	character varying(64)	Name of a partitioned table index.
partition_count	bigint	Number of index partitions of a partitioned table index.
partitioning_key_count	integer	Number of partition keys of a partitioned table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
partitioning_type	text	Partitioning policy of a partitioned table. **NOTE** For details about the supported partitioning policies of the current partitioned table, see **CREATE TABLE PARTITION**.
schema	character varying(64)	Name of the schema to which a partitioned table index belongs.
table_name	character varying(64)	Name of the partitioned table to which a partitioned table index belongs.
subpartitioning_type	text	Partitioning policy of a subpartitioned table. For a partitioned table, **NONE** is displayed. **NOTE** For details about the supported partitioning policies of the current subpartitioned table, see **CREATE TABLE SUBPARTITION**.
def_subpartition_count	integer	Default number of subpartitions to be created. The value is **1** for a subpartitioned table and **0** for a partitioned table.
subpartitioning_key_count	integer	Number of subpartition keys of a partitioned table.

7.3.1.13 DB_PART_KEY_COLUMNS

DB_PART_KEY_COLUMNS displays information about the partition key columns of partitioned tables or partitioned indexes accessible to the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-168 DB_PART_KEY_COLUMNS columns

Name	Type	Description
owner	character varying(128)	Owner of a partitioned table or index.
name	character varying(128)	Name of a partitioned table or index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
object_type	character(5)	Object type. • Its value is **table** for a partitioned table. • Its value is **index** for a partitioned index.
column_name	character varying(4000)	Partition key column name of a partitioned table or index.
column_position	numeric	Position of a column in a partition.
collated_column_id	numeric	Not supported. The value is **NULL**.

7.3.1.14 DB_PART_TABLES

DB_PART_TABLES displays information about partitioned tables accessible to the current user. This view exists in both PG_CATALOG and SYS schemas.

Table 7-169 DB_PART_TABLES columns

Name	Type	Description
table_owner	character varying(64)	Owner name of a partitioned table.
table_name	character varying(64)	Name of a partitioned table.
partitioning_type	text	Partitioning policy of a partitioned table. **NOTE** For details about the supported partitioning policies of the current partitioned table, see **CREATE TABLE PARTITION**.
partition_count	bigint	Number of partitions of a partitioned table.
partitioning_key_count	integer	Number of partition keys of a partitioned table.
def_tablespace_name	name	Tablespace name of a partitioned table.
schema	character varying(64)	Schema of a partitioned table.
subpartitioning_type	text	Partitioning policy of a subpartitioned table. For a partitioned table, **NONE** is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
def_subpartition_count	integer	Default number of subpartitions to be created. The value is **1** for a subpartitioned table and **0** for a partitioned table.
subpartitioning_key_count	integer	Number of subpartition keys of a partitioned table.
status	character varying(8)	Not supported. The value is **valid**.
def_pct_free	numeric	Default value of **PCTFREE** used when a partition is added.
def_pct_used	numeric	Not supported. The value is **NULL**.
def_ini_trans	numeric	Default value of **INITRANS** used when a partition is added.
def_max_trans	numeric	Default value of **MAXTRANS** used when a partition is added.
def_initial_extent	character varying(40)	Not supported. The value is **NULL**.
def_next_extent	character varying(40)	Not supported. The value is **NULL**.
def_min_extents	character varying(40)	Not supported. The value is **NULL**.
def_max_extents	character varying(40)	Not supported. The value is **NULL**.
def_max_size	character varying(40)	Not supported. The value is **NULL**.
def_pct_increase	character varying(40)	Not supported. The value is **NULL**.
def_freelists	numeric	Not supported. The value is **NULL**.
def_freelist_groups	numeric	Not supported. The value is **NULL**.
def_logging	character varying(7)	Not supported. The value is **NULL**.
def_compression	character varying(8)	Default compression mode used when a partition is added. • NONE - Not specified • ENABLED • DISABLED

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
def_compress_for	character varying(30)	Default compression mode used when a partition is added. **NOTE** For available compression modes and compression levels, see **WITH ({ storage_parameter = value } [, ...])**.
def_buffer_pool	character varying(7)	Not supported. The value is **DEFAULT**.
def_flash_cache	character varying(7)	Not supported. The value is **NULL**.
def_cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
ref_ptn_constraint_name	character varying(128)	Not supported. The value is **NULL**.
interval	character varying(1000)	Interval.
autolist	character varying(3)	Not supported. The value is **NO**.
interval_subpartition	character varying(1000)	Not supported. The value is **NULL**.
autolist_subpartition	character varying(3)	Not supported. The value is **NO**.
is_nested	character varying(3)	Not supported. The value is **NO**.
def_segment_creation	character varying(4)	Currently, the segment page mode is not supported. When the segment page mode is enabled, the value is **YES**.
def_indexing	character varying(3)	Not supported. The value is **ON**.
def_inmemory	character varying(8)	Not supported. The value is **NONE**.
def_inmemory_priority	character varying(8)	Not supported. The value is **NULL**.
def_inmemory_distribute	character varying(15)	Not supported. The value is **NULL**.
def_inmemory_compression	character varying(17)	Not supported. The value is **NULL**.
def_inmemory_duplicate	character varying(13)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
def_read_only	character varying(3)	Not supported. The value is **NO**.
def_cellmemory	character varying(24)	Not supported. The value is **NULL**.
def_inmemory_service	character varying(12)	Not supported. The value is **NULL**.
def_inmemory_service_name	character varying(1000)	Not supported. The value is **NULL**.

7.3.1.15 DB_SUBPART_KEY_COLUMNS

DB_SUBPART_KEY_COLUMNS displays information about the partition key columns of subpartitioned tables or partitioned indexes accessible to the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas. The distributed system does not support subpartitioned tables. All columns in this view are set to **NULL**.

Table 7-170 DB_SUBPART_KEY_COLUMNS columns

Name	Type	Description
owner	character varying(128)	Owner of a subpartitioned table or index.
name	character varying(128)	Name of a subpartitioned table or index.
object_type	character(5)	Object type. • Its value is **table** for a partitioned table. • Its value is **index** for a partitioned index.
column_name	character varying(4000)	Subpartition key column name of a subpartitioned table.
column_position	numeric	Position of a column in a partition.
collated_column_id	numeric	Not supported. The value is **NULL**.

7.3.1.16 DB_TAB_PARTITIONS

DB_TAB_PARTITIONS displays information about partitions (including subpartitioned tables) accessible to the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-171 DB_TAB_PARTITIONS columns

Name	Type	Description
table_owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.
partition_name	character varying(64)	Partition name.
high_value	text	Limit of the partition. • For range partitioning, the upper limit of each partition is displayed. • For list partitioning, the value list of each partition is displayed. • For hash partitioning, the number of each partition is displayed.
tablespace_name	name	Tablespace name of a partitioned table.
schema	character varying(64)	Name of a namespace.
composite	character varying(3)	Specifies whether the table is a subpartitioned table.
subpartition_count	bigint	Number of subpartitions.
high_value_length	integer	Length of the partition boundary value expression.
partition_position	numeric	Position of the partition in the table.
pct_free	numeric	Minimum percentage of available space in a block.
pct_used	numeric	Not supported. The value is **NULL**.
ini_trans	numeric	Initial number of transactions. The default value is **4**. The value is **NULL** for non-Ustore partitioned tables.
max_trans	numeric	Maximum number of transactions. The default value is **128**. The value is **NULL** for non-Ustore partitioned tables.
initial_extent	numeric	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
next_extent	numeric	Not supported. The value is **NULL**.
min_extent	numeric	Not supported. The value is **NULL**.
max_extent	numeric	Not supported. The value is **NULL**.
max_size	numeric	Not supported. The value is **NULL**.
pct_increase	numeric	Not supported. The value is **NULL**.
freelists	numeric	Not supported. The value is **NULL**.
freelist_groups	numeric	Not supported. The value is **NULL**.
logging	character varying(7)	Specifies whether the changes made to a table are logged.
compression	character varying(8)	Actual compression attribute of a partitioned table.
compress_for	character varying(30)	Not supported. The value is **NULL**.
num_rows	numeric	Number of rows in a partition.
blocks	numeric	Not supported. The value is **NULL**.
empty_blocks	numeric	Not supported. The value is **NULL**.
avg_space	numeric	Not supported. The value is **NULL**.
chain_cnt	numeric	Not supported. The value is **NULL**.
avg_row_len	numeric	Not supported. The value is **NULL**.
sample_size	numeric	Not supported. The value is **NULL**.
last_analyzed	timestamp with time zone	Date when a partition was last analyzed.
buffer_pool	character varying(7)	Buffer pool allocated to a partitioned block.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
flash_cache	character varying(7)	Not supported. The value is **NULL**.
cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
global_stats	character varying(3)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
is_nested	character varying(3)	Specifies whether this partitioned table is a nested partitioned table.
parent_table_partition	character varying(128)	Not supported. The value is **NULL**.
interval	character varying(3)	Specifies whether this partition is an interval partitioned table. The distributed system does not support interval partitioning. Therefore, the value of this parameter is **NO**.
segment_created	character varying(4)	Specifies whether a table partition has segments created.
indexing	character varying(4)	Not supported. The value is **NULL**.
read_only	character varying(4)	Not supported. The value is **NULL**.
inmemory	character varying(8)	Not supported. The value is **NULL**.
inmemory_priority	character varying(8)	Not supported. The value is **NULL**.
inmemory_distribute	character varying(15)	Not supported. The value is **NULL**.
inmemory_compression	character varying(17)	Not supported. The value is **NULL**.
inmemory_duplicate	character varying(13)	Not supported. The value is **NULL**.
cellmemory	character varying(24)	Not supported. The value is **NULL**.
inmemory_service	character varying(12)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
inmemory_service_name	character varying(100)	Not supported. The value is **NULL**.
memoptimize_read	character varying(8)	Not supported. The value is **NULL**.
memoptimize_write	character varying(8)	Not supported. The value is **NULL**.

7.3.1.17 DB_TAB_SUBPARTITIONS

DB_TAB_SUBPARTITIONS displays information about subpartitioned tables accessible to the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-172 DB_TAB_SUBPARTITIONS columns

Name	Type	Description
table_owner	character varying(64)	Table owner.
table_name	character varying(64)	Relational table name
partition_name	character varying(64)	Partition name.
subpartition_name	character varying(64)	Subpartition name.
high_value	text	Limit of a subpartition. • For range partitioning, the upper limit of each partition is displayed. • For list partitioning, the value list of each partition is displayed. • For hash partitioning, the number of each partition is displayed.
tablespace_name	name	Tablespace name of a subpartitioned table.
schema	character varying(64)	Name of a namespace.
high_value_length	integer	Character length of the limit of a subpartition.
partition_position	numeric	Position of the partition in the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
subpartition_position	numeric	Position of the subpartition in a partition.
pct_free	numeric	Minimum percentage of available space in a block.
pct_used	numeric	Not supported. The value is **NULL**.
ini_trans	numeric	Initial number of transactions. The default value is **4**. The value is **NULL** for non-Ustore partitioned tables.
max_trans	numeric	Maximum number of transactions. The default value is **128**. The value is **NULL** for non-Ustore partitioned tables.
initial_extent	numeric	Not supported. The value is **NULL**.
next_extent	numeric	Not supported. The value is **NULL**.
min_extent	numeric	Not supported. The value is **NULL**.
max_extent	numeric	Not supported. The value is **NULL**.
max_size	numeric	Not supported. The value is **NULL**.
pct_increase	numeric	Not supported. The value is **NULL**.
freelists	numeric	Not supported. The value is **NULL**.
freelist_groups	numeric	Not supported. The value is **NULL**.
logging	character varying(3)	Specifies whether the changes made to a table are logged.
compression	character varying(8)	Actual compression attribute of a partitioned table.
compress_for	character varying(30)	Not supported. The value is **NULL**.
num_rows	numeric	Number of rows in a subpartition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
blocks	numeric	Not supported. The value is **NULL**.
empty_blocks	numeric	Not supported. The value is **NULL**.
avg_space	numeric	Not supported. The value is **NULL**.
chain_cnt	numeric	Not supported. The value is **NULL**.
avg_row_len	numeric	Not supported. The value is **NULL**.
sample_size	numeric	Not supported. The value is **NULL**.
last_analyzed	timestamp with time zone	Date when a partition was last analyzed.
buffer_pool	character varying(7)	Buffer pool of a subpartition.
flash_cache	character varying(7)	Not supported. The value is **NULL**.
cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
global_stats	character varying(3)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
interval	character varying(3)	Specifies whether this partition is an interval partitioned table. The distributed system does not support interval partitioning. Therefore, the value of this parameter is **NO**.
segment_created	character varying(3)	Specifies whether a segment is created in a subpartitioned table.
indexing	character varying(3)	Not supported. The value is **NULL**.
read_only	character varying(3)	Not supported. The value is **NULL**.
inmemory	character varying(8)	Not supported. The value is **NULL**.
inmemory_priority	character varying(8)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
inmemory_distribute	character varying(15)	Not supported. The value is **NULL**.
inmemory_compression	character varying(17)	Not supported. The value is **NULL**.
inmemory_duplicate	character varying(13)	Not supported. The value is **NULL**.
inmemory_service	character varying(12)	Not supported. The value is **NULL**.
inmemory_service_name	character varying(1000)	Not supported. The value is **NULL**.
cellmemory	character varying(24)	Not supported. The value is **NULL**.
memoptimize_read	character varying(8)	Not supported. The value is **NULL**.
memoptimize_write	character varying(8)	Not supported. The value is **NULL**.

7.3.1.18 MY_IND_PARTITIONS

MY_IND_PARTITIONS displays partition information about local indexes (excluding global indexes of partitioned tables) of partitioned tables accessible to the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-173 MY_IND_PARTITIONS columns

Name	Type	Description
index_owner	character varying(64)	Owner name of the partitioned table index to which an index partition belongs.
index_name	character varying(64)	Name of the partitioned table index to which an index partition belongs.
partition_name	character varying(64)	Name of an index partition.
def_tablespace_name	name	Tablespace name of an index partition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
high_value	text	Upper limit of the partition corresponding to the index partition. • For range partitioning, the upper limit of each partition is displayed. • For list partitioning, the value list of each partition is displayed. • For hash partitioning, the number of each partition is displayed.
index_partition_usable	boolean	Specifies whether an index partition is available. • **t** (true): yes. • **f** (false): no.
schema	character varying(64)	Schema of the partitioned table index to which an index partition belongs.
high_value_length	integer	Character length of the limit of the partition corresponding to an index partition.
composite	character varying(3)	Specifies whether the index is a local index on the subpartitioned table. This table does not store subpartition information. Therefore, the value is **NO**.
subpartition_count	numeric	Number of subpartitions in a partition. This table does not store subpartition information. Therefore, the value is **0**.
partition_position	numeric	Position of an index partition in the index.
status	character varying(8)	Specifies whether an index partition is available.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
tablespace_name	name	Name of the tablespace that contains a partition.
pct_free	numeric	Minimum percentage of available space in a block.
ini_trans	numeric	Initial number of transactions. The default value is **4**. The value is **NULL** for non-Ustore partitioned tables.
max_trans	numeric	Maximum number of transactions. The default value is **128**. The value is **NULL** for non-Ustore partitioned tables.
initial_extent	numeric	Not supported. The value is **NULL**.
next_extent	numeric	Not supported. The value is **NULL**.
min_extent	numeric	Not supported. The value is **NULL**.
max_extent	numeric	Not supported. The value is **NULL**.
max_size	numeric	Not supported. The value is **NULL**.
pct_increase	numeric	Not supported. The value is **NULL**.
freelists	numeric	Not supported. The value is **NULL**.
freelist_groups	numeric	Not supported. The value is **NULL**.
logging	character varying(7)	Specifies whether the changes made to an index are logged.
compression	character varying(13)	Specifies whether an index compression is enabled for a partitioned index.
blevel	numeric	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
leaf_blocks	numeric	Not supported. The value is **NULL**.
distinct_keys	numeric	Not supported. The value is **NULL**.
avg_leaf_blocks_per_key	numeric	Not supported. The value is **NULL**.
avg_data_blocks_per_key	numeric	Not supported. The value is **NULL**.
clustering_factor	numeric	Sequence of a row in the table based on the value of the index. You need to run the **analyze** command to collect statistics.
num_rows	numeric	Not supported. The value is **NULL**.
sample_size	numeric	Not supported. The value is **NULL**.
last_analyzed	timestamp with time zone	Date when a partition was last analyzed. Database restart is not supported. Otherwise, data loss will occur.
buffer_pool	character varying(7)	Actual buffer pool of a partition.
flash_cache	character varying(7)	Not supported. The value is **NULL**.
cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
pct_direct_access	numeric	Not supported. The value is **NULL**.
global_stats	character varying(3)	Not supported. The value is **NULL**.
domidx_opstatus	character varying(6)	Not supported. The value is **NULL**.
parameters	character varying(1000)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
interval	character varying(3)	Specifies whether this partition is an interval partitioned table. The distributed system does not support interval partitioning. Therefore, the value of this parameter is **NO**.
segment_created	character varying(3)	Specifies whether an index partition segment has been created.
orphaned_entries	character varying(3)	Not supported. The value is **NULL**.

7.3.1.19 MY_IND_SUBPARTITIONS

MY_IND_SUBPARTITIONS displays partition information about local indexes (excluding global indexes of partitioned tables) of subpartitioned tables owned by the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-174 MY_IND_SUBPARTITIONS columns

Name	Type	Description
index_owner	character varying(64)	Owner name of the partitioned table index to which an index partition belongs.
index_name	character varying(64)	Name of the partitioned table index to which an index partition belongs.
partition_name	character varying(64)	Name of the partition where an index is located.
subpartition_name	character varying(64)	Name of the subpartition where an index is located.
def_tablespace_name	name	Tablespace name of an index partition.
high_value	text	Limit of the partition corresponding to an index partition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
index_partition_usable	boolean	Specifies whether an index partition is available. • **t** (true): yes. • **f** (false): no.
schema	character varying(64)	Schema of the partitioned table index to which an index partition belongs.
high_value_length	integer	Character length of the limit of the partition corresponding to an index partition.
partition_position	numeric	Position of an index partition in the index.
subpartition_position	numeric	Position of the subpartition in a partition.
status	character varying(8)	Specifies whether an index partition is available.
tablespace_name	name	Tablespace name of an index partition.
pct_free	numeric	Minimum percentage of available space in a block.
ini_trans	numeric	Initial number of transactions. The default value is **4**. The value is **NULL** for non-Ustore partitioned tables.
max_trans	numeric	Maximum number of transactions. The default value is **128**. The value is **NULL** for non-Ustore partitioned tables.
initial_extent	numeric	Not supported. The value is **NULL**.
next_extent	numeric	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
min_extent	numeric	Not supported. The value is **NULL**.
max_extent	numeric	Not supported. The value is **NULL**.
max_size	numeric	Not supported. The value is **NULL**.
pct_increase	numeric	Not supported. The value is **NULL**.
freelists	numeric	Not supported. The value is **NULL**.
freelist_groups	numeric	Not supported. The value is **NULL**.
logging	character varying(7)	Specifies whether the changes made to an index are logged.
compression	character varying(13)	Compression type used for subpartitions.
blevel	numeric	Not supported. The value is **NULL**.
leaf_blocks	numeric	Not supported. The value is **NULL**.
distinct_keys	numeric	Not supported. The value is **NULL**.
avg_leaf_blocks_per_key	numeric	Not supported. The value is **NULL**.
avg_data_blocks_per_key	numeric	Not supported. The value is **NULL**.
clustering_factor	numeric	Sequence of a row in the table based on the value of the index. You need to run the **analyze** command to collect statistics.
num_rows	numeric	Number of rows in a subpartition. You need to run the vacuum command to collect statistics.
sample_size	numeric	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
last_analyzed	timestamp with time zone	Date when a partition was last analyzed. Database restart is not supported. Otherwise, data loss will occur.
buffer_pool	character varying(7)	Buffer pool of a subpartition.
flash_cache	character varying(7)	Not supported. The value is **NULL**.
cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
global_stats	character varying(3)	Not supported. The value is **NULL**.
interval	character varying(3)	Specifies whether this partition is an interval partitioned table. The distributed system does not support interval partitioning. Therefore, the value of this parameter is **NO**.
segment_created	character varying(3)	Specifies whether an index partition segment has been created.
domidx_opstatus	character varying(6)	Not supported. The value is **NULL**.
parameters	character varying(1000)	Not supported. The value is **NULL**.

7.3.1.20 MY_PART_COL_STATISTICS

MY_PART_COL_STATISTICS displays column statistics and histogram information about table partitions owned by the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-175 MY_PART_COL_STATISTICS columns

Name	Type	Description
table_name	character varying(128)	Table name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
partition_name	character varying(128)	Table partition name.
column_name	character varying(4000)	Column name.
num_distinct	numeric	Number of different values in a column.
low_value	raw	Low value in a column.
high_value	raw	High value in a column.
density	numeric	• If there is a histogram on the column, this column displays the selectivity of values in the histogram that span less than two endpoints. It does not represent the selectivity of values that span two or more endpoints. • If no histogram is available on the column, the value of this column is **1/NUM_DISTINCT**.
num_nulls	numeric	Number of empty values in a column.
num_buckets	numeric	Number of buckets in the histogram of a column.
sample_size	numeric	Sample size used to analyze a column.
last_analyzed	date	Date when a column was last analyzed. Database restart is not supported. Otherwise, data loss will occur.
global_stats	character varying(3)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
notes	character varying(63)	Not supported. The value is **NULL**.
avg_col_len	numeric	Average length of a column, in bytes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
histogram	character varying(15)	Specifies whether the histogram exists and the type of the histogram. • **NONE**: no histogram. • **FREQUENCY**: frequency histogram. • **EQUI-WIDTH**: equal-width histogram.
schema	character varying(64)	Name of the namespace to which a column belongs.

NOTE

In ORA-compatible mode, if the GUC parameter **mapping_date_to_datea** is set to **on** during database initialization, the type of the **last_analyzed** column in the view is changed to datea. If the GUC parameter **mapping_date_to_datea** is set to **off**, the type of the **last_analyzed** column in the view is changed to timestamp(0) without time zone.

7.3.1.21 MY_PART_INDEXES

MY_PART_INDEXES displays information about partitioned table indexes accessible to the current user. This view exists in both PG_CATALOG and SYS schemas.

Table 7-176 MY_PART_INDEXES columns

Name	Type	Description
def_tablespace_name	name	Tablespace name of a partitioned table index.
index_owner	character varying(64)	Owner name of a partitioned table index.
index_name	character varying(64)	Name of a partitioned table index.
partition_count	bigint	Number of index partitions of a partitioned table index.
partitioning_key_count	integer	Number of partition keys of a partitioned table.
partitioning_type	text	Partitioning policy of a partitioned table. **NOTE** For details about the supported partitioning policies of the current partitioned table, see **CREATE TABLE PARTITION**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
schema	character varying(64)	Schema of a partitioned table index.
table_name	character varying(64)	Name of the partitioned table to which a partitioned table index belongs.
subpartitioning_type	text	Partitioning policy of a subpartitioned table. For a partitioned table, **NONE** is displayed. **NOTE** For details about the supported partitioning policies of the current subpartitioned table, see **CREATE TABLE SUBPARTITION**.
def_subpartition_count	integer	Default number of subpartitions to be created. The value is **1** for a subpartitioned table and **0** for a partitioned table.
subpartitioning_key_count	integer	Number of subpartition keys of a partitioned table.

7.3.1.22 MY_PART_KEY_COLUMNS

MY_PART_KEY_COLUMNS displays information about the partition key columns of partitioned tables or partitioned indexes owned by the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-177 MY_PART_KEY_COLUMNS columns

Name	Type	Description
name	character varying(128)	Name of a partitioned table or index.
object_type	character(5)	Object type. • Its value is **table** for a partitioned table. • Its value is **index** for a partitioned index.
column_name	character varying(4000)	Partition key column name of a partitioned table or index.
column_position	numeric	Position of a column in a partition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
collated_column_id	numeric	Not supported. The value is **NULL**.

7.3.1.23 MY_PART_TABLES

MY_PART_TABLES displays information about partitioned tables accessible to the current user. This view exists in both PG_CATALOG and SYS schemas.

Table 7-178 MY_PART_TABLES columns

Name	Type	Description
table_owner	character varying(64)	Owner name of a partitioned table.
table_name	character varying(64)	Name of a partitioned table.
partitioning_type	text	Partitioning policy of a partitioned table. **NOTE** For details about the supported partitioning policies of the current partitioned table, see **CREATE TABLE PARTITION**.
partition_count	bigint	Number of partitions of a partitioned table.
partitioning_key_count	integer	Number of partition keys of a partitioned table.
def_tablespace_name	name	Tablespace name of a partitioned table.
schema	character varying(64)	Schema of a partitioned table.
subpartitioning_type	text	Partitioning policy of a subpartitioned table. For a partitioned table, **NONE** is displayed.
def_subpartition_count	integer	Default number of subpartitions to be created. The value is **1** for a subpartitioned table and **0** for a partitioned table.
subpartitioning_key_count	integer	Number of subpartition keys of a partitioned table.
status	character varying(8)	Not supported. The value is **valid**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
def_pct_free	numeric	Default value of **PCTFREE** used when a partition is added.
def_pct_used	numeric	Not supported. The value is **NULL**.
def_ini_trans	numeric	Default value of **INITRANS** used when a partition is added.
def_max_trans	numeric	Default value of **MAXTRANS** used when a partition is added.
def_initial_extent	character varying(40)	Not supported. The value is **NULL**.
def_next_extent	character varying(40)	Not supported. The value is **NULL**.
def_min_extents	character varying(40)	Not supported. The value is **NULL**.
def_max_extents	character varying(40)	Not supported. The value is **NULL**.
def_max_size	character varying(40)	Not supported. The value is **NULL**.
def_pct_increase	character varying(40)	Not supported. The value is **NULL**.
def_freelists	numeric	Not supported. The value is **NULL**.
def_freelist_groups	numeric	Not supported. The value is **NULL**.
def_logging	character varying(7)	Not supported. The value is **NULL**.
def_compression	character varying(8)	Default compression mode used when a partition is added. • NONE • ENABLED • DISABLED

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
def_compress_for	character varying(30)	Default compression mode used when a partition is added. **NOTE** For available compression modes and compression levels, see **WITH ({ storage_parameter = value } [, ...])**.
def_buffer_pool	character varying(7)	Not supported. The value is **DEFAULT**.
def_flash_cache	character varying(7)	Not supported. The value is **NULL**.
def_cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
ref_ptn_constraint_name	character varying(128)	Not supported. The value is **NULL**.
interval	character varying(1000)	Interval.
autolist	character varying(3)	Not supported. The value is **NO**.
interval_subpartition	character varying(1000)	Not supported. The value is **NULL**.
autolist_subpartition	character varying(3)	Not supported. The value is **NO**.
is_nested	character varying(3)	Not supported. The value is **NO**.
def_segment_creation	character varying(4)	Currently, the segment page mode is not supported. When the segment page mode is enabled, the value is **YES**.
def_indexing	character varying(3)	Not supported. The value is **ON**.
def_inmemory	character varying(8)	Not supported. The value is **NONE**.
def_inmemory_priority	character varying(8)	Not supported. The value is **NULL**.
def_inmemory_distribute	character varying(15)	Not supported. The value is **NULL**.
def_inmemory_compression	character varying(17)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
def_inmemory_duplicate	character varying(13)	Not supported. The value is **NULL**.
def_read_only	character varying(3)	Not supported. The value is **NO**.
def_cellmemory	character varying(24)	Not supported. The value is **NULL**.
def_inmemory_service	character varying(12)	Not supported. The value is **NULL**.
def_inmemory_service_name	character varying(1000)	Not supported. The value is **NULL**.

7.3.1.24 MY_SUBPART_KEY_COLUMNS

MY_SUBPART_KEY_COLUMNS displays information about the partition key columns of subpartitioned tables or partitioned indexes owned by the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas. The distributed system does not support subpartitioned tables. All columns in this view are set to **NULL**.

Table 7-179 MY_SUBPART_KEY_COLUMNS columns

Name	Type	Description
name	character varying(128)	Name of a subpartitioned table or index.
object_type	character(5)	Object type. • Its value is **table** for a partitioned table. • Its value is **index** for a partitioned index.
column_name	character varying(4000)	Subpartition key column name of a subpartitioned table.
column_position	numeric	Position of a column in a partition.
collated_column_id	numeric	Not supported. The value is **NULL**.

7.3.1.25 MY_TAB_PARTITIONS

MY_TAB_PARTITIONS displays information about all partitions (including subpartitioned tables) accessible to the current user. Each partition of a partitioned table accessible to the current user has one record in

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

MY_TAB_PARTITIONS. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-180 MY_TAB_PARTITIONS columns

Name	Type	Description
table_owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.
partition_name	character varying(64)	Partition name.
high_value	text	Limit of the partition. • For range partitioning, the upper limit of each partition is displayed. • For list partitioning, the value list of each partition is displayed. • For hash partitioning, the number of each partition is displayed.
tablespace_name	name	Tablespace name of a partitioned table.
schema	character varying(64)	Name of a namespace.
composite	character varying(3)	Specifies whether the table is a subpartitioned table.
subpartition_count	bigint	Number of subpartitions.
high_value_length	integer	Length of the partition boundary value expression.
partition_position	numeric	Position of the partition in the table.
pct_free	numeric	Minimum percentage of available space in a block.
pct_used	numeric	Not supported. The value is **NULL**.
ini_trans	numeric	Initial number of transactions. The default value is **4**. The value is **NULL** for non-Ustore partitioned tables.
max_trans	numeric	Maximum number of transactions. The default value is **128**. The value is **NULL** for non-Ustore partitioned tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
initial_extent	numeric	Not supported. The value is **NULL**.
next_extent	numeric	Not supported. The value is **NULL**.
min_extent	numeric	Not supported. The value is **NULL**.
max_extent	numeric	Not supported. The value is **NULL**.
max_size	numeric	Not supported. The value is **NULL**.
pct_increase	numeric	Not supported. The value is **NULL**.
freelists	numeric	Not supported. The value is **NULL**.
freelist_groups	numeric	Not supported. The value is **NULL**.
logging	character varying(7)	Specifies whether the changes made to a table are logged.
compression	character varying(8)	Actual compression attribute of a partitioned table.
compress_for	character varying(30)	Not supported. The value is **NULL**.
num_rows	numeric	Number of rows in a partition.
blocks	numeric	Not supported. The value is **NULL**.
empty_blocks	numeric	Not supported. The value is **NULL**.
avg_space	numeric	Not supported. The value is **NULL**.
chain_cnt	numeric	Not supported. The value is **NULL**.
avg_row_len	numeric	Not supported. The value is **NULL**.
sample_size	numeric	Not supported. The value is **NULL**.
last_analyzed	timestamp with time zone	Date when a partition was last analyzed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
buffer_pool	character varying(7)	Buffer pool allocated to a partitioned block.
flash_cache	character varying(7)	Not supported. The value is **NULL**.
cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
global_stats	character varying(3)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
is_nested	character varying(3)	Specifies whether this partitioned table is a nested partitioned table.
parent_table_partition	character varying(128)	Not supported. The value is **NULL**.
interval	character varying(3)	Specifies whether this partition is an interval partitioned table. The distributed system does not support interval partitioning. Therefore, the value of this parameter is **NO**.
segment_created	character varying(4)	Specifies whether a table partition has segments created.
indexing	character varying(4)	Not supported. The value is **NULL**.
read_only	character varying(4)	Not supported. The value is **NULL**.
inmemory	character varying(8)	Not supported. The value is **NULL**.
inmemory_priority	character varying(8)	Not supported. The value is **NULL**.
inmemory_distribute	character varying(15)	Not supported. The value is **NULL**.
inmemory_compression	character varying(17)	Not supported. The value is **NULL**.
inmemory_duplicate	character varying(13)	Not supported. The value is **NULL**.
cellmemory	character varying(24)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
inmemory_service	character varying(12)	Not supported. The value is **NULL**.
inmemory_service_name	character varying(100)	Not supported. The value is **NULL**.
memoptimize_read	character varying(8)	Not supported. The value is **NULL**.
memoptimize_write	character varying(8)	Not supported. The value is **NULL**.

7.3.1.26 MY_TAB_SUBPARTITIONS

MY_TAB_SUBPARTITIONS displays information about all subpartitioned tables accessible to the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-181 MY_TAB_SUBPARTITIONS columns

Name	Type	Description
table_owner	character varying(64)	Table owner.
schema	character varying(64)	Schema name.
table_name	character varying(64)	Table name.
partition_name	character varying(64)	Partition name.
subpartition_name	character varying(64)	Subpartition name.
high_value	text	Boundary value expression in a subpartition.
high_value_length	integer	Length of the boundary value expression in subpartition.
partition_position	numeric	Position of the partition in the table.
subpartition_position	numeric	Position of the subpartition in a partition.
tablespace_name	name	Name of the tablespace that contains a subpartition.
pct_free	numeric	Minimum percentage of available space in a block.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
pct_used	numeric	Not supported. The value is **NULL**.
ini_trans	numeric	Initial number of transactions.
max_trans	numeric	Maximum number of transactions.
initial_extent	numeric	Not supported. The value is **NULL**.
next_extent	numeric	Not supported. The value is **NULL**.
min_extent	numeric	Not supported. The value is **NULL**.
max_extent	numeric	Not supported. The value is **NULL**.
max_size	numeric	Not supported. The value is **NULL**.
pct_increase	numeric	Not supported. The value is **NULL**.
freelists	numeric	Not supported. The value is **NULL**.
freelist_groups	numeric	Not supported. The value is **NULL**.
logging	character varying(7)	Specifies whether the changes made to a table are logged. • YES • NO
compression	character varying(8)	Specifies whether a subpartition is compressed. Value range: **ENABLED** and **DISABLED**.
compress_for	character varying(30)	Not supported. The value is **NULL**.
num_rows	numeric	Not supported. The value is **NULL**.
blocks	numeric	Not supported. The value is **NULL**.
empty_blocks	numeric	Not supported. The value is **NULL**.
avg_space	numeric	Not supported. The value is **NULL**.
chain_cnt	numeric	Not supported. The value is **NULL**.
avg_row_len	numeric	Not supported. The value is **NULL**.
sample_size	numeric	Not supported. The value is **NULL**.
last_analyzed	timestamp with time zone	Date when a table was last analyzed. Database restart is not supported. Otherwise, data loss will occur.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
buffer_pool	character varying(7)	Buffer pool of a subpartition. • DEFAULT • KEEP • RECYCLE • NULL
flash_cache	character varying(7)	Not supported. The value is **NULL**.
cell_flash_cache	character varying(7)	Not supported. The value is **NULL**.
global_stats	character varying(3)	Not supported. The value is **NULL**.
user_stats	character varying(3)	Not supported. The value is **NULL**.
interval	character varying(3)	Specifies whether this partition is an interval partitioned table. The distributed system does not support interval partitioning. Therefore, the value of this parameter is **NO**.
segment_created	character varying(3)	Specifies whether a subpartition segment has been created. • **YES**: created. • **NO**: not created. • **N/A**: The table does not have subpartitions.
indexing	character varying(3)	Not supported. The value is **NULL**.
read_only	character varying(5)	Not supported. The value is **NULL**.
inmemory	character varying(8)	Not supported. The value is **NULL**.
inmemory_priority	character varying(8)	Not supported. The value is **NULL**.
inmemory_distribute	character varying(15)	Not supported. The value is **NULL**.
inmemory_compression	character varying(17)	Not supported. The value is **NULL**.
inmemory_duplicate	character varying(13)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
inmemory_service	character varying(12)	Not supported. The value is **NULL**.
inmemory_service_name	character varying(1000)	Not supported. The value is **NULL**.
cellmemory	character varying(24)	Not supported. The value is **NULL**.
memoptimize_read	character varying(8)	Not supported. The value is **NULL**.
memoptimize_write	character varying(8)	Not supported. The value is **NULL**.

7.3.1.27 GS_STATIO_ALL_PARTITIONS

GS_STATIO_ALL_PARTITIONS contains I/O statistics about each partition in a partitioned table of the current database. The information can be queried by using the gs_statio_get_all_partitions_stats() function.

Table 7-182 GS_STATIO_ALL_PARTITIONS columns

Name	Type	Description
partition_oid	oid	Partition OID.
schemaname	name	Name of a partition schema.
relname	name	Name of the table where a partition is located.
partition_name	name	Name of the partition to which a partition belongs.
sub_partition_name	name	Name of the subpartition to which a partition belongs. The distributed system does not support subpartitions. Set it to **NULL**.
heap_blks_read	bigint	Number of disk blocks read from the partition.
heap_blks_hit	bigint	Number of cache hits in the partition.
idx_blks_read	bigint	Number of disk blocks read from all indexes on the partition.
idx_blks_hit	bigint	Number of cache hits of all indexes in a partition.
toast_blks_read	bigint	Number of disk blocks read from the partition's TOAST table partition (if any).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
toast_blks_hit	bigint	Number of buffer hits in the partition's TOAST table partition (if any).
tidx_blks_read	bigint	Number of disk blocks read from the partition's TOAST table partitioned index (if any).
tidx_blks_hit	bigint	Specifies the number of buffer hits in the partition's TOAST table partitioned index (if any).

7.3.1.28 GS_STAT_XACT_ALL_PARTITIONS

GS_STAT_XACT_ALL_PARTITIONS displays the transaction status about all partitions of partitioned tables in a namespace. The information can be queried by using the gs_stat_get_xact_all_partitions_stats() function.

Table 7-183 GS_STAT_XACT_ALL_PARTITIONS columns

Name	Type	Description
partition_oid	oid	Partition OID.
schemaname	name	Name of a partition schema.
relname	name	Name of the table where a partition is located.
partition_name	name	Name of the partition to which a partition belongs.
sub_partition_name	name	Name of the subpartition to which a partition belongs. The distributed system does not support subpartitions. Set it to **NULL**.
seq_scan	bigint	Number of sequential scans initiated by a partition.
seq_tup_read	bigint	Specifies the number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated by a partition.
idx_tup_fetch	bigint	Specifies the number of live rows fetched by index scans.
n_tup_ins	bigint	Specifies the number of rows inserted.
n_tup_upd	bigint	Specifies the number of rows updated.
n_tup_del	bigint	Specifies the number of rows deleted.
n_tup_hot_upd	bigint	Number of rows HOT updated (with no separate index update required).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.1.29 GS_STAT_ALL_PARTITIONS

GS_STAT_ALL_PARTITIONS displays information about each partition in all partitioned tables in the current database. Each partition occupies a row, showing statistics about access to the partition. The current version does not support the update of subpartition statistics. The view information is queried by the gs_stat_get_all_partitions_stats() function.

Table 7-184 GS_STAT_ALL_PARTITIONS columns

Name	Type	Description
partition_oid	oid	Partition OID.
schemaname	name	Schema name of the table to which a partition belongs.
relname	name	Name of the table where a partition is located.
partition_name	name	Name of the partition to which a partition belongs.
sub_partition_name	name	Name of the subpartition to which a partition belongs.
seq_scan	bigint	Number of sequential scans initiated by a partition.
seq_tup_read	bigint	Specifies the number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated by a partition.
idx_tup_fetch	bigint	Specifies the number of live rows fetched by index scans.
n_tup_ins	bigint	Specifies the number of rows inserted.
n_tup_upd	bigint	Specifies the number of rows updated.
n_tup_del	bigint	Specifies the number of rows deleted.
n_tup_hot_upd	bigint	Number of rows HOT updated (with no separate index update required).
n_live_tup	bigint	Estimated number of live rows.
n_dead_tup	bigint	Estimated number of inactive rows. It indicates the number of dead row pointers in Ustore tables.
last_vacuum	timestamp with time zone	Time when the partition was last cleared.
last_autovacuum	timestamp with time zone	Time when the partition was last cleared by the autovacuum daemon thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
last_analyze	timestamp with time zone	Time when the partition was last analyzed.
last_autoanalyze	timestamp with time zone	Time when the partition was last analyzed by the autovacuum daemon thread.
vacuum_count	bigint	Number of times that a partition is cleared.
autovacuum_count	bigint	Number of times that a partition is cleared by the autovacuum daemon thread.
analyze_count	bigint	Number of times that a partition is analyzed.
autoanalyze_count	bigint	Number of times that a partition is analyzed by the autovacuum daemon thread.

7.3.2 Workload Management

7.3.2.1 GS_WLM_CGROUP_INFO

GS_WLM_CGROUP_INFO displays information about a Cgroup for a job that is being executed.

NOTE

If no data is displayed in the queried view, contact Huawei technical support.

Table 7-185 GS_WLM_CGROUP_INFO columns

Name	Type	Description
cgroup_name	text	Cgroup name.
priority	integer	Priority of the job.
usage_percent	integer	Percentage of resources used by the Cgroup.
shares	bigint	CPU quota allocated to the Cgroup.
cpuacct	bigint	Allocated CPU quota.
cpuset	text	Allocated CPU cores.
relpath	text	Relative path of the Cgroup.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
valid	text	Specifies whether the Cgroup is valid.
node_group	text	Reserved column.

7.3.2.2 GS_WLM_REBUILD_USER_RESOURCE_POOL

Rebuilds a user's resource pool information in memory on the current connection node. It is used as a remedy only when resource pool information is missing or disordered. Only users with the SYSADMIN permission can query this view.

Table 7-186 GS_WLM_REBUILD_USER_RESOURCE_POOL columns

Name	Type	Description
gs_wlm_rebuild_user_resource_pool	Boolean	Rebuilds information about the user resource pool in the memory. **t** indicates success, and **f** indicates failure.

7.3.2.3 GS_WLM_USER_INFO

GS_WLM_USER_INFO displays user statistics. Only users with the SYSADMIN permission can query this view.

NOTE

If no data is displayed in the queried view, contact Huawei technical support.

Table 7-187 GS_WLM_USER_INFO columns

Name	Type	Description
userid	oid	OID of a user.
username	name	Username.
sysadmin	Boolean	Specifies whether the user is an administrator. • **t** (true): yes. • **f** (false): no.
rpoid	oid	OID of the associated resource pool.
respool	name	Name of the associated resource pool.
parentid	oid	OID of the user group.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
totalspace	bigint	Available space limit of the user.
spacelimit	bigint	User table space limit.
childcount	integer	Number of child users.
childlist	text	Child user list.

7.3.2.4 GLOBAL_WLM_RESPOOL_CONCURRENCY_INFO

Views the real-time concurrency usage of resource pools on all nodes in a distributed system. Only the user with the SYSADMIN or MONADMIN permission can query this view.

This view can be used to query only the data on the primary DN node. You need to log in to the standby node and use the system function **gs_wlm_respool_concurrency_info()** to query the usage of concurrent tasks.

Table 7-188 GLOBAL_WLM_RESPOOL_CONCURRENCY_INFO columns

Name	Type	Description
node_name	name	Node name.
respool_name	text	Name of a resource pool.
max_concurrency	integer	Maximum number of concurrent queries allowed by the resource pool.
running_concurrency	integer	Number of concurrent tasks that are being executed in the current resource pool.
waiting_concurrency	integer	Number of concurrent tasks that are waiting in the current resource pool.

Example:

```
gaussdb=# SELECT * FROM GLOBAL_WLM_RESPOOL_CONCURRENCY_INFO;
 node_name   | respool_name | max_concurrency | running_concurrency | waiting_concurrency
-------------+--------------+-----------------+---------------------+---------------------
 coordinator1 | default_pool |              -1 |                   1 |                   0
 datanode3    | default_pool |              -1 |                   1 |                   0
 datanode1    | default_pool |              -1 |                   1 |                   0
 datanode2    | default_pool |              -1 |                   1 |                   0
(4 rows)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.2.5 GLOBAL_WLM_RESPOOL_CONNECTION_INFO

This view is used to view the real-time connection usage of resource pool on all nodes in a distributed system. Only the user with the SYSADMIN or MONADMIN permission can query this view.

This view can be used to query only the data on the primary DN node. You need to log in to the standby node and use the system function **gs_wlm_respool_connection_info()** to query the usage of concurrent tasks.

Table 7-189 GLOBAL_WLM_RESPOOL_CONNECTION_INFO columns

Name	Type	Description
node_name	name	Node name.
respool_name	text	Name of a resource pool.
max_connections	integer	Maximum number of connections to a resource pool.
curr_connections	integer	Number of existing connections in the current resource pool.

Example:

```
gaussdb=# SELECT * FROM GLOBAL_WLM_RESPOOL_CONCURRENCY_INFO;
 node_name   | respool_name | max_connections | curr_connections
-------------+--------------+-----------------+------------------
 coordinator1 | default_pool |              -1 |                1
 datanode2    | default_pool |              -1 |                2
 datanode3    | default_pool |              -1 |                2
 datanode1    | default_pool |              -1 |                2
(4 rows)
```

7.3.2.6 GLOBAL_WLM_RESPOOL_CPU_INFO

This view is used to view the real-time CPU usage of resource pools on all nodes in a distributed system. Only the user with the SYSADMIN or MONADMIN permission can query this view.

This view can be used to query only the data on the primary DN node. You need to log in to the standby node and use the system function **gs_wlm_respool_cpu_info()** to query the usage of CPU.

Table 7-190 GLOBAL_WLM_RESPOOL_CPU_INFO columns

Name	Type	Description
node_name	name	Node name.
respool_name	text	Name of a resource pool.
control_group	text	Cgroup name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
cpu_affinity	text	Value of cores bound to the CPU.
cpu_usage	integer	CPU usage of a resource pool. **NOTE** If the CN and DN are deployed together, the CN and DN share the same CPU resources. Therefore, the **cpu_usage** values of the CN and DN are the same. If the CN and DN are independently deployed, the **cpu_usage** values of the CN and DN are displayed separately.

Example:

```
gaussdb=# SELECT * FROM GLOBAL_WLM_RESPOOL_CONCURRENCY_INFO;
 node_name   | respool_name |    control_group    | cpu_affinity | cpu_usage
--------------+--------------+---------------------+--------------+-----------
 coordinator1 | default_pool | DefaultClass:Medium | 0-71         |         0
 datanode1    | default_pool | DefaultClass:Medium | 0-71         |         0
 datanode2    | default_pool | DefaultClass:Medium | 0-71         |         0
 datanode3    | default_pool | DefaultClass:Medium | 0-71         |         0
(4 rows)
```

7.3.2.7 GLOBAL_WLM_RESPOOL_IO_INFO

This view is used to view the real-time I/O usage of resource pools on all nodes in a distributed system. Only the user with the SYSADMIN or MONADMIN permission can query this view.

This view can be used to query only the data on the primary DN node. You need to log in to the standby node and use the system function **gs_wlm_respool_io_info()** to query the usage of I/Os.

Table 7-191 GLOBAL_WLM_RESPOOL_IO_INFO columns

Name	Type	Description
node_name	name	Node name.
respool_name	text	Name of a resource pool.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
io_limits	integer	Upper limit of IOPS. The value **0** indicates there is no limit. • Row-store: The unit is determined by GUC parameter **io_control_unit**, which is used to count the number of I/Os during I/O control. The value of **io_control_unit** is **1**, which indicates one IOPS.
io_priority	text	I/O priority set for jobs that consume I/O resources. It takes effect when the I/O usage reaches 90%. **None** indicates there is no control.
current_iops	integer	Number of times that the current I/O has been triggered. The current I/O statistics occasionally exceed the upper limit, which is related to the I/O statistics algorithm and is normal.

Example:

```
gaussdb=# SELECT * FROM GLOBAL_WLM_RESPOOL_CONCURRENCY_INFO;
 node_name   | respool_name | io_limits | io_priority | current_iops
--------------+--------------+-----------+-------------+--------------
 coordinator1 | default_pool |         0 | None        |            0
 datanode2    | default_pool |         0 | None        |            0
 datanode3    | default_pool |         0 | None        |            0
 datanode1    | default_pool |         0 | None        |            0
(4 rows)
```

7.3.2.8 GLOBAL_WLM_RESPOOL_MEMORY_INFO

This view is used to view the real-time memory usage of resource pools on all nodes in a distributed system. Only the user with the SYSADMIN or MONADMIN permission can query this view.

This view can be used to query only the data on the primary DN node. You need to log in to the standby node and use the system function **gs_wlm_respool_memory_info()** to query the memory usage.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-192 GLOBAL_WLM_RESPOOL_MEMORY_INFO columns

Name	Type	Description
node_name	name	Node name.
respool_name	text	Name of a resource pool.
max_dynamic_memory	text	Maximum dynamic memory that can be used.
current_dynamic_memory	text	Used dynamic memory.
max_shared_memory	text	Maximum shared memory that can be used.
current_shared_memory	text	Shared memory that has been used.
shared_memory_hits_percent	integer	This function is unavailable and is reserved for future evolution. This column is reserved and the value is **0**.

Example:

```
gaussdb=# SELECT * FROM GLOBAL_WLM_RESPOOL_CONCURRENCY_INFO;
 node_name   | respool_name | max_dynamic_memory | current_dynamic_memory | max_shared_memory |
current_shared_memory | shared_memory_hits_percent
-------------+--------------+--------------------+------------------------+-------------------+------------------------
+---------------------------
 coordinator1 | default_pool | -1                 | 9415kB                 | -1                | 3856kB
|                          0
 datanode1    | default_pool | -1                 | 10724kB                | -1                | 2808kB
|                          0
 datanode2    | default_pool | -1                 | 10724kB                | -1                | 2808kB
|                          0
 datanode3    | default_pool | -1                 | 10724kB                | -1                | 560kB
|                          0
(4 rows)
```

7.3.2.9 PGXC_WLM_REBUILD_USER_RESPOOL

Rebuilds a user's resource pool information in memory on all nodes. It is used as a remedy only when resource pool information is missing or disordered. Only users with the SYSADMIN permission can query this view.

NOTE

If no data is displayed in the queried view, contact Huawei technical support.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-193 PGXC_WLM_REBUILD_USER_RESPOOL columns

Name	Type	Description
gs_wlm_rebuild_user_resource_pool	Boolean	Rebuilds information about the user resource pool in the memory. **t** indicates success, and **f** indicates failure.

7.3.3 OLTP Table Compression

7.3.3.1 GS_ADM_ILMDATAMOVEMENTPOLICIES

GS_ADM_ILMDATAMOVEMENTPOLICIES displays the data movement summary of ILM policies, including the policy name, action type, and conditions. Only system administrators can access this system view.

Table 7-194 GS_ADM_ILMDATAMOVEMENTPOLICIES columns

Name	Type	Description
policy_name	character varying(128)	Name of an ADO policy, which is automatically generated. The name format is p+*Policy ID*.
action_type	character varying(11)	Action type. The current version supports only compression.
scope	character varying(7)	Scope. In the current version, only rows are supported.
compression_level	character varying(30)	Compression level. This parameter is available when the action type is compression.
tier_tablespace	character varying(128)	Target space. This parameter is available only when the action type is migration. The value is **null** in the current version.
tier_status	character varying(9)	Specifies whether the target space is read-only. The value is **null** in the current version.
condition_type	character varying(22)	Condition type. The current version supports only the last modification time.
condition_days	numeric	Number of days as a condition.
custom_function	character varying(128)	Name of a user-defined function. The value is **null** in the current version.
policy_subtype	character varying(10)	Policy subtype. The value is **null** in the current version.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
action_clause	clob	Text that can be automatically executed during policy execution. The value is **null** in the current version.
tier_to	character varying(10)	Type of the target database to which data is migrated. The value is **null** in the current version.
node_name	name	Name of a data source node.

7.3.3.2 GS_ADM_ILMOBJECTS

GS_ADM_ILMOBJECTS displays the brief information about all data objects to which ILM policies are applied and the corresponding policies, including the policy name, data object name, policy source, and policy enabling/disabling status. Only SYSADMINs can access this system view.

Table 7-195 GS_ADM_ILMOBJECTS columns

Name	Type	Description
policy_name	character varying(128)	Name of an ADO policy, which is automatically generated. The name format is p+*Policy ID*.
object_owner	character varying(128)	Owner of a data object.
object_name	character varying(128)	Data object name.
subobject_name	character varying(128)	Partition name of a data object.
object_type	character varying(24)	Data object type. Value range: • **TABLE**: table. • **TABLE PARTITION**: partition. • **TABLE SUBPARTITION**: subpartition. • **INDEX**: ordinary index. • **GLOBAL PARTITIONED INDEX**: global index. • **GLOBAL SECONDARY INDEX**: global secondary index. • **INDEX PARTITION**: index partition.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
inherited_from	character varying(20)	Object whose policy is inherited by the current policy. Value range: • **TABLE**: table. • **TABLE PARTITION**: partition. • **TABLE SUBPARTITION**: subpartition. • **INDEX**: ordinary index. • **POLICY NOT INHERITED**: not inherited.
tbs_inherited_from	character varying(30)	Tablespace whose policy is inherited by the current policy. In the current version, this parameter is set to **NULL**.
enabled	character varying(7)	Specifies whether the policy is enabled for the current object.
deleted	character varying(7)	Specifies that the ILM policy on the object is deleted. If the partition or the policy on the partition is deleted, the record (ILMOBJ) is deleted.
node_name	name	Name of a data source node.

7.3.3.3 GS_ADM_ILMPOLICIES

GS_ADM_ILMPOLICIES displays the brief information about an ILM policy, including the policy name, type, enabling status, disabling status, and deletion status. Only users with the SYSADMIN permission can access this system view.

Table 7-196 GS_ADM_ILMPOLICIES columns

Name	Type	Description
policy_name	character varying(128)	Name of an ADO policy, which is automatically generated. The name format is p+*Policy ID*.
policy_type	character varying(13)	Policy type.
tablespace	character varying(30)	Tablespace name. The value is **null** in the current version.
enabled	character varying(6)	Specifies whether the policy is enabled.
deleted	character varying(7)	Specifies whether the policy is deleted.
node_name	name	Name of a data source node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.3.4 GS_ADM_ILMEVALUATIONDETAILS

GS_ADM_ILMEVALUATIONDETAILS displays the evaluation details of an ADO task, including the task ID, policy information, object information, evaluation result, and ADO job name. By default, only SYSADMINs can access the system view.

Table 7-197 GS_ADM_ILMEVALUATIONDETAILS columns

Name	Type	Description
task_id	bigint	ID of an ADO task.
policy_name	character varying(128)	Name of an ADO policy, which is automatically generated. The name format is p+*Policy ID*.
object_owner	character varying(128)	Name of the schema where the data object is located.
object_name	character varying(128)	Data object name.
subobject_name	character varying(128)	Partition name of a data object.
object_type	character varying(24)	Data object type. • **TABLE**: table. • **TABLE PARTITION**: partition. • **TABLE SUBPARTITION**: subpartition. • **INDEX**: ordinary index. • **GLOBAL PARTITIONED INDEX**: global index. • **GLOBAL SECONDARY INDEX**: global secondary index. • **INDEX PARTITION**: index partition.
selected_for_execution	character varying(42)	Evaluation result: • **SELECTED FOR EXECUTION**: The evaluation is passed. • **PRECONDITION NOT SATISFIED**: The evaluation fails. • **JOB ALREADY EXISTS**: The task already exists.
job_name	character varying(128)	Name of the task that executes a specific ADO job. If the evaluation result is "passed", this parameter has a value.
comments	character varying(4000)	Reserved column.
node_name	name	Name of a data source node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.3.5 GS_ADM_ILMPARAMETERS

GS_ADM_ILMPARAMETERS displays the environment parameters related to ILM scheduling and execution. The parameters can be modified through the DBE_ILM_ADMIN.CUSTOMIZE_ILM API. By default, only users with the SYSADMIN permission can access the system view.

Table 7-198 GS_ADM_ILMPARAMETERS columns

Name	Type	Description
name	character varying(128)	ADO-related environment parameters, which can be modified through the DBE_ILM_ADMIN.CUSTOMIZE_ILM API.
value	numeric	Parameter value.

7.3.3.6 GS_ADM_ILMRESULTS

GS_ADM_ILMRESULTS displays the execution details of an ADO job, including the task ID, job name, job status, and job time. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized.

Table 7-199 GS_ADM_ILMRESULTS columns

Name	Type	Description
task_id	bigint	ID of an ADO task.
job_name	character varying(128)	ADO job name.
job_state	character varying(35)	ADO job status.
start_time	timestamp with time zone	Time when a job starts to be scheduled.
completion_time	timestamp with time zone	Completion time.
comments	character varying(4000)	If a job fails, the failure cause is recorded here.
statistics	clob	Statistics.
node_name	name	Name of a data source node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.3.7 GS_ADM_ILMTASKS

GS_ADM_ILMTASKS displays the brief information about an ADO task, including the task ID, task owner, status, and time. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized.

Table 7-200 GS_ADM_ILMTASKS columns

Name	Type	Description
task_id	bigint	ID of an ADO task.
task_owner	character varying(128)	User who initiates an ADO task. This column exists only in the ADM view.
state	character varying(9)	Status. • **INACTIVE** • **ACTIVE** • **COMPLETED** • **UNKNOWN**: being evaluated.
creation_time	timestamp with time zone	Creation time.
start_time	timestamp with time zone	Time when the status changes to active.
completion_time	timestamp with time zone	Completion time.
node_name	name	Name of a data source node.

7.3.3.8 GS_MY_ILMEVALUATIONDETAILS

GS_MY_ILMEVALUATIONDETAILS displays the evaluation details of an ADO task, including the task ID, policy information, object information, evaluation result, and ADO job name.

Table 7-201 GS_MY_ILMEVALUATIONDETAILS columns

Name	Type	Description
task_id	bigint	ID of an ADO task.
policy_name	character varying(128)	Name of an ADO policy, which is automatically generated. The name format is p+*Policy ID*.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
object_owner	character varying(128)	Schema name of a data object.
object_name	character varying(128)	Data object name.
subobject_name	character varying(128)	Partition name of a data object.
object_type	character varying(24)	Data object type. • **TABLE**: table. • **TABLE PARTITION**: partition. • **TABLE SUBPARTITION**: subpartition. • **INDEX**: ordinary index. • **GLOBAL PARTITIONED INDEX**: global index. • **GLOBAL SECONDARY INDEX**: global secondary index. • **INDEX PARTITION**: index partition.
selected_for_execution	character varying(42)	Evaluation result: • **SELECTED FOR EXECUTION**: The evaluation is passed. • **PRECONDITION NOT SATISFIED**: The evaluation fails. • **JOB ALREADY EXISTS**: The task already exists.
job_name	character varying(128)	Name of the task that executes a specific ADO job. If the evaluation result is "passed", this parameter has a value.
comments	character varying(4000)	Reserved column.
node_name	name	Name of a data source node.

7.3.3.9 GS_MY_ILMRESULTS

GS_MY_ILMRESULTS displays the execution details of an ADO job, including the task ID, job name, job status, and job time.

Table 7-202 GS_MY_ILMRESULTS columns

Name	Type	Description
task_id	bigint	ID of an ADO task.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
job_name	character varying(128)	ADO job name.
job_state	character varying(35)	ADO job status.
start_time	timestamp with time zone	Time when a job starts to be scheduled.
completion_time	timestamp with time zone	Completion time.
comments	character varying(4000)	If a job fails, the failure cause is recorded here.
statistics	clob	Statistics.
node_name	name	Name of a data source node.

7.3.3.10 GS_MY_ILMTASKS

GS_MY_ILMTASKS displays the brief information about an ADO task, including the task ID, task owner, status, and time.

Table 7-203 GS_MY_ILMTASKS columns

Name	Type	Description
task_id	bigint	ID of an ADO task.
task_owner	character varying(128)	User who initiates an ADO task.
state	character varying(9)	Status. • **INACTIVE** • **ACTIVE** • **COMPLETED** • **UNKNOWN**: being evaluated.
creation_time	timestamp with time zone	Creation time.
start_time	timestamp with time zone	Time when the status changes to active.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
completion_time	timestamp with time zone	Completion time.
node_name	name	Name of a data source node.

7.3.3.11 GS_MY_ILMDATAMOVEMENTPOLICIES

GS_MY_ILMDATAMOVEMENTPOLICIES displays the data movement summary of ILM policies, including the policy name, action type, and conditions.

Table 7-204 GS_MY_ILMDATAMOVEMENTPOLICIES columns

Name	Type	Description
policy_name	character varying(128)	Name of an ADO policy, which is automatically generated. The name format is p+*Policy ID*.
action_type	character varying(11)	Action type. The current version supports only compression.
scope	character varying(7)	Scope. In the current version, only rows are supported.
compression_level	character varying(30)	Compression level. This parameter is available when the action type is compression.
tier_tablespace	character varying(128)	Target space. This parameter is available only when the action type is migration. The value is **null** in the current version.
tier_status	character varying(9)	Specifies whether the target space is read-only. The value is **null** in the current version.
condition_type	character varying(22)	Condition type. The current version supports only the last modification time.
condition_days	numeric	Number of days as a condition.
custom_function	character varying(128)	Name of a user-defined function. The value is **null** in the current version.
policy_subtype	character varying(10)	Policy subtype. The value is **null** in the current version.
action_clause	clob	Text that can be automatically executed during policy execution. The value is **null** in the current version.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
tier_to	character varying(10)	Type of the target database to which data is migrated. The value is **null** in the current version.
node_name	name	Name of a data source node.

7.3.3.12 GS_MY_ILMOBJECTS

GS_MY_ILMOBJECTS displays the brief information about all data objects to which ILM policies are applied and the corresponding policies, including the policy name, data object name, policy source, and policy enabling/disabling status.

Table 7-205 GS_MY_ILMOBJECTS

Name	Type	Description
policy_name	character varying(128)	Name of an ADO policy, which is automatically generated. The name format is p+*Policy ID*.
object_owner	character varying(128)	Schema name of a data object.
object_name	character varying(128)	Data object name.
subobject_name	character varying(128)	Partition name of a data object.
object_type	character varying(24)	Data object type. • **TABLE**: table. • **TABLE PARTITION**: partition. • **TABLE SUBPARTITION**: subpartition. • **INDEX**: ordinary index. • **GLOBAL PARTITIONED INDEX**: global index. • **GLOBAL SECONDARY INDEX**: global secondary index. • **INDEX PARTITION**: index partition.
inherited_from	character varying(20)	Object whose policy is inherited by the current policy. The options are as follows: • **TABLE**: table. • **TABLE PARTITION**: partition. • **TABLE SUBPARTITION**: subpartition. • **INDEX**: ordinary index. • **POLICY NOT INHERITED**: not inherited.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
tbs_inherited_from	character varying(30)	Tablespace whose policy is inherited by the current policy. The value is **null** in the current version.
enabled	character varying(7)	Specifies whether the policy is enabled for the current object.
deleted	character varying(7)	Specifies that the ILM policy on the object is deleted. If the partition or the policy on the partition is deleted, the record (ILMOBJ) is deleted.
node_name	name	Name of a data source node.

7.3.3.13 GS_MY_ILMPOLICIES

GS_MY_ILMPOLICIES displays the brief information about an ILM policy, including the policy name, type, enabling status, disabling status, and deletion status.

Table 7-206 GS_MY_ILMPOLICIES

Name	Type	Description
policy_name	character varying(128)	Name of an ADO policy, which is automatically generated. The name format is p+*Policy ID*.
policy_type	character varying(13)	Policy type.
tablespace	character varying(30)	Tablespace name. This parameter has a value when the policy is created on a tablespace. The value is **null** in the current version.
enabled	character varying(6)	Specifies whether the policy is enabled.
deleted	character varying(7)	Specifies whether the policy is deleted.
node_name	name	Name of a data source node.

7.3.4 Communications

7.3.4.1 COMM_CLIENT_INFO

COMM_CLIENT_INFO displays information about active client connections of a single node (You can query this view on a DN to view the information about the connection between the CN and DN). By default, only users with the SYSADMIN permission can access this view.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-207 COMM_CLIENT_INFO columns

Name	Type	Description
node_name	text	Name of the current DN, for example, dn_6001_6002_6003.
app	text	Query a view on a DN. The app displays the client connected to the current DN, such as the coordinator (CN), GTM, or DN.
tid	bigint	Thread ID of the current thread.
lwtid	integer	Lightweight thread ID of the current thread.
query_id	bigint	Query ID. It is equivalent to **debug_query_id**.
socket	integer	**socket fd** is displayed if the connection is a physical connection.
remote_ip	text	Peer node IP address.
remote_port	text	Peer node port.
logic_id	integer	Displayed if the connection is a logical connection.

7.3.4.2 GLOBAL_COMM_CLIENT_INFO

GLOBAL_COMM_CLIENT_INFO queries information about active client connections of global nodes in a cluster. By default, only users with the SYSADMIN permission can access this system view.

Table 7-208 GLOBAL_COMM_CLIENT_INFO columns

Name	Type	Description
node_name	text	Current node name.
app	text	Client that connects to the current DN, such as a CN, GTM, or DN.
tid	bigint	Thread ID of the current thread.
lwtid	integer	Lightweight thread ID of the current thread.
query_id	bigint	Query ID. It is equivalent to **debug_query_id**.
socket	integer	Displayed if the connection is a physical connection.
remote_ip	text	Peer node IP address.
remote_port	text	Peer node port.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
logic_id	integer	Displayed if the connection is a logical connection.

7.3.4.3 GS_COMM_LISTEN_ADDRESS_EXT_INFO

GS_COMM_LISTEN_ADDRESS_EXT_INFO queries DFX information, such as threads, sessions, and socket, for connecting the extended IP addresses of the current DN. For details about the application scenarios, see **gs_comm_listen_address_ext_info()**.

Table 7-209 GS_COMM_LISTEN_ADDRESS_EXT_INFO columns

Name	Type	Description
node_name	text	Name of the current instance.
app	text	Client connected to the DN.
tid	bigint	Thread ID of the current thread.
lwtid	integer	Lightweight thread ID of the current thread.
query_id	bigint	Query ID of the current thread.
socket	integer	Socket FD of the current physical connection.
remote_ip	text	Peer IP address of the current connection.
remote_port	text	Peer port of the current connection.
local_ip	text	Local IP address of the current connection.
local_port	text	Local port of the current connection.

7.3.4.4 GS_GET_LISTEN_ADDRESS_EXT_INFO

GS_GET_LISTEN_ADDRESS_EXT_INFO queries the extended IP address configured for the current DN instance. For details about the application scenarios, see **gs_get_listen_address_ext_info()**.

Table 7-210 GS_GET_LISTEN_ADDRESS_EXT_INFO columns

Name	Type	Description
node_name	text	DN name.
host	text	Listening IP address of a DN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
port	bigint	Listening port of a DN.
ext_listen_ip	text	Extended IP address configured for a DN.

7.3.4.5 GS_LIBCOMM_FD_INFO

Records information about socket persistent connections of the libcomm communications library. Only the user with the PUBLIC permission can query this view.

Table 7-211 GS_LIBCOMM_FD_INFO columns

Name	Type	Description
node_name	text	DN name.
ip	text	IP address of the DN.
ctrl_port	bigint	Control channel port.
data_port	bigint	Data channel port.
remote_name	text	Instance name of the DN at the peer site.
remote_ip	text	Peer IP address.
remote_port	bigint	Peer port number.
local_ip	text	Local IP address.
local_port	bigint	Local port number.
fd	bigint	Socket connection.
type	text	Type of the TCP connection channel. The options are as follows: • **DATA_SEND_FD**: socket of the sender of the data channel • **DATA_RECV_FD**: socket of the receiver of the data channel. • **CTL_SEND_FD**: socket of the sender of the control channel. • **CTL_RECV_FD**: socket of the receiver of the control channel.

7.3.4.6 GS_LIBCOMM_MEMORY_INFO

Records the basic configuration information used by the internal memory of the current libcomm communications library. Only the user with the PUBLIC permission can query this view.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-212 GS_LIBCOMM_MEMORY_INFO columns

Name	Type	Description
node_name	text	DN name.
current_used_memory	bigint	Memory used by the libcomm communications library, in bytes.
current_data_item_num	bigint	Number of data blocks in the libcomm lockless queue.
init_mailbox_memory	bigint	Memory used by libcomm to initialize the mailbox, in bytes.
max_datanode	bigint	Maximum number of DNs supported by the libcomm communications library.
max_stream	bigint	Maximum number of data streams supported by a single TCP connection of the libcomm.
max_quota_size	bigint	Total buffer size of the libcomm logical connection, in bytes.
max_usable_memory	bigint	Maximum memory that can be used by the libcomm communications library cache, in bytes.
max_memory_pool	bigint	Total size of the memory resource pool that can be used by the libcomm communications library on each DN. The unit is byte.

7.3.4.7 GS_STATIC_THREADPOOL_CTRL_STATUS

GS_STATIC_THREADPOOL_CTRL_STATUS returns statistics about static threads related to the thread pool of the current instance. When the thread pool is enabled, the CN thread pool can be queried using this view. A blank line is returned when a DN is connected for query or the thread pool is disabled. If **static_thread_pool_num** is set to **0** or less than the number of groups in the thread pool, the default value of the static thread pool is **0**. Only the user with the PUBLIC permission can query this view.

Table 7-213 GS_STATIC_THREADPOOL_CTRL_STATUS columns

Name	Type	Description
node_name	text	Instance name.
group_id	integer	Group ID of the thread pool.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
worker_info	text	Dynamic statistics about the running thread pool of the current group. The information includes: **default** and **default_s**: number of threads, referring to the default number of threads in the dynamic thread pool and that of the static thread pool, respectively. **expect** and **expect_s**: number of threads, referring to the expected number of threads in the dynamic thread pool and that of the static thread pool, respectively. **actual**: actual number of running threads, including dynamic pool threads and static pool threads. **static threads limit**: indicates the number of static pool threads configured for the current group. **has static threads**: specifies whether a static pool is created for the current group. The default value is **0**, indicating that no user is created. **idle static threads**: number of idle static pool threads. **wait session num**: number of waiting sessions.

7.3.4.8 PG_COMM_DELAY

PG_COMM_DELAY displays the communications library delay status for a single DN.

Table 7-214 PG_COMM_DELAY columns

Name	Type	Description
node_name	text	Node name.
remote_name	text	Name of the peer node.
remote_host	text	IP address of the peer node
stream_num	integer	Number of logical stream connections used by the current physical connection
min_delay	integer	Minimum delay of the current physical connection within 1 minute, in microsecond. **NOTE** A negative result is invalid. Wait until the delay status is updated and query again.
average	integer	Average delay of the current physical connection within 1 minute, in microsecond.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
max_delay	integer	Maximum delay of the current physical connection within 1 minute, in microsecond.

7.3.4.9 PG_COMM_RECV_STREAM

PG_COMM_RECV_STREAM displays the receiving stream status of all the communications libraries for a single DN.

Table 7-215 PG_COMM_RECV_STREAM columns

Name	Type	Description
node_name	text	Node name.
local_tid	bigint	ID of the thread using this stream.
remote_name	text	Name of the peer node.
remote_tid	bigint	Peer thread ID.
idx	integer	Peer DN ID in the local DN.
sid	integer	Stream ID in the physical connection.
tcp_sock	integer	TCP socket used in the stream.
state	text	Status of the stream. • **UNKNOWN**: The logical connection status is unknown. • **READY**: The logical connection is ready. • **RUN**: The logical connection sends packets normally. • **HOLD**: The logical connection is waiting to send packets. • **CLOSED**: The logical connection is closed. • **TO_CLOSED**: The logical connection will be closed.
query_id	bigint	**debug_query_id** corresponding to the stream.
pn_id	integer	**plan_node_id** of the query executed by the stream.
send_smp	integer	**smpid** of the sender of the query executed by the stream.
recv_smp	integer	**smpid** of the receiver of the query executed by the stream.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
recv_bytes	bigint	Total data volume received by the stream, in byte.
time	bigint	Current service life of the stream, in ms.
speed	bigint	Average receiving rate of the stream, in byte/s.
quota	bigint	Current communication quota of the stream, in bytes.
buff_usize	bigint	Current size of the data cache of the stream, in byte.

7.3.4.10 PG_COMM_SEND_STREAM

PG_COMM_SEND_STREAM displays the sending stream status of all the communication libraries for a single DN.

Table 7-216 PG_COMM_SEND_STREAM columns

Name	Type	Description
node_name	text	Node name.
local_tid	bigint	ID of the thread using this stream.
remote_name	text	Name of the peer node.
remote_tid	bigint	Peer thread ID.
idx	integer	Peer DN ID in the local DN.
sid	integer	Stream ID in the physical connection.
tcp_sock	integer	TCP socket used in the stream.
state	text	Status of the stream. • **UNKNOWN**: The logical connection status is unknown. • **READY**: The logical connection is ready. • **RUN**: The logical connection sends packets normally. • **HOLD**: The logical connection is waiting to send packets. • **CLOSED**: The logical connection is closed. • **TO_CLOSED**: The logical connection will be closed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
query_id	bigint	**debug_query_id** corresponding to the stream.
pn_id	integer	**plan_node_id** of the query executed by the stream.
send_smp	integer	**smpid** of the sender of the query executed by the stream.
recv_smp	integer	**smpid** of the receiver of the query executed by the stream.
send_bytes	bigint	Total data volume sent by the stream, in bytes.
time	bigint	Current lifecycle of a stream. The unit is ms.
speed	bigint	Average sending rate of the stream, in byte/s.
quota	bigint	Current communication quota of the stream, in bytes.
wait_quota	bigint	Extra time generated when the stream waits the quota value, in ms.

7.3.4.11 PG_COMM_STATUS

PG_COMM_STATUS displays the communications library status for a single DN.

Table 7-217 PG_COMM_STATUS columns

Name	Type	Description
node_name	text	Node name.
rxpck_rate	integer	Receiving rate of the communications library on the node, in byte/s.
txpck_rate	integer	Sending rate of the communications library on the node, in byte/s.
rxkbyte_rate	bigint	Receiving rate of the communications library on the node, in KB/s.
txkbyte_rate	bigint	Sending rate of the communications library on the node, in KB/s.
buffer	bigint	Cache size of the cmailbox.
memkbyte_libcomm	bigint	Communication memory size of the **libcomm** thread, in bytes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
memkbyte_libpq	bigint	Communication memory size of the **libpq** thread, in bytes.
used_pm	integer	Real-time usage of the **postmaster** thread.
used_sflow	integer	Real-time usage of the **gs_sender_flow_controller** thread.
used_rflow	integer	Real-time usage of the **gs_receiver_flow_controller** thread.
used_rloop	integer	Highest real-time usage among multiple **gs_receivers_loop** threads.
stream	integer	Total number of used logical connections.

7.3.4.12 PG_GET_INVALID_BACKENDS

PG_GET_INVALID_BACKENDS displays information about current standby DN background threads connected to the CN. Only system administrators and monitor administrators can access this view.

Table 7-218 PG_GET_INVALID_BACKENDS columns

Name	Type	Description
pid	bigint	Thread ID
node_name	text	Node information connected to the background thread.
dbname	name	Name of the connected database.
backend_start	timestamp with time zone	Background thread startup time.
query	text	Query statement executed by the background thread.

7.3.4.13 PG_POOLER_STATUS

PG_POOLER_STATUS queries the cache connection status in the pooler. This view can only query on the CN, and displays the connection cache information about the pooler module.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-219 PG_POOLER_STATUS columns

Name	Type	Description
database	text	Database name.
user_name	text	Username.
tid	bigint	• ID of the thread connected to the CN for non-thread pool logic. • Session ID for connecting to the CN for thread pool logic.
node_oid	bigint	OID of the node connected
node_name	name	Name of the node connected
in_use	boolean	Specifies whether the connection is currently used. • **t** (true): The connection is in use. • **f** (false): The connection is not in use.
local_host	text	Local IP address.
local_port	bigint	Local port number.
remote_host	text	IP address of the node connected.
node_port	bigint	Port number of the connected instance node.
fdsock	bigint	Peer socket.
remote_pid	bigint	If the peer site is not in the thread pool logic, the value is the thread ID of the peer site. If the peer site is in the thread pool logic, the value is the session ID of the peer site.
session_params	text	GUC session parameter delivered by the connection.
used_count	bigint	Number of reuse times of a connection.
idx	bigint	Logical connection ID of the connected instance node.
streamid	bigint	Stream ID corresponding to each logical connection.

7.3.4.14 PGXC_COMM_DELAY

PGXC_COMM_DELAY displays the communications library delay status for all DNs. Only the SYSADMIN and MONADMIN can query this view.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-220 PGXC_COMM_DELAY columns

Name	Type	Description
node_name	text	Node name.
remote_name	text	Name of the peer node.
remote_host	text	IP address of the peer node
stream_num	integer	Number of logical stream connections used by the current physical connection
min_delay	integer	Minimum delay of the current physical connection within 1 minute, in microsecond. **NOTE** A negative result is invalid. Wait until the delay status is updated and query again.
average	integer	Average delay of the current physical connection within 1 minute, in microsecond.
max_delay	integer	Maximum delay of the current physical connection within 1 minute, in microsecond.

7.3.4.15 PGXC_COMM_RECV_STREAM

PGXC_COMM_RECV_STREAM displays the receiving stream status of the communications libraries for all the DNs.

Table 7-221 PGXC_COMM_RECV_STREAM columns

Name	Type	Description
node_name	text	Node name.
local_tid	bigint	ID of the thread using this stream.
remote_name	text	Name of the peer node.
remote_tid	bigint	Peer thread ID.
idx	integer	Peer DN ID in the local DN.
sid	integer	Stream ID in the physical connection.
tcp_sock	integer	TCP socket used in the stream.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
state	text	Status of the stream. • **UNKNOWN**: The logical connection status is unknown. • **READY**: The logical connection is ready. • **RUN**: The logical connection sends packets normally. • **HOLD**: The logical connection is waiting to send packets. • **CLOSED**: The logical connection is closed. • **TO_CLOSED**: The logical connection will be closed.
query_id	bigint	**debug_query_id** corresponding to the stream.
pn_id	integer	**plan_node_id** of the query executed by the stream.
send_smp	integer	**smpid** of the sender of the query executed by the stream.
recv_smp	integer	**smpid** of the receiver of the query executed by the stream.
recv_bytes	bigint	Total data volume received by the stream, in byte.
time	bigint	Current service life of the stream, in ms.
speed	bigint	Average receiving rate of the stream, in byte/s.
quota	bigint	Current communication quota of the stream, in bytes.
buff_usize	bigint	Current size of the data cache of the stream, in byte.

7.3.4.16 PGXC_COMM_SEND_STREAM

PGXC_COMM_SEND_STREAM displays the sending stream status of the communications libraries for all the DNs.

Table 7-222 PGXC_COMM_SEND_STREAM columns

Name	Type	Description
node_name	text	Node name.
local_tid	bigint	ID of the thread using this stream.
remote_name	text	Name of the peer node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
remote_tid	bigint	Peer thread ID.
idx	integer	Peer DN ID in the local DN.
sid	integer	Stream ID in the physical connection.
tcp_sock	integer	TCP socket used in the stream.
state	text	Status of the stream. The options are as follows: • **UNKNOWN**: The connection status is unknown. • **READY**: The connection is ready. • **RUN**: The connection sends packets normally. • **HOLD**: The connection is waiting to send packets. • **CLOSED**: The connection is closed. • **TO_CLOSED**: The connection will be closed.
query_id	bigint	**debug_query_id** corresponding to the stream.
pn_id	integer	**plan_node_id** of the query executed by the stream.
send_smp	integer	**smpid** of the sender of the query executed by the stream.
recv_smp	integer	**smpid** of the receiver of the query executed by the stream.
send_bytes	bigint	Total data volume sent by the stream, in bytes.
time	bigint	Current service life of the stream, in ms.
speed	bigint	Average sending rate of the stream, in byte/s.
quota	bigint	Current communication quota of the stream, in bytes.
wait_quota	bigint	Extra time generated when the stream waits the quota value, in ms.

7.3.4.17 PGXC_COMM_STATUS

PGXC_COMM_STATUS displays the communications library status for all the DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-223 PGXC_COMM_STATUS columns

Name	Type	Description
node_name	text	Node name.
rxpck_rate	integer	Receiving rate of the communications library on the node, in byte/s.
txpck_rate	integer	Sending rate of the communications library on a node, in bytes/s.
rxkbyte_rate	bigint	Receiving rate of the communications library on the bigint node, in KB/s.
txkbyte_rate	bigint	Sending rate of the communications library on the bigint node, in KB/s.
buffer	bigint	Cache size of the cmailbox.
memkbyte_libcomm	bigint	Communication memory size of the libcomm process, in bytes.
memkbyte_libpq	bigint	Communication memory size of the libpq process in bytes.
used_pm	integer	Real-time usage of the postmaster thread.
used_sflow	integer	Real-time usage of the **gs_sender_flow_controller** thread.
used_rflow	integer	Real-time usage of the **gs_receiver_flow_controller** thread.
used_rloop	integer	Highest real-time usage among multiple **gs_receivers_loop** threads.
stream	integer	Total number of used logical connections.

7.3.5 Segment-Page Storage

7.3.5.1 GLOBAL_GS_SEG_DATAFILES

GLOBAL_GS_SEG_DATAFILES displays information about data files in all tablespaces on all nodes, which can only be executed on the CN. Only an administrator can query the information.

Table 7-224 GLOBAL_GS_SEG_DATAFILES columns

Name	Type	Description
node_name	text	Node name.
file_name	text	Data file name. Example: **base/17467/2_fsm**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
file_id	integer	Data file ID. Value range: an int4 value in the range [1,5].
bucketnode	integer	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
forknum	integer	Fork type of a segment object. Value range: • **0**: main fork. • **1**: FSM fork. • **2**: VM fork.
tablespace_name	name	Name of the tablespace to which a data file belongs.
contents	text	Storage content of a data file. Value range: • **permanent**: permanent. • **unlogged**: no log. • **temporary**: global temporary. • **temporary2**: local temporary.
extent_size	integer	Extent size of a data file.
meta_blocks	bigint	Number of allocated metadata pages of a data file.
data_blocks	bigint	Number of allocated data pages of a data file.
total_blocks	bigint	Total number of physical pages in a data file.
high_water_mark	bigint	High watermark of the number of pages used by a data file.
utilization	real	Percentage of used blocks to the total number of blocks, that is, (**data_blocks** + **meta_blocks**)/**total_blocks**.

7.3.5.2 GLOBAL_GS_SEG_EXTENTS

GLOBAL_GS_SEG_EXTENTS displays the extent information about all tablespaces on all nodes, which can only be executed on the CN. This view displays all extents

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

of user segments, including segment heads, fork heads, and level-1 pages in file 1 and data extents in files 2 to 5. Only an administrator can query the information.

Table 7-225 GLOBAL_GS_SEG_EXTENTS columns

Name	Type	Description
node_name	text	Node name.
tablespace_name	name	Tablespace to which a segment object belongs.
bucketnode	integer	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
head_block_id	bigint	Page number of the segment header.
extent_id	integer	Logical extent number.
file_id	integer	ID of the data file where the extent is located. Value range: an int4 value in the range [1,5].
forknum	integer	Fork of a segment object. Value range: • **0**: main. • **1**: fsm. • **2**: vm.
block_id	bigint	Start page number in the data file where the extent is located.
blocks	integer	Extent size. The value can be **1**, **8**, **128**, **1024**, or **4096**.
usage_type	text	Usage type of the extension. Value range: • **segment head**: segment header. • **fork head**: fork header. • **level1 page**: level-1 page. • **data extent**: data extents.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.5.3 GLOBAL_GS_SEG_SEGMENTS

GLOBAL_GS_SEG_SEGMENTS displays the segment information of all tablespaces on all nodes, including the table, index, TOAST and TOAST INDEX segments, and fsm fork and vm fork segments. It can only be executed on the CN. Only an administrator can query the information.

Table 7-226 GLOBAL_GS_SEG_SEGMENTS columns

Name	Type	Description
node_name	text	Node name.
schema_name	name	Namespace to which a segment belongs.
segment_name	name	Segment object name, which is obtained from pg_class and **relname** in pg_partition.
partition_name	name	Partition name of the segment object; **NULL** if the segment object is not partitioned. Source: pg_partition.relname.
forknum	integer	Fork type of a segment object. The options are as follows: ● **0**: main fork. ● **1**: FSM fork. ● **2**: VM fork.
segment_type	text	Segment object type. Value range: ● **table**: segment-page ordinary table. ● **table partition**: segment-page partitioned table (main table and child table) and segment-page subpartitioned table (partitioned table in it). ● **table subpartition**: segment-page subpartitioned table (top node table and subpartitioned table). ● **index**: index of a segment-page ordinary table. ● **index partition**: index of a segment-page partitioned table or subpartitioned table. ● **global partition index**: global index for segment-page partitioned tables and subpartitioned tables. ● **toast**: segment-page TOAST table. ● **toast index**: index of a segment-page TOAST table.
tablespace_name	name	Tablespace to which a segment object belongs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
bucketnode	integer	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
head_block_id	bigint	Page number of the segment header.
contents	text	Storage content of a data file. The options are as follows: • **permanent**: permanent. • **unlogged**: no log. • **temporary**: global temporary. • **temporary2**: local temporary.
table_name	name	Name of the base table to which a segment belongs.
blocks	bigint	Number of logical pages of a segment.
total_blocks	bigint	Number of physical pages of a segment.
extents	integer	Number of logical extents of a segment.
total_extents	integer	Number of physical extents of a segment.
head_lsn	text	Segment header LSN.
level0_slots	bigint[]	Level-0 slot array of segment extent mapping.
level1_slots	bigint[]	Level-1 slot array of segment extent mapping.
fork_head	bigint[]	Fork head array of a segment.

7.3.5.4 GLOBAL_GS_SEG_SPC_EXTENTS

GLOBAL_GS_SEG_SPC_EXTENTS displays the information about used extents of all tablespaces on all nodes, which can only be executed on the CN. The output contains segment heads, fork heads, level-1 pages, data extents. Only an administrator can query the information.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-227 GLOBAL_GS_SEG_SPC_EXTENTS

Name	Type	Description
node_name	text	Node name.
tablespace_name	name	Tablespace name.
file_id	integer	Data file ID. Value range: an int4 value in the range [1,5].
bucketnode	integer	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
forknum	integer	Fork type of a segment object. Value range: • **0**: main fork • **1**: fsm fork • **2**: vm fork
block_id	bigint	Start page number of a data extent.
blocks	integer	Data extent size. The value can be **1**, **8**, **128**, **1024**, or **4096**.
contents	text	Storage content of a data file. Value range: • **permanent**: permanent • **unlogged**: no log • **temporary**: global temporary • **temporary2**: local temporary
in_used	text	Specifies whether a resource has been allocated. The value can be **Y** or **N**.
mapblock_location	text	Position of the extension in the map block. The format is (*page_id*, *offset*).
head_file_id	integer	Segment header file ID.
head_block_id	bigint	Page number of the segment header.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
usage_type	text	Extended usage type. The options are as follows: • **segment head** indicates the segment header. • **fork head** indicates the fork header. • **level1 page** indicates the level-1 page. • **data extent** indicates data extents.
remain_flag	text	Specifies whether it is a residual extent after the SHRINK operation. The value can be **Y** or **N**.
special_data	integer	Special data section of the inverse pointer corresponding to an extent.
ipblock_location	text	Position of the extended inverse pointer. The format is (*block_id*, *offset*).

7.3.5.5 GLOBAL_GS_SEG_SPC_SEGMENTS

GLOBAL_GS_SEG_SPC_SEGMENTS displays the information about used segments of all tablespaces on all nodes, which can only be executed on the CN. Only an administrator can query the information.

Table 7-228 GLOBAL_GS_SEG_SPC_SEGMENTS columns

Name	Type	Description
node_name	text	Node name.
tablespace_name	name	Tablespace name.
file_id	integer	Data file ID. Value range: an int4 value in the range [1,5].
bucketnode	integer	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
forknum	integer	Fork type of a segment object. The options are as follows: • **0**: main fork • **1**: fsm fork • **2**: vm fork
block_id	bigint	Start page number of a data extent.
blocks	integer	Data extent size. The value can be **1**, **8**, **128**, **1024**, or **4096**.
contents	text	Storage content of a data file. Value range: • **permanent**: permanent • **unlogged**: no log • **temporary**: global temporary • **temporary2**: local temporary
in_used	text	Specifies whether a resource has been allocated. The value can be **Y** or **N**.
mapblock_location	text	Position of the extension in the map block. The format is (*page_id*, *offset*).
head_file_id	integer	Segment header file ID.
head_block_id	bigint	Page number of the segment header.
usage_type	text	Extended usage type. The options are as follows: • **segment head** indicates the segment header. • **fork head** indicates the fork header. • **level1 page** indicates the level-1 page. • **data extent** indicates data extents.
remain_flag	text	Specifies whether it is a shrink residual extension. The value can be **Y** or **N**.
special_data	integer	Special data section of the inverse pointer corresponding to an extent.
ipblock_location	text	Position of the extended inverse pointer. The format is (*block_id*, *offset*).

7.3.5.6 GS_SEG_DATAFILES

GS_SEG_DATAFILES displays information about data files in all tablespaces. Only an administrator can query the information.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-229 GS_SEG_DATAFILES columns

Name	Type	Description
node_name	text	Node name.
file_name	text	Data file name, for example, **base/17467/2_fsm**.
file_id	integer	Data file ID. Value range: an int4 value in the range [1,5].
bucketnode	integer	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
forknum	integer	Segment fork type. Value range: • **0**: main fork. • **1**: FSM fork. • **2**: VM fork.
tablespace_name	name	Name of the tablespace to which a data file belongs.
contents	text	Storage content of a data file. Value range: • **permanent**: permanent. • **unlogged**: no log. • **temporary**: global temporary. • **temporary2**: local temporary.
extent_size	integer	Extent size of a data file.
meta_blocks	bigint	Number of allocated metadata pages of a data file.
data_blocks	bigint	Number of allocated data pages of a data file.
total_blocks	bigint	Total number of physical pages in a data file.
high_water_mark	bigint	High watermark of the number of pages used by a data file.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
utilization	real	Percentage of used blocks to the total number of blocks, that is, (**data_blocks** + **meta_blocks**)/**total_blocks**.

7.3.5.7 GS_SEG_DATAFILE_LAYOUT

GS_SEG_DATAFILE_LAYOUT displays the static layout of data files 1 to 5 in segment-page mode. Only an administrator can query the information.

Table 7-230 GS_SEG_DATAFILE_LAYOUT columns

Name	Type	Description
version	text	Segment-page version. Default value: **1.0**.
seg_storage_type	text	• **segment** indicates common segment-page data. • **hashbucket** indicates hash bucket data.
file_id	integer	Data file ID. Value range: an int4 value in the range [1,5].
section_id	integer	Data section ID of a data file.
section_type	text	Type of the data file section. Value range: • **file_header** indicates the file header. • **spc_header** indicates the space header. • **map_header** indicates the mapping header. • **map_pages** indicates the mapping page. • **ip_pages(inverse pointer pages)** indicates the inverse pointer page. • **data_pages** indicates the data page.
page_start	bigint	Start page number of the data section.
page_end	bigint	End page number of the data section.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
page_count	bigint	Number of pages in the data section.
total_size	bigint	Size of the data section. The unit is byte.

7.3.5.8 GS_SEG_EXTENTS

GS_SEG_EXTENTS displays extent information about all tablespaces. This view displays all extents of user segments, including segment heads, fork heads, and level-1 pages in file 1 and data extents in files 2 to 5. Only an administrator can query the information.

Table 7-231 GS_SEG_EXTENTS columns

Name	Type	Description
node_name	text	Node name.
tablespace_name	name	Tablespace to which a segment object belongs.
bucketnode	integer	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
head_block_id	bigint	Page number of the segment header.
extent_id	integer	Logical extent number.
file_id	integer	ID of the data file where the extent is located. Value range: an int4 value in the range [1,5].
forknum	integer	Fork type of a segment object. Value range: • **0**: main fork. • **1**: FSM fork. • **2**: VM fork.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
block_id	bigint	Start page number in the data file where the extent is located.
blocks	integer	Extent size. The value can be **1**, **8**, **128**, **1024**, or **4096**.
usage_type	text	Usage type of the extension. Value range: ● **segment head**: segment header. ● **fork head**: fork header. ● **level1 page**: level-1 page. ● **data extent**: data extents.

7.3.5.9 GS_SEG_SEGMENTS

GS_SEG_SEGMENTS displays information about segments in all tablespaces, including tables, indexes, TOAST, TOAST INDEX segments, and fsm fork and vm fork segments. Only an administrator can query the information.

Table 7-232 GS_SEG_SEGMENTS columns

Name	Type	Description
node_name	text	Node name.
schema_name	name	Namespace to which a segment belongs.
segment_name	name	Part object name, which is obtained from pg_class and **relname** in pg_partition.
partition_name	name	Partition name of the segment object (**NULL** if the segment object is not partitioned). Source: pg_partition.relname.
forknum	integer	Fork type of a segment object. Value range: ● **0**: main fork. ● **1**: FSM fork. ● **2**: VM fork.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
segment_type	text	Segment object type. Value range: • **table**: segment-page ordinary table. • **table partition**: segment-page partitioned table (main table and child table) and segment-page subpartitioned table (partitioned table in it). • **table subpartition**: segment-page subpartitioned table (top node table and subpartitioned table). • **index**: index of a segment-page ordinary table. • **index partition**: index of a segment-page partitioned table or subpartitioned table. • **global partition index**: global index for segment-page partitioned tables and subpartitioned tables. • **toast**: segment-page TOAST table. • **toast index**: index of a segment-page TOAST table.
tablespace_name	name	Tablespace to which a segment object belongs.
bucketnode	integer	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
head_block_id	bigint	Page number of the segment header.
contents	text	Storage content of a data file. Value range: • **permanent**: permanent. • **unlogged**: no log. • **temporary**: global temporary. • **temporary2**: local temporary.
table_name	name	Name of the base table to which a segment belongs.
blocks	bigint	Number of logical pages of a segment.
total_blocks	bigint	Number of physical pages of a segment.
extents	integer	Number of logical extents of a segment.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
total_extents	integer	Number of physical extents of a segment.
head_lsn	text	Segment header LSN.
level0_slots	bigint[]	Level-0 slot array of segment extent mapping.
level1_slots	bigint[]	Level-1 slot array of segment extent mapping.
fork_head	bigint[]	Fork head array of a segment.

7.3.5.10 GS_SEG_SEGMENT_LAYOUT

GS_SEG_SEGMENT_LAYOUT displays the static layout of segment-page files. Only an administrator can query the information.

Table 7-233 GS_SEG_SEGMENT_LAYOUT columns

Name	Type	Description
version	text	Segment-page version. Default value: **1.0**.
section_id	integer	ID of the data section divided by a segment.
section_type	text	Extent type of the segment data section. Value range: • **meta**: segment header. • **data**: data.
extent_size	integer	Extent size. The unit is byte.
extent_page_count	integer	Number of extent pages.
extent_count_start	bigint	Start extent number.
extent_count_end	bigint	End extent number.
total_size	bigint	Size of the segment data section. The unit is byte.

7.3.5.11 GS_SEG_SPC_EXTENTS

GS_SEG_SPC_EXTENTS displays information about the used extents of all tablespaces. The output contains segment heads, fork heads, level-1 pages, data extents. Only an administrator can query the information.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-234 GS_SEG_SPC_EXTENTS

Name	Type	Description
node_name	text	Node name.
tablespace_name	name	Tablespace name.
file_id	integer	Data file ID. Value range: an int4 value in the range [1,5].
bucketnode	integer	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
forknum	integer	Data file fork.
block_id	bigint	Start page number of a data extent.
blocks	integer	Data extent size. The value can be **1**, **8**, **128**, **1024**, or **4096**.
contents	text	Storage content of a data file. The options are as follows: • **permanent**: permanent. • **unlogged**: no log. • **temporary**: global temporary. • **temporary2**: local temporary.
in_used	text	Specifies whether a resource has been allocated. The value can be **Y** or **N**.
mapblock_location	text	Position of the extension in the map block. The format is (*page_id*, *offset*).
head_file_id	integer	Segment header file ID.
head_block_id	bigint	Page number of the segment header.
usage_type	text	Extended usage type. The options are as follows: • **segment head**: segment header. • **fork head**: fork header. • **level1 page**: level-1 page. • **data extent**: data extents.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
remain_flag	text	Specifies whether it is a residual extent after the SHRINK operation. The value can be **Y** or **N**.
special_data	integer	Special data section of the inverse pointer corresponding to an extent.
ipblock_location	text	Position of the extended inverse pointer. The format is (*block_id*, *offset*).

7.3.5.12 GS_SEG_SPC_SEGMENTS

GS_SEG_SPC_SEGMENTS displays information about used segments in all tablespaces. Only an administrator can query the information.

Table 7-235 GS_SEG_SPC_SEGMENTS columns

Name	Type	Description
node_name	text	Node name.
tablespace_name	name	Tablespace name.
file_id	integer	Data file ID. Value range: an int4 value in the range [1,5].
bucketnode	integer	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
forknum	integer	Fork type of a segment object. Value range: • **0**: main fork. • **1**: FSM fork. • **2**: VM fork.
block_id	bigint	Start page number of a data extent.
blocks	integer	Data extent size. The value can be **1**, **8**, **128**, **1024**, or **4096**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
contents	text	Storage content of a data file. Value range: • **permanent**: permanent. • **unlogged**: no log. • **temporary**: global temporary. • **temporary2**: local temporary.
in_used	text	Specifies whether a resource has been allocated. The value can be **Y** or **N**.
mapblock_location	text	Position of the extension in the map block. The format is (*page_id*, *offset*).
head_file_id	integer	Segment header file ID.
head_block_id	bigint	Page number of the segment header.
usage_type	text	Extended usage type. The options are as follows: • **segment head**: segment header. • **fork head**: fork header. • **level1 page**: level-1 page. • **data extent**: data extents.
remain_flag	text	Specifies whether it is a residual extent after the SHRINK operation. The value can be **Y** or **N**.
special_data	integer	Special data section of the inverse pointer corresponding to an extent.
ipblock_location	text	Position of the extended inverse pointer. The format is (*block_id*, *offset*).

7.3.5.13 GS_SEG_SPC_REMAIN_EXTENTS

GS_SEG_SPC_REMAIN_EXTENTS displays the residual isolated extent information of the index tablespace. The extent information is not included in the residual segment and can be used as an independent clearing unit. Only an administrator can query the information.

Table 7-236 GS_SEG_SPC_REMAIN_EXTENTS columns

Name	Type	Description
node_name	text	Node name.
tablespace_name	name	Tablespace name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
file_id	integer	Data file ID. Value range: an int4 value in the range [1,5].
bucketnode	integer	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
forknum	integer	Fork type of a segment object. Value range: • **0**: main fork • **1**: fsm fork • **2**: vm fork
block_id	bigint	Start page number of a data extent.
blocks	integer	Data extent size. The value can be **1**, **8**, **128**, **1024**, or **4096**.
contents	text	Storage content of a data file. Value range: • **permanent**: permanent • **unlogged**: no log • **temporary**: global temporary • **temporary2**: local temporary
in_used	text	Specifies whether a resource has been allocated. The value can be **Y** or **N**.
mapblock_location	text	Position of the extension in the map block. The format is (*page_id*, *offset*).
head_file_id	integer	Segment header file ID.
head_block_id	bigint	Page number of the segment header.
usage_type	text	Usage type of the extension. Value range: • **segment head** indicates the segment header. • **fork head** indicates the fork header. • **level1 page** indicates the level-1 page. • **data extent** indicates data extents.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
remain_flag	text	Specifies whether it is a residual extent after the SHRINK operation. The value can be **Y** or **N**.
special_data	integer	Special data section of the inverse pointer corresponding to an extent.
ipblock_location	text	Position of the extended inverse pointer. The format is (*block_id*, *offset*).

7.3.5.14 GS_SEG_SPC_REMAIN_SEGMENTS

GS_SEG_SPC_REMAIN_SEGMENTS displays the residual segment information of the index tablespace, including the main fork, fsm fork, and vm fork segments. Only an administrator can query the information.

Table 7-237 GS_SEG_SPC_REMAIN_SEGMENTS columns

Name	Type	Description
node_name	text	Node name.
tablespace_name	name	Tablespace name.
file_id	integer	Data file ID. Value range: an int4 value in the range [1,5].
bucketnode	integer	• **0** to **1023** indicate the bucket nodes of a hash bucket table. • **1024** indicates the bucket node of a segment-page ordinary table. • **1025** indicates the bucket node of a segment-page global temporary table. • **1026** indicates the bucket node of a segment-page unlogged table. • **1027** indicates the bucket node of a segment-page local temporary table.
forknum	integer	Data file fork.
block_id	bigint	Start page number of a data extent.
blocks	integer	Data extent size. The value can be **1**, **8**, **128**, **1024**, or **4096**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
contents	text	Storage content of a data file. Value range: • **permanent**: permanent • **unlogged**: no log • **temporary**: global temporary • **temporary2**: local temporary
in_used	text	Specifies whether a resource has been allocated. The value can be **Y** or **N**.
mapblock_location	text	Position of the extension in the map block. The format is (*page_id*, *offset*).
head_file_id	integer	Segment header file ID.
head_block_id	bigint	Page number of the segment header.
usage_type	text	Usage type of the extension. Value range: • **segment head** indicates the segment header. • **fork head** indicates the fork header. • **level1 page** indicates the level-1 page. • **data extent** indicates data extents.
remain_flag	text	Specifies whether it is a residual extent after the SHRINK operation. The value can be **Y** or **N**.
special_data	integer	Special data section of the inverse pointer corresponding to an extent.
ipblock_location	text	Position of the extended inverse pointer. The format is (*block_id*, *offset*).

NOTE

Due to the mechanism for determining transaction visibility, when transaction A creates a segment-page object and the transaction is not committed, other sessions can query the segment generated by transaction A when executing **GS_SEG_SPC_REMAIN_SEGMENTS**. Actually, this segment is not residual and will not be cleared even if the residual clearing function is automatically or manually executed. After transaction A ends, you can determine whether the segment is residual.

7.3.6 SPM

7.3.6.1 GS_SPM_SQL_BASELINE

GS_SPM_SQL_BASELINE is a system view of the SPM feature. Currently, this feature is not supported in the distributed system. This view is used to view the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

baseline information of the current user. Users with permissions higher than common users can access this view.

Table 7-238 GS_SPM_SQL_BASELINE columns

Name	Type	Description
sql_namespace	oid	OID of a schema.
sql_hash	bigint	Unique ID of an SQL statement in the current schema.
plan_hash	bigint	Unique ID of a plan in the current SQL statement.
outline	text	Outline text, which can be used to fix a group of hints of the current plan.
status	text	Status of a plan. Value range: • **UNACC**: indicates that the plan is not accepted. • **ACC**: indicates that the plan has been accepted. • **FIXED**: indicates a special ACC plan. The matching priority of this plan is higher than that of other ACC plans.
gplan	Boolean	Specifies whether the plan corresponding to the current outline is a gplan.
cost	double precision	Total plan cost.
sql_text	text	SQL text string.
param_num	integer	Number of SQL parameters.
source	text	Baseline source.
creation_time	timestamp with time zone	Time when the baseline is created.
modification_time	timestamp with time zone	Time when the baseline is modified.
jump_intercept_cnt	bigint	Number of baseline interception plan changes.
invalid	Boolean	Specifies whether the current baseline is invalid.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
namespace_oids	text	OID list in the current **search_path**. The OIDs must meet the following requirements: • OIDs of the schema on which the user has permissions. • OIDs of the existing schemas. • OIDs of the schemas where the non-local temporary tables are located.

7.3.6.2 GS_SPM_SQL_PARAM

GS_SPM_SQL_PARAM is a system view of the SPM feature. Currently, this feature is not supported in the distributed system. This view is used to view the SQL parameter information of the current user. Users with permissions higher than common users can access this view.

Table 7-239 GS_SPM_SQL_PARAM columns

Name	Type	Description
sql_namespace	oid	OID of a schema.
sql_hash	bigint	Unique ID of an SQL statement in the current schema.
sql_text	text	SQL text string.
position	integer	Position index of a parameter in the SQL statement, starting from 0.
datatype	integer	OID of the parameter type.
datatype_string	text	Character string of a parameter type.
value_string	text	Character string of a parameter value.
is_null	Boolean	Specifies whether a parameter value is **NULL**.
hash_value	bigint	Hash value of a parameter value.
creation_time	timestamp with time zone	Time when a record is created.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
namespace_oids	text	OID list in the current **search_path**. The OIDs must meet the following requirements: • OIDs of the schema on which the user has permissions. • OIDs of the existing schemas. • OIDs of the schemas where the non-local temporary tables are located.

7.3.6.3 GS_SPM_SQL_EVOLUTION

GS_SPM_SQL_EVOLUTION is a system view of the SPM feature. Currently, this feature is not supported in the distributed system. This view is used to view the planned evolution result of the current user. Users with permissions higher than common users can access this view.

Table 7-240 GS_SPM_SQL_EVOLUTION columns

Name	Type	Description
sql_namespace	oid	OID of a schema.
sql_hash	bigint	Unique ID of an SQL statement in the current schema.
plan_hash	bigint	Unique ID of a plan in the current SQL statement.
better	Boolean	Determines whether it is a positive evolution.
status	text	Specifies whether exceptions occur during the evolution. Value range: • **SUCCESS**: The evolution is successful. • **FAILED**: The evolution fails.
refer_plan	bigint	Plan hash used as a reference for report generation.
sql_text	text	SQL text string.
outline	text	Hint string list of the current plan.
reason	text	Content of the evolution report.
gplan	Boolean	Determines whether the plan is a gplan.
creation_time	timestamp with time zone	Time when the evolution result is created.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
namespace_oids	text	OID list in the current **search_path**. The OIDs must meet the following requirements: • OIDs of the schema on which the user has permissions. • OIDs of the existing schemas. • OIDs of the schemas where the non-local temporary tables are located.

7.3.6.4 GS_SPM_SYS_BASELINE

GS_SPM_SYS_BASELINE is a system view of the SPM feature. This feature is not supported in the distributed system. This view is used to view baseline information in the database. The user with the SYSADMIN permission and initial user can access this view.

Table 7-241 GS_SPM_SYS_BASELINE columns

Name	Type	Description
sql_hash	bigint	Unique ID of the SQL statement in the SPM.
plan_hash	bigint	Unique ID of a plan in the current SQL statement.
unique_sql_id	bigint	Unique ID of an SQL statement in the database.
outline	text	Outline text, which can be used to fix a group of hints of the current plan.
status	integer	Status of a plan. Value range: • **0** (UNACC): plan that is not accepted. • **1** (ACC): plan that has been accepted. • **2** (FIXED): a special ACC plan. The matching priority of this plan is higher than that of other ACC plan.
gplan	Boolean	Specifies whether the plan corresponding to the current outline is a gplan.
cost	double precision	Total plan cost.
sql_text	text	SQL text string.
param_num	integer	Number of SQL parameters.
source	text	Baseline source.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
creation_time	timestamp with time zone	Time when the baseline is created.
modification_time	timestamp with time zone	Time when the baseline is modified.
jump_intercept_cnt	bigint	Number of baseline interception plan changes.
invalid	Boolean	Specifies whether the current baseline is invalid.
namespace_oids	text	OID list in the current **search_path**. The OIDs must meet the following requirements: • OIDs of the schema on which the user has permissions. • OIDs of the existing schemas. • OIDs of the schemas where the non-local temporary tables are located.

7.3.7 Audit

7.3.7.1 ADM_AUDIT_OBJECT

ADM_AUDIT_OBJECT displays the audit trail records of all objects in the database. This view exists in both PG_CATALOG and SYS schemas. Only users with the AUDITADMIN attribute and the SELECT permission on ADM_AUDIT_OBJECT can view audit information. If separation of duties is disabled, users with the SYSADMIN attribute can also view audit information.

Table 7-242 ADM_AUDIT_OBJECT columns

Name	Type	Description
os_username	character varying(255)	Not supported. The value is **NULL**.
username	character varying(128)	Name of the user whose operation is audited, not the user ID.
userhost	character varying(128)	Not supported. The value is **NULL**.
terminal	character varying(255)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
timestamp	date	Date and time when an audit trail entry is created in the local database session time zone (user login date and time of the entry created by the audit session). **TIMESTAMP (0) WITHOUT TIME ZONE** is displayed in an O-compatible database. **NOTE** During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of this column is changed to **datea**.
owner	character varying(128)	Creator of the object affected by the operation.
obj_name	character varying(128)	Name of the object affected by the operation.
action_name	character varying(28)	Action name corresponding to the numeric code in the **ACTION** column in DBA_AUDIT_TRAIL **NOTE** The **action_name** column in GaussDB is inconsistent with the audit action of database ORA.
new_owner	character varying(128)	Not supported. The value is **NULL**.
new_name	character varying(128)	Not supported. The value is **NULL**.
ses_actions	character varying(19)	Not supported. The value is **NULL**.
comment_text	character varying(4000)	Not supported. The value is **NULL**.
sessionid	numeric	Not supported. The value is **NULL**.
entryid	numeric	Not supported. The value is **NULL**.
statementid	numeric	Not supported. The value is **NULL**.
returncode	numeric	Not supported. The value is **NULL**.
priv_used	character varying(40)	Not supported. The value is **NULL**.
client_id	character varying(128)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
econtext_id	character varying(64)	Not supported. The value is **NULL**.
session_cpu	numeric	Not supported. The value is **NULL**.
extended_timestamp	timestamp(6) with time zone	Timestamp when an audit trail entry is created (user login timestamp of the entry created by the audit session in UTC).
proxy_sessionid	numeric	Not supported. The value is **NULL**.
global_uid	character varying(32)	Not supported. The value is **NULL**.
instance_number	numeric	Not supported. The value is **NULL**.
os_process	character varying(16)	Not supported. The value is **NULL**.
transactionid	text	Identifier of the transaction that accesses or modifies an object. **NOTE** The type of the **transactionid** column in GaussDB must be the same as that in database ORA.
scn	numeric	Not supported. The value is **NULL**.
sql_bind	nvarchar2(2000)	Not supported. The value is **NULL**.
sql_text	character varying	SQL text of the query. **NOTE** The **sql_text** column in GaussDB is the parsed SQL description statement, which is not completely the same as the executed SQL statement.
obj_edition_name	character varying(128)	Not supported. The value is **NULL**.

7.3.7.2 ADM_AUDIT_SESSION

ADM_AUDIT_SESSION displays all audit trail records concerning CONNECT and DISCONNECT. The audit information of GaussDB is mainly queried through the gs_query_audit function. This view exists in both PG_CATALOG and SYS schemas. Only users with the AUDITADMIN attribute and the SELECT permission on ADM_AUDIT_SESSION can view audit information. If separation of duties is disabled, users with the SYSADMIN attribute can also view audit information.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-243 ADM_AUDIT_SESSION columns

Name	Type	Description
os_username	character varying(255)	Not supported. The value is **NULL**.
username	character varying(128)	Name of the user whose operation is audited, not the user ID.
userhost	character varying(128)	Not supported. The value is **NULL**.
terminal	character varying(255)	Not supported. The value is **NULL**.
timestamp	date	Date and time when an audit trail entry is created (user login date and time of the entry created by AUDIT SESSION). **NOTE** During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of this column is changed to **datea**.
action_name	character varying(28)	Action name corresponding to the numeric code in the action column in DBA_AUDIT_TRAIL. **NOTE** The **action_name** column in GaussDB is inconsistent with the audit action of database ORA.
logoff_time	date	Not supported. The value is **NULL**.
logoff_lread	numeric	Not supported. The value is **NULL**.
logoff_pread	numeric	Not supported. The value is **NULL**.
logoff_lwrite	numeric	Not supported. The value is **NULL**.
logoff_dlock	numeric	Not supported. The value is **NULL**.
sessionid	numeric	Not supported. The value is **NULL**.
returncode	numeric	Not supported. The value is **NULL**.
client_id	character varying(128)	Not supported. The value is **NULL**.
session_cpu	numeric	Not supported. The value is **NULL**.
extended_timestamp	timestamp(6) with time zone	Timestamp when an audit trail entry is created (user login timestamp of the entry created by AUDIT SESSION in UTC).
proxy_sessionid	numeric	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
global_uid	character varying(32)	Not supported. The value is **NULL**.
instance_number	numeric	Not supported. The value is **NULL**.
os_process	character varying(16)	Not supported. The value is **NULL**.

7.3.7.3 ADM_AUDIT_STATEMENT

ADM_AUDIT_STATEMENT displays all grant and revoke audit trail entries. The audit information of GaussDB is mainly queried through the gs_query_audit function. This view exists in both PG_CATALOG and SYS schemas. Only users with the AUDITADMIN attribute and the SELECT permission on ADM_AUDIT_STATEMENT can view audit information. If separation of duties is disabled, users with the SYSADMIN attribute can also view audit information.

Table 7-244 ADM_AUDIT_STATEMENT columns

Name	Type	Description
os_username	character varying(255)	Not supported. The value is **NULL**.
username	character varying(128)	Name of the user whose operation is audited, not the user ID.
userhost	character varying(128)	Not supported. The value is **NULL**.
terminal	character varying(255)	Not supported. The value is **NULL**.
timestamp	date	Date and time when an audit trail entry is created in the local database session time zone (user login date and time of the entry created by the audit session). **NOTE** During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of this column is changed to **datea**.
owner	character varying(128)	Creator of the object affected by the operation.
obj_name	character varying(128)	Name of the object affected by the operation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
action_name	character varying(28)	Action type corresponding to the code in the action column. **NOTE** The **action_name** column in GaussDB is inconsistent with the audit action of database ORA.
new_name	character varying(128)	Not supported. The value is **NULL**.
obj_privilege	character varying(32)	Not supported. The value is **NULL**.
sys_privilege	character varying(40)	Not supported. The value is **NULL**.
admin_option	character varying(1)	Not supported. The value is **NULL**.
grantee	character varying(128)	Not supported. The value is **NULL**.
audit_option	character varying(40)	Not supported. The value is **NULL**.
ses_actions	character varying(19)	Not supported. The value is **NULL**.
comment_text	character varying(4000)	Not supported. The value is **NULL**.
sessionid	numeric	Not supported. The value is **NULL**.
entryid	numeric	Not supported. The value is **NULL**.
statementid	numeric	Not supported. The value is **NULL**.
returncode	numeric	Not supported. The value is **NULL**.
priv_used	character varying(40)	Not supported. The value is **NULL**.
client_id	character varying(128)	Not supported. The value is **NULL**.
econtext_id	character varying(64)	Not supported. The value is **NULL**.
session_cpu	numeric	Not supported. The value is **NULL**.
extended_timestamp	timestamp(6) with time zone	Time zone of the timestamp when an audit trail entry is created (user login date and time of the entry created by the audit session in UTC).
proxy_sessionid	numeric	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
global_uid	character varying(32)	Not supported. The value is **NULL**.
instance_number	numeric	Not supported. The value is **NULL**.
os_process	character varying(16)	Not supported. The value is **NULL**.
transactionid	text	Identifier of the transaction that accesses or modifies an object. **NOTE** The type of the **transactionid** column in GaussDB must be the same as that in database ORA.
scn	numeric	Not supported. The value is **NULL**.
sql_bind	nvarchar2(2000)	Not supported. The value is **NULL**.
sql_text	character varying(2000)	SQL text of the query. **NOTE** The **sql_text** column in GaussDB is the parsed SQL description statement, which is not completely the same as the executed SQL statement.
obj_edition_name	character varying(128)	Not supported. The value is **NULL**.

7.3.7.4 ADM_AUDIT_TRAIL

ADM_AUDIT_TRAIL displays all standard audit trail entries. The audit information of GaussDB is mainly queried through the gs_query_audit function. This view exists in both PG_CATALOG and SYS schemas. Only users with the AUDITADMIN attribute and the SELECT permission on ADM_AUDIT_TRAIL can view audit information. If separation of duties is disabled, users with the SYSADMIN attribute can also view audit information.

CAUTION

In a distributed system, the gs_query_audit function can only be used to query the audit information of the current node.

Table 7-245 ADM_AUDIT_TRAIL columns

Name	Type	Description
os_username	character varying(255)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
username	character varying(128)	Name of the user whose operation is audited, not the user ID.
userhost	character varying(128)	Not supported. The value is **NULL**.
terminal	character varying(255)	Not supported. The value is **NULL**.
timestamp	date	Date and time when an audit trail entry is created in the local database session time zone (user login date and time of the entry created by the audit session). **NOTE** During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of this column is changed to **datea**.
owner	character varying(128)	Creator of the object affected by the operation.
obj_name	character varying(128)	Name of the object affected by the operation.
action	numeric	Not supported. The value is **NULL**.
action_name	character varying(28)	Action type corresponding to the code in the action column. **NOTE** The **action_name** column in GaussDB is inconsistent with the audit action of database ORA.
new_owner	character varying(128)	Not supported. The value is **NULL**.
new_name	character varying(128)	Not supported. The value is **NULL**.
obj_privilege	character varying(32)	Not supported. The value is **NULL**.
sys_privilege	character varying(40)	Not supported. The value is **NULL**.
admin_option	character varying(1)	Not supported. The value is **NULL**.
grantee	character varying(128)	Not supported. The value is **NULL**.
audit_option	character varying(40)	Not supported. The value is **NULL**.
ses_actions	character varying(19)	Not supported. The value is **NULL**.
logoff_time	date	Not supported. The value is **NULL**.
logoff_lread	numeric	Not supported. The value is **NULL**.
logoff_pread	numeric	Not supported. The value is **NULL**.
logoff_lwrite	numeric	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
logoff_dlock	character varying(40)	Not supported. The value is **NULL**.
comment_text	character varying(4000)	Not supported. The value is **NULL**.
sessionid	numeric	Not supported. The value is **NULL**.
entryid	numeric	Not supported. The value is **NULL**.
statementid	numeric	Not supported. The value is **NULL**.
returncode	numeric	Not supported. The value is **NULL**.
priv_used	character varying(40)	Not supported. The value is **NULL**.
client_id	character varying(128)	Not supported. The value is **NULL**.
econtext_id	character varying(64)	Not supported. The value is **NULL**.
session_cpu	numeric	Not supported. The value is **NULL**.
extended_timestamp	timestamp(6) with time zone	Time zone of the timestamp when an audit trail entry is created (user login date and time of the entry created by the audit session in UTC).
proxy_sessionid	numeric	Not supported. The value is **NULL**.
global_uid	character varying(32)	Not supported. The value is **NULL**.
instance_number	numeric	Not supported. The value is **NULL**.
os_process	character varying(16)	Not supported. The value is **NULL**.
transactionid	text	Identifier of the transaction that accesses or modifies an object. **NOTE** The type of the **transactionid** column in GaussDB must be the same as that in database ORA.
scn	numeric	Not supported. The value is **NULL**.
sql_bind	nvarchar2(2000)	Not supported. The value is **NULL**.
sql_text	nvarchar2	SQL text of the query. **NOTE** The **sql_text** column in GaussDB is the parsed SQL description statement, which is not completely the same as the executed SQL statement.
obj_edition_name	character varying(128)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
dbid	numeric	Not supported. The value is **NULL**.
rls_info	clob	Not supported. The value is **NULL**.
current_user	character varying(128)	Not supported. The value is **NULL**.

7.3.7.5 GS_AUDITING

GS_AUDITING displays all audit information about database-related operations. By default, only users with the SYSADMIN or POLADMIN permission can access it. Common users can access it only after being authorized.

Table 7-246 GS_AUDITING columns

Name	Type	Description
polname	name	Policy name, which must be unique.
pol_type	text	Audit policy type. The value can be **'access'** or **'privilege'**. • **access**: DML operations are audited. • **privilege**: DDL operations are audited.
polenabled	boolean	Specifies whether the policy is enabled. • **t** (true): enabled. • **f** (false): disabled.
access_type	name	DML database operation types, such as SELECT, INSERT, and DELETE, or DDL database operation types, such as CREATE, ALTER, and DROP.
label_name	name	Resource label name. This column corresponds to the **polname** column in the **GS_AUDITING_POLICY** system catalog.
priv_object	text	Path of the database asset.
filter_name	text	Logical character string of a filter criterion.

7.3.7.6 GS_AUDITING_ACCESS

GS_AUDITING_ACCESS displays all audit information about database ACCESS-related operations. By default, only users with the SYSADMIN or POLADMIN permission can access it. Common users can access it only after being authorized.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-247 GS_AUDITING_ACCESS columns

Name	Type	Description
polname	name	Policy name, which must be unique.
pol_type	text	Audit policy type. The value is **'access'**, indicating that DML operations are audited.
polenabled	boolean	Specifies whether the policy is enabled. • **t** (true): enabled. • **f** (false): disabled.
access_type	name	ACCESS database operation type. Example: SELECT, INSERT, and DELETE.
label_name	name	Resource label name. This column corresponds to the **polname** column in the GS_AUDITING_POLICY system catalog.
access_object	text	Path of the database asset.
filter_name	text	Logical character string of a filter criterion.

7.3.7.7 GS_AUDITING_PRIVILEGE

GS_AUDITING_PRIVILEGE displays all audit information about database PRIVILEGE-related operations. By default, only users with the SYSADMIN or POLADMIN permission can access it. Common users can access it only after being authorized.

Table 7-248 GS_AUDITING_PRIVILEGE columns

Name	Type	Description
polname	name	Policy name, which must be unique.
pol_type	text	Audit policy type. The value is **'privilege'**, indicating that DDL operations are audited.
polenabled	boolean	Specifies whether to enable the policy. • **t** (true): enabled. • **f** (false): disabled.
access_type	name	PRIVILEGE database operation type. For example, CREATE, ALTER, and DROP.
label_name	name	Resource label name. This column corresponds to the **polname** column in the **GS_AUDITING_POLICY** system catalog.
priv_object	text	Full domain name of a database object.
filter_name	text	Logical character string of a filter criterion.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.8 User and Permission Management

7.3.8.1 ADM_COL_PRIVS

ADM_COL_PRIVS displays permission granting information about all columns. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-249 ADM_COL_PRIVS columns

Name	Type	Description
grantor	character varying(128)	Name of the user who grants the permission.
owner	character varying(128)	Object owner.
grantee	character varying(128)	Name of the user or role to which a permission is granted.
table_schema	character varying(128)	Schema of an object.
table_name	character varying(128)	Object name.
column_name	character varying(128)	Column name.
privilege	character varying(40)	Permission on a column.
grantable	character varying(3)	Specifies whether to grant privileges. • **YES** • **NO**
common	character varying(3)	Not supported. The value is **NULL**.
inherited	character varying(3)	Not supported. The value is **NULL**.

7.3.8.2 ADM_ROLE_PRIVS

ADM_ROLE_PRIVS displays information about roles granted to all users and roles. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-250 ADM_ROLE_PRIVS columns

Name	Type	Description
grantee	character varying(128)	Name of the user or role to which a permission is granted.
granted_role	character varying(128)	Role to be granted.
admin_option	character varying(3)	Specifies whether the grant contains the **ADMIN** option. • **YES** • **NO**
delegate_option	character varying(3)	Not supported. The value is **NULL**.
default_role	character varying(3)	Not supported. The value is **NULL**.
os_granted	character varying(3)	Not supported. The value is **NULL**.
common	character varying(3)	Not supported. The value is **NULL**.
inherited	character varying(3)	Not supported. The value is **NULL**.

7.3.8.3 ADM_ROLES

ADM_ROLES displays information about database roles. This view exists in both PG_CATALOG and SYS schemas. By default, only users with the SYSADMIN permission can access this view.

Table 7-251 ADM_ROLES columns

Name	Type	Description
role	character varying(128)	Role name.
role_id	oid	Role ID.
authentication_type	text	Authentication mechanism of the role. • **password**: Password authentication is required. • **null**: Authentication is not required.
common	character varying(3)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
oracle_maintained	character varying(1)	Not supported. The value is **NULL**.
inherited	character varying(3)	Not supported. The value is **NULL**.
implicit	character varying(3)	Not supported. The value is **NULL**.
external_name	character varying(4000)	Not supported. The value is **NULL**.

7.3.8.4 ADM_SYS_PRIVS

ADM_SYS_PRIVS displays information about system permissions granted to users and roles. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-252 ADM_SYS_PRIVS columns

Name	Type	Description
grantee	character varying(128)	Name of the user or role to which a permission is granted.
privilege	character varying(40)	System permissions or ANY permissions of the user. • System permissions include rolsuper, rolinherit, rolcreaterole, rolcreatedb, rolcatupdate, rolcanlogin, rolreplication, rolauditadmin, rolsystemadmin, roluseft, rolmonitoradmin, roloperatoradmin, and rolpolicyadmin. • For details about the value of the ANY permissions, see **Table 1-402**.
admin_option	character varying(3)	Specifies whether the grant contains the **ADMIN** option. • **YES** • **NO**
common	character varying(3)	Not supported. The value is **NULL**.
inherited	character varying(3)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.8.5 ADM_TAB_PRIVS

ADM_TAB_PRIVS displays authorization information about all objects in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-253 ADM_TAB_PRIVS columns

Name	Type	Description
grantee	character varying(128)	Name of the user or role to which a permission is granted.
owner	character varying(128)	Object owner.
table_name	character varying(128)	Object name.
grantor	character varying(128)	Name of the user who grants the permission.
privilege	character varying(40)	Permissions on an object, including USAGE, UPDATE, DELETE, INSERT, CONNECT, SELECT, and EXECUTE.
grantable	character varying(3)	Specifies whether the grant contains the **GRANT** option. • **YES** • **NO**
type	character varying(24)	Object type. Its value can be NODE GROUP, COLUMN_ENCRYPTION_KEY, PACKAGE, COLUMN, TABLE, VIEW, SEQUENCE, TYPE, INDEX, DATABASE, DIRECTORY, FOREIGN DATA WRAPPER, FOREIGN SERVER, LANGUAGE, SCHEMA, TEMPLATE, FUNCTION, PROCEDURE or TABLESPACE.
hierarchy	character varying(3)	Not supported. The value is **NULL**.
common	character varying(3)	Not supported. The value is **NULL**.
inherited	character varying(3)	Not supported. The value is **NULL**.

7.3.8.6 ADM_USERS

ADM_USERS displays information about all database users. This view is accessible only to users with the SYSADMIN permission. It exists in both PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-254 ADM_USERS columns

Name	Type	Description
username	character varying(128)	Username.
user_id	oid	User ID.
account_status	character varying(32)	Account status. • **NULL**: The account is the initial system administrator with the highest permission. • **0**: normal. • **1**: The account is locked for a specific period of time because the number of failed login attempts exceeds the threshold. • **2**: The account is locked by the administrator.
lock_date	timestamp with time zone	By default, the creation date of the account is displayed. If the account is locked by the administrator or the account is locked because the number of login failures exceeds the threshold, the date when the account is locked is displayed. The value is **NULL** for the initial system administrator.
expiry_date	timestamp with time zone	Account expiration date.
default_tablespace	character varying(4000)	Default tablespace for storing data.
temporary_tablespace	character varying(4000)	Name of the default tablespace or tablespace group of a temporary table.
local_temp_tablespace	character varying(30)	Not supported. The default value is **NULL**.
created	timestamp with time zone	Date when a user is created.
profile	character varying(128)	Not supported. The default value is **NULL**.
initial_rsrc_consumer_group	character varying(128)	Not supported. The default value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
external_name	character varying(4000)	Not supported. The default value is **NULL**.
password_versions	character varying(12)	Encryption mode of the account password. The value can be **MD5**, **SHA256**, or **SM3**.
editions_enabled	character varying(1)	Not supported. The default value is **NULL**.
authentication_type	text	Authentication mechanism of the user.
proxy_only_connect	character varying(1)	Not supported. The default value is **NULL**.
common	character varying(3)	Not supported. The default value is **NULL**.
last_login	timestamp with time zone	Last login.
oracle_maintained	character varying(1)	Not supported. The default value is **NULL**.
inherited	character varying(3)	Not supported. The default value is **NULL**.
default_collation	character varying(100)	Default collation of the user schema.
implicit	character varying(3)	Not supported. The default value is **NULL**.
all_shard	character varying(3)	Not supported. The default value is **NULL**.
password_change_date	timestamp with time zone	Date when a password was last set.

7.3.8.7 DB_COL_PRIVS

DB_COL_PRIVS displays the following granting information:

- Column permission granting information when the current user is the object owner, grantor, or grantee.
- Column permission granting information when the enabled role or PUBLIC role is the grantee.

By default, this view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-255 DB_COL_PRIVS columns

Name	Type	Description
grantor	character varying(128)	Name of the user who grants the permission.
owner	character varying(128)	Object owner.
grantee	character varying(128)	Name of the user or role to which a permission is granted.
table_schema	character varying(128)	Schema of an object.
table_name	character varying(128)	Object name.
column_name	character varying(128)	Column name.
privilege	character varying(40)	Permission on a column.
grantable	character varying(3)	Specifies whether to grant privileges. • **YES** • **NO**
common	character varying(3)	Not supported. The value is **NULL**.
inherited	character varying(3)	Not supported. The value is **NULL**.

7.3.8.8 DB_DIRECTORIES

DB_DIRECTORIES is used to view all directories on which the current user has the operation permission. Administrators can query all directories, and common users can query only directories on which the current user has the operation permission. This view exists in both PG_CATALOG and SYS schemas.

Table 7-256 DB_DIRECTORIES columns

Name	Type	Description
owner	oid	OID of the owner of a directory object.
directory_name	name	Name of a directory object.
directory_path	text	Directory path represented by a directory object.
origin_con_id	character varying(256)	ID of the container where a directory object is created. Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.8.9 DB_TAB_PRIVS

DB_TAB_PRIVS displays authorization information about all objects accessible to the current user. This view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-257 DB_TAB_PRIVS columns

Name	Type	Description
grantor	character varying(128)	Name of the user who grants the permission.
grantee	character varying(128)	Name of the user or role to which a permission is granted.
table_schema	character varying(128)	Schema of an object.
table_name	character varying(128)	Object name.
privilege	character varying(40)	Permissions on an object, including USAGE, UPDATE, DELETE, INSERT, CONNECT, SELECT, and EXECUTE.
grantable	character varying(3)	Specifies whether the grant contains the **GRANT** option. • **YES** • **NO**
type	character varying(24)	Object type. Its value can be NODE GROUP, COLUMN_ENCRYPTION_KEY, PACKAGE, COLUMN, TABLE, VIEW, SEQUENCE, TYPE, INDEX, DATABASE, DIRECTORY, FOREIGN DATA WRAPPER, FOREIGN SERVER, LANGUAGE, SCHEMA, TEMPLATE, FUNCTION, PROCEDURE, or TABLESPACE.
hierarchy	character varying(3)	Not supported. The value is **NULL**.
common	character varying(3)	Not supported. The value is **NULL**.
inherited	character varying(3)	Not supported. The value is **NULL**.

7.3.8.10 DB_USERS

DB_USERS displays all users of the database visible to the current user. However, it does not describe the users. By default, only users with the SYSADMIN permission can access this view. This view exists in both PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-258 DB_USERS columns

Name	Type	Description
user_id	oid	User OID.
username	name	Username.

7.3.8.11 GS_DB_PRIVILEGES

GS_DB_PRIVILEGES displays the granting of ANY permissions. Each record corresponds to a piece of authorization information.

Table 7-259 GS_DB_PRIVILEGES columns

Name	Type	Description
rolename	name	Username.
privilege_type	text	ANY permission of a user. For details about the value, see **Table 1-402**.
admin_option	text	Specifies whether the ANY permission recorded in the **privilege_type** column can be re-granted. • **yes** • **no**

7.3.8.12 GS_LABELS

GS_LABELS displays all configured resource labels. Only users with the SYSADMIN or POLADMIN permission can access this view.

Table 7-260 GS_LABELS columns

Name	Type	Description
labelname	name	Resource label name.
labeltype	name	Resource label type. This parameter corresponds to the **labeltype** column in the **GS_POLICY_LABEL** system catalog.
fqdntype	name	Database resource type. For example, table, schema, or index.
schemaname	name	Name of the schema to which a database resource belongs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
fqdnname	name	Database resource name.
columnname	name	Database resource column name. It is empty if the labeled database resource is not a table column.

7.3.8.13 MY_COL_PRIVS

MY_COL_PRIVS displays the column permission granting information of the current user as the object owner, grantor, or grantee. By default, this view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-261 MY_COL_PRIVS columns

Name	Type	Description
grantor	character varying(128)	Name of the user who grants the permission.
owner	character varying(128)	Object owner.
grantee	character varying(128)	Name of the user or role to which a permission is granted.
table_schema	character varying(128)	Schema of an object.
table_name	character varying(128)	Object name.
column_name	character varying(128)	Column name.
privilege	character varying(40)	Permission on a column.
grantable	character varying(3)	Specifies whether to grant privileges. • **YES** • **NO**
common	character varying(3)	Not supported. The value is **NULL**.
inherited	character varying(3)	Not supported. The value is **NULL**.

7.3.8.14 MY_ROLE_PRIVS

MY_ROLE_PRIVS displays permission information about roles (including the public role) granted to the current user. By default, this view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-262 MY_ROLE_PRIVS columns

Name	Type	Description
grantee	character varying(128)	Name of the user or role to which a permission is granted.
granted_role	character varying(128)	Role to be granted.
admin_option	character varying(3)	Specifies whether the grant contains the **ADMIN** option. • **YES** • **NO**
delegate_option	character varying(3)	Not supported. The value is **NULL**.
default_role	character varying(3)	Not supported. The value is **NULL**.
os_granted	character varying(3)	Not supported. The value is **NULL**.
common	character varying(3)	Not supported. The value is **NULL**.
inherited	character varying(3)	Not supported. The value is **NULL**.

7.3.8.15 MY_SYS_PRIVS

MY_SYS_PRIVS displays information about system permissions granted to the current user. By default, this view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-263 MY_SYS_PRIVS columns

Name	Type	Description
grantee	character varying(128)	Name of the user or role to which a permission is granted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
privilege	character varying(40)	System permissions or ANY permissions of the user. • System permissions include rolsuper, rolinherit, rolcreaterole, rolcreatedb, rolcatupdate, rolcanlogin, rolreplication, rolauditadmin, rolsystemadmin, roluseft, rolmonitoradmin, roloperatoradmin, and rolpolicyadmin. • For details about the value of the ANY permissions, see Table 1-402.
admin_option	character varying(3)	Specifies whether the grant contains the **ADMIN** option. • **YES** • **NO**
common	character varying(3)	Not supported. The value is **NULL**.
inherited	character varying(3)	Not supported. The value is **NULL**.

7.3.8.16 PG_RLSPOLICIES

PG_RLSPOLICIES displays information about row-level security policies. The initial user and users with the SYSADMIN attribute can view all policy information. Other users can view only the policy information in their own tables.

Table 7-264 PG_RLSPOLICIES columns

Name	Type	Description
schemaname	name	Name of the schema of the table object to which a row-level security policy is applied.
tablename	name	Name of the table object to which a row-level security policy is applied.
policyname	name	Name of a row-level security policy.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
policypermissive	text	Expression concatenation mode of a row-level security policy. Value range: • **PERMISSIVE**: permissive policy, which is concatenated using the OR expression. • **RESTRICTIVE**: restrictive policy, which is concatenated using the AND expression.
policyroles	name[]	List of users affected by the row-level security policy. If this parameter is not specified, all users are affected.
policycmd	text	SQL operations affected by a row-level security policy.
policyqual	text	Expression of a row-level security policy.

7.3.8.17 GS_RLSPOLICIES

GS_RLSPOLICIES displays information about row-level security policies. The initial user and users with the SYSADMIN attribute can view all policy information. Other users can view only the policy information in their own tables.

Table 7-265 GS_RLSPOLICIES columns

Name	Type	Description
schemaname	name	Name of the schema of the table object to which a row-level security policy is applied.
tablename	name	Name of the table object to which a row-level security policy is applied.
policyname	name	Name of a row-level security policy.
policypermissive	text	Expression concatenation mode of a row-level security policy. Value range: • **PERMISSIVE**: permissive policy, which is concatenated using the OR expression. • **RESTRICTIVE**: restrictive policy, which is concatenated using the AND expression.
policyroles	name[]	List of users affected by the row-level security policy. If this parameter is not specified, all users are affected.
policycmd	text	SQL operations affected by a row-level security policy.
policyqual	text	Expression of a row-level security policy.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.8.18 PG_ROLES

PG_ROLES displays information about database roles. The initial user and users with the SYSADMIN or CREATEROLE attribute can view information about all roles. Other users can view only their own information.

Table 7-266 PG_ROLES columns

Name	Type	Reference	Description
rolname	name	-	Role name.
rolsuper	boolean	-	Specifies whether a role is the initial SYSADMIN with the highest permissions. ● **t** (true): yes. ● **f** (false): no.
rolinherit	boolean	-	Specifies whether a role inherits the permissions for this type of roles. ● **t** (true): yes. ● **f** (false): no.
rolcreaterole	boolean	-	Specifies whether a role has the permissions to create other roles. ● **t** (true): yes. ● **f** (false): no.
rolcreatedb	boolean	-	Specifies whether a role has the permissions to create databases. ● **t** (true): yes. ● **f** (false): no.
rolcatupdate	boolean	-	Specifies whether a role has the permissions to directly update system catalogs. Only the initial SYSADMIN whose **usesysid** is set to **10** has this permission. This permission is unavailable for other users. ● **t** (true): yes. ● **f** (false): no.
rolcanlogin	boolean	-	Specifies whether a role has the permissions to log in to the database. ● **t** (true): yes. ● **f** (false): no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
rolreplication	boolean	-	Specifies whether a role has the permissions to be replicated. • **t** (true): yes. • **f** (false): no.
rolauditadmin	boolean	-	Specifies whether a role is AUDITADMIN. • **t** (true): yes. • **f** (false): no.
rolsystemadmin	boolean	-	Specifies whether a role is SYSADMIN. • **t** (true): yes. • **f** (false): no.
rolconnlimit	integer	-	Sets the maximum number of concurrent connections that this role can initiate if this role can log in. The value **-1** indicates no limit.
rolpassword	text	-	Encrypted user password. The value is displayed as ********.
rolvalidbegin	timestamp with time zone	-	Account validity start time. If this column is not set, the value is **NULL**.
rolvaliduntil	timestamp with time zone	-	Account validity end time. If this column is not set, the value is **NULL**.
rolrespool	name	-	Resource pool that a user can use.
rolparentid	oid	**PG_AUTHID**.rolparentid	OID of the group user to which a user belongs.
roltabspace	text	-	Storage space of the user permanent table, in KB.
rolconfig	text[]	**PG_DB_ROLE_SETTING**.setconfig	Default value of runtime configuration items.
oid	oid	**PG_AUTHID**.oid	Role ID.
roluseft	boolean	**PG_AUTHID**.roluseft	Specifies whether a role has the permissions to perform operations on foreign tables. • **t** (true): yes. • **f** (false): no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
rolkind	"char"	-	Role type. • **n**: common user, that is, non-permanent user. • **p**: permanent user.
nodegroup	name	-	Name of the node group associated with a role. If no node group is associated, leave it blank.
roltempspace	text	-	Storage space of the user temporary table, in KB.
rolspillspace	text	-	Operator disk spill space of the user, in KB.
rolmonitoradmin	boolean	-	Specifies whether a role is MONADMIN. • **t** (true): yes. • **f** (false): no.
roloperatoradmin	boolean	-	Specifies whether a role is OPRADMIN. • **t** (true): yes. • **f** (false): no.
rolpolicyadmin	boolean	-	Specifies whether a role is POLADMIN. • **t** (true): yes. • **f** (false): no.

7.3.8.19 GS_ROLES

GS_ROLES displays information about database roles. The initial user and users with the SYSADMIN or CREATEROLE attribute can view information about all roles. Other users can view only their own information.

Table 7-267 GS_ROLES columns

Name	Type	Reference	Description
rolname	name	-	Role name.
rolsuper	boolean	-	Specifies whether a role is the initial SYSADMIN with the highest permissions. • **t** (true): yes. • **f** (false): no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
rolinherit	boolean	-	Specifies whether a role inherits the permissions for this type of roles. ● **t** (true): yes. ● **f** (false): no.
rolcreaterole	boolean	-	Specifies whether a role has the permissions to create other roles. ● **t** (true): yes. ● **f** (false): no.
rolcreatedb	boolean	-	Specifies whether a role has the permissions to create databases. ● **t** (true): yes. ● **f** (false): no.
rolcatupdate	boolean	-	Specifies whether a role has the permissions to directly update system catalogs. Only the initial SYSADMIN whose **usesysid** is set to **10** has this permission. This permission is unavailable for other users. ● **t** (true): yes. ● **f** (false): no.
rolcanlogin	boolean	-	Specifies whether a role has the permissions to log in to the database. ● **t** (true): yes. ● **f** (false): no.
rolreplication	boolean	-	Specifies whether a role has the permissions to be replicated. ● **t** (true): yes. ● **f** (false): no.
rolauditadmin	boolean	-	Specifies whether a role is AUDITADMIN. ● **t** (true): yes. ● **f** (false): no.
rolsystemadmin	boolean	-	Specifies whether a role is SYSADMIN. ● **t** (true): yes. ● **f** (false): no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
rolconnlimit	integer	-	Sets the maximum number of concurrent connections that this role can initiate if this role can log in. The value **-1** indicates no limit.
rolpassword	text	-	Encrypted user password. The value is displayed as ********.
rolvalidbegin	timestamp with time zone	-	Account validity start time. If this column is not set, the value is **NULL**.
rolvaliduntil	timestamp with time zone	-	Account validity end time. If this column is not set, the value is **NULL**.
rolrespool	name	-	Resource pool that a user can use.
rolparentid	oid	**PG_AUTHID**.rolparentid	OID of the group user to which a user belongs.
roltabspace	text	-	Storage space of the user permanent table, in KB.
rolconfig	text[]	**PG_DB_ROLE_SETTING**.setconfig	Default value of runtime configuration items.
oid	oid	**PG_AUTHID**.oid	Role ID.
roluseft	boolean	**PG_AUTHID**.roluseft	Specifies whether a role has the permissions to perform operations on foreign tables. • **t** (true): yes. • **f** (false): no.
rolkind	"char"	-	Role type. • **n**: common user, that is, non-permanent user. • **p**: permanent user.
nodegroup	name	-	Name of the node group associated with a role. If no node group is associated, it is left blank.
roltempspace	text	-	Storage space of the user temporary table, in KB.
rolspillspace	text	-	Operator disk spill space of the user, in KB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
rolmonitoradmin	boolean	-	Specifies whether a role is MONADMIN. • **t** (true): yes. • **f** (false): no.
roloperatoradmin	boolean	-	Specifies whether a role is OPRADMIN. • **t** (true): yes. • **f** (false): no.
rolpolicyadmin	boolean	-	Specifies whether a role is POLADMIN. • **t** (true): yes. • **f** (false): no.

7.3.8.20 PG_SECLABELS

PG_SECLABELS provides information about security labels.

Table 7-268 PG_SECLABELS columns

Name	Type	Reference	Description
objoid	oid	Any OID column	OID of the object to which a security label belongs.
classoid	oid	PG_CLASS.oid	OID of the system catalog that contains the object to which a security label belongs.
objsubid	integer	-	• Column number for the security label on a table column (by referring to **objoid** and **classoid** in the table). • For all other object types, this column is **0**.
objtype	text	-	Type of the object to which a security label belongs, in text format. Example: • **table**: table type. • **column**: column type.
objnamespace	oid	PG_NAMESPACE.oid	OID of the namespace for the object, if applicable; otherwise, **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
objname	text	-	Name of the object to which a security label belongs, in text format.
provider	text	PG_SECLABEL.provider	Security label provider.
label	text	PG_SECLABEL.label	Security label name.

7.3.8.21 GS_SECLABELS

GS_SECLABELS provides information about security labels.

Table 7-269 GS_SECLABELS columns

Name	Type	Reference	Description
objoid	oid	Any OID column	OID of the object to which a security label belongs.
classoid	oid	PG_CLASS.oid	OID of the system catalog that contains the object to which a security label belongs.
objsubid	integer	-	• Column number for the security label on a table column (by referring to **objoid** and **classoid** in the table). • For all other object types, this column is **0**.
objtype	text	-	Type of the object to which a security label belongs, in text format. • **table**: table type. • **column**: column type.
objnamespace	oid	PG_NAMESPACE.oid	OID of the namespace for the object, if applicable; otherwise, **NULL**.
objname	text	-	Name of the object to which a security label belongs, in text format.
provider	text	PG_SECLABEL.provider	Security label provider.
label	text	PG_SECLABEL.label	Security label name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.8.22 PG_SHADOW

PG_SHADOW displays the attributes of all roles marked with rolcanlogin in **PG_AUTHID**. Only the users with the SYSADMIN permission can access this system view.

The information in this view is basically the same as that of the view in **PG_USER**. The difference is that passwords are sensitive and displayed as ******** in the latter.

Table 7-270 PG_SHADOW columns

Name	Type	Reference	Description
usename	name	**PG_AUTHID**.rolname	Username.
usesysid	oid	**PG_AUTHID**.oid	User ID.
usecreatedb	boolean	-	Specifies whether a user has the permissions to create databases. • **t** (true): yes. • **f** (false): no.
usesuper	boolean	-	Specifies whether a user is SYSADMIN. • **t** (true): yes. • **f** (false): no.
usecatupd	boolean	-	Specifies whether a user has the permissions to update views. Even users with the SYSADMIN permission cannot update the view unless this column is **t**. • **t** (true): yes. • **f** (false): no.
userepl	boolean	-	Specifies whether a user has the permissions to duplicate data streams. • **t** (true): yes. • **f** (false): no.
passwd	text	**PG_AUTHID**.rolpassword	Password ciphertext. If there is no password, the value is **NULL**.
valbegin	timestamp with time zone	-	Account validity start time. If this column is not set, the value is **NULL**.
valuntil	timestamp with time zone	-	Account validity end time. If this column is not set, the value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
respool	name	-	Resource pool where the user is located.
parent	oid	-	Parent user OID.
spacelimit	text	-	Storage space of the permanent table, in KB.
useconfig	text[]	PG_DB_ROLE_SETTING.setconfig	Default value of runtime configuration items.
tempspacelimit	text	-	Storage space of the temporary table, in KB.
spillspacelimit	text	-	Operator disk flushing space, in KB.
usemonitoradmin	boolean	-	Specifies whether a user is MONADMIN. • **t** (true): yes. • **f** (false): no.
useoperatoradmin	boolean	-	Specifies whether a user is OPRADMIN. • **t** (true): yes. • **f** (false): no.
usepolicyadmin	boolean	-	Specifies whether a user is POLADMIN. • **t** (true): yes. • **f** (false): no.

7.3.8.23 GS_SHADOW

GS_SHADOW displays the attributes of all roles marked with rolcanlogin in PG_AUTHID. Only the users with the SYSADMIN permission can access this system view.

The information in this view is basically the same as that of the view in GS_USER. The difference is that passwords are sensitive and displayed as ******** in the latter.

Table 7-271 GS_SHADOW columns

Name	Type	Reference	Description
usename	name	PG_AUTHID.rolname	Username.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
usesysid	oid	PG_AUTHID.oid	User ID.
usecreatedb	boolean	-	Specifies whether a user has the permissions to create databases. • **t** (true): yes. • **f** (false): no.
usesuper	boolean	-	Specifies whether a user is SYSADMIN. • **t** (true): yes. • **f** (false): no.
usecatupd	boolean	-	Specifies whether a user has the permissions to update views. Even users with the SYSADMIN permission cannot update the view unless this column is **t**. • **t** (true): yes. • **f** (false): no.
userepl	boolean	-	Specifies whether a user has the permissions to duplicate data streams. • **t** (true): yes. • **f** (false): no.
passwd	text	PG_AUTHID.rolpassword	Password ciphertext. If there is no password, the value is **NULL**.
valbegin	timestamp with time zone	-	Account validity start time. If this column is not set, the value is **NULL**.
valuntil	timestamp with time zone	-	Account validity end time. If this column is not set, the value is **NULL**.
respool	name	-	Resource pool where the user is located.
parent	oid	-	Parent user OID.
spacelimit	text	-	Storage space of the permanent table, in KB.
useconfig	text[]	PG_DB_ROLE_SETTING.setconfig	Default value of runtime configuration items.
tempspacelimit	text	-	Storage space of the temporary table, in KB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
spillspacelimit	text	-	Operator disk flushing space, in KB.
usemonitoradmin	boolean	-	Specifies whether a user is MONADMIN. • **t** (true): yes. • **f** (false): no.
useoperatoradmin	boolean	-	Specifies whether a user is OPRADMIN. • **t** (true): yes. • **f** (false): no.
usepolicyadmin	boolean	-	Specifies whether a user is POLADMIN. • **t** (true): yes. • **f** (false): no.

7.3.8.24 PG_USER

PG_USER displays information about database users. By default, only the initial user and users with the SYSADMIN attribute can view the information. Other users can view the information only after being granted permissions.

Table 7-272 PG_USER columns

Name	Type	Description
usename	name	Username.
usesysid	oid	User ID.
usecreatedb	boolean	Specifies whether a user has the permissions to create databases. • **t** (true): yes. • **f** (false): no.
usesuper	boolean	Specifies whether a user is the initial system administrator with the highest permissions. • **t** (true): yes. • **f** (false): no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
usecatupd	boolean	Specifies whether a user has the permissions to directly update system catalogs. Only the initial SYSADMIN whose **usesysid** is set to **10** has this permission. This permission is unavailable for other users. • **t** (true): yes. • **f** (false): no.
userepl	boolean	Specifies whether a user has the permissions to duplicate data streams. • **t** (true): yes. • **f** (false): no.
passwd	text	Encrypted user password. The value is displayed as ********.
valbegin	timestamp with time zone	Account validity start time. If this column is not set, the value is **NULL**.
valuntil	timestamp with time zone	Account validity end time. If this column is not set, the value is **NULL**.
respool	name	Resource pool where the user is located.
parent	oid	Parent user OID.
spacelimit	text	Storage space of the permanent table, in KB.
useconfig	text[]	Default value of runtime configuration items. For details, see **setconfig** in **PG_DB_ROLE_SETTING**.
nodegroup	name	Name of the node group associated with a user. If the user is not associated with any node groups, it is left blank.
tempspacelimit	text	Storage space of the temporary table, in KB.
spillspacelimit	text	Operator disk flushing space, in KB.
usemonitoradmin	boolean	Specifies whether a user is MONADMIN. • **t** (true): yes. • **f** (false): no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
useoperatoradmin	boolean	Specifies whether a user is OPRADMIN. ● **t** (true): yes. ● **f** (false): no.
usepolicyadmin	boolean	Specifies whether this user is POLADMIN. ● **t** (true): yes. ● **f** (false): no.

7.3.8.25 GS_USER

GS_USER displays information about database users. By default, only the initial user and users with the SYSADMIN attribute can view the information. Other users can view the information only after being granted permissions.

Table 7-273 GS_USER columns

Name	Type	Description
usename	name	Username.
usesysid	oid	User ID.
usecreatedb	boolean	Specifies whether a user has the permissions to create databases. ● **t** (true): yes. ● **f** (false): no.
usesuper	boolean	Specifies whether a user is the initial system administrator with the highest permissions. ● **t** (true): yes. ● **f** (false): no.
usecatupd	boolean	Specifies whether a user has the permissions to directly update system catalogs. Only the initial SYSADMIN whose **usesysid** is set to **10** has this permission. This permission is unavailable for other users. ● **t** (true): yes. ● **f** (false): no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
userepl	boolean	Specifies whether a user has the permissions to duplicate data streams. • **t** (true): yes. • **f** (false): no.
passwd	text	Encrypted user password. The value is displayed as ********.
valbegin	timestamp with time zone	Account validity start time. If this column is not set, the value is **NULL**.
valuntil	timestamp with time zone	Account validity end time. If this column is not set, the value is **NULL**.
respool	name	Resource pool where the user is located.
parent	oid	Parent user OID.
spacelimit	text	Storage space of the permanent table, in KB.
useconfig	text[]	Default value of runtime configuration items. For details, see **setconfig** in **PG_DB_ROLE_SETTING**.
nodegroup	name	Name of the node group associated with a user. If the user is not associated with any node groups, it is left blank.
tempspacelimit	text	Storage space of the temporary table, in KB.
spillspacelimit	text	Operator disk flushing space, in KB.
usemonitoradmin	boolean	Specifies whether a user is MONADMIN. • **t** (true): yes. • **f** (false): no.
useoperatoradmin	boolean	Specifies whether a user is OPRADMIN. • **t** (true): yes. • **f** (false): no.
usepolicyadmin	boolean	Specifies whether a user is POLADMIN. • **t** (true): yes. • **f** (false): no.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.8.26 PG_USER_MAPPINGS

PG_USER_MAPPINGS displays user mapping information. All users can view the report.

Table 7-274 PG_USER_MAPPINGS columns

Name	Type	Reference	Description
umid	oid	PG_USER_MAPPING.oid	OID of the user mapping.
srvid	oid	PG_FOREIGN_SERVER.oid	OID of the foreign server that contains the mapping.
srvname	name	PG_FOREIGN_SERVER.srvname	Name of the foreign server.
umuser	oid	PG_AUTHID.oid	OID of the local role being mapped (**0** if the user mapping is public).
usename	name	-	Name of the local user to be mapped.
umoptions	text[]	-	User mapping specific options. If the current user is the owner of the foreign server, it is a string in the format of "keyword=value". Otherwise, it is **NULL**.

7.3.8.27 ROLE_ROLE_PRIVS

ROLE_ROLE_PRIVS displays roles granted to other roles and provides only information about the roles accessible to the user. By default, this view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-275 ROLE_ROLE_PRIVS columns

Name	Type	Description
role	character varying(128)	Role name.
granted_role	character varying(128)	Role to be granted.
admin_option	character varying(3)	Specifies whether the grant contains the **ADMIN** option. • **YES** • **NO**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
common	character varying(3)	Not supported. The value is **NULL**.
inherited	character varying(3)	Not supported. The value is **NULL**.

7.3.8.28 ROLE_SYS_PRIVS

ROLE_SYS_PRIVS displays information about system privileges granted to roles (only roles accessible to the user are displayed). By default, this view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-276 ROLE_SYS_PRIVS columns

Name	Type	Description
role	character varying(128)	Role name.
privilege	character varying(40)	System permissions or ANY permissions of the user. • System permissions include rolsuper, rolinherit, rolcreaterole, rolcreatedb, rolcatupdate, rolcanlogin, rolreplication, rolauditadmin, rolsystemadmin, roluseft, rolmonitoradmin, roloperatoradmin, and rolpolicyadmin. • For details about the value of the ANY permissions, see **Table 1-402**.
admin_option	character varying(3)	Specifies whether the grant contains the **ADMIN** option. • **YES** • **NO**
common	character varying(3)	Not supported. The value is **NULL**.
inherited	character varying(3)	Not supported. The value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.8.29 ROLE_TAB_PRIVS

ROLE_TAB_PRIVS displays information about object permissions granted to roles (only roles accessible to the user are displayed). By default, this view is accessible to all users. It exists in both PG_CATALOG and SYS schemas.

Table 7-277 ROLE_TAB_PRIVS columns

Name	Type	Description
role	character varying(128)	Role name.
owner	character varying(128)	Object owner.
table_name	character varying(128)	Object name. Object types include tables, packages, indexes, and sequences.
column_name	character varying(128)	Not supported. The value is **NULL**.
privilege	character varying(40)	Permissions on an object, including USAGE, UPDATE, DELETE, INSERT, CONNECT, SELECT, and EXECUTE.
grantable	character varying(3)	Specifies whether the grant contains the **GRANT** option. • **YES** • **NO**
common	character varying(3)	Not supported. The value is **NULL**.
inherited	character varying(3)	Not supported. The value is **NULL**.

7.3.9 Dynamic Data Masking

7.3.9.1 GS_MASKING

GS_MASKING displays all configured dynamic masking policies. Only the users with the SYSADMIN or POLADMIN permission can access this view.

Table 7-278 GS_MASKING columns

Name	Type	Description
polname	name	Name of the masking policy
polenabled	Boolean	Specifies whether to enable the masking policy.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
maskaction	name	Masking function
labelname	name	Name of the label to which the masking function applies.
masking_object	text	Masking database resource object
filter_name	text	Logical expression of a filter criterion

7.3.10 Transparent Data Encryption (TDE)

7.3.10.1 PG_TDE_INFO

PG_TDE_INFO displays encryption information of the entire cluster.

Table 7-279 PG_TDE_INFO columns

Name	Type	Description
is_encrypt	Boolean	Specifies whether the cluster is an encrypted cluster. • **f**: Non-encrypted cluster. • **t**: Encrypted cluster.
g_tde_algo	text	Encryption algorithm. • **SM4-CTR-128** • **AES-CTR-128**
remain	text	Reserved column.

7.3.11 DATABASE LINK

7.3.11.1 GS_DB_LINKS

GS_DB_LINKS displays information about database links. You can view information about your own database links and database links at the PUBLIC level.

Table 7-280 GS_DB_LINKS columns

Name	Type	Description
dblinkid	oid	OID of the current database link.
dlname	name	Name of the current DATABASE LINK object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
dlowner	oid	Owner ID of the current database link. If the object owner is **public**, the value is **0**.
dlownername	name	Name of the owner of the current database link.
options	text[]	Connection information of the current database link. The value is a character string in the "keyword=value" format.
useroptions	text	Information of the remote user to which the current database link connects.
heterogeneous	text	Not supported yet. Its value is **NULL**.
protocol	text	Not supported yet. Its value is **NULL**.
opencursors	text	Not supported yet. Its value is **NULL**.
intransaction	Boolean	Specifies whether the current database link exists in a transaction.
updatasent	Boolean	Specifies whether the current database link uses statements for updating data in a transaction and is in the uncommitted phase.

7.3.11.2 V$DBLINK

V$DBLINK displays information about database links. You can view information about your own database links and database links at the PUBLIC level. By default, only administrators can view this view.

NOTE

The data in this view is obtained from **GS_DB_LINKS**, but this view only records the information about database links that have been used or are being used in the current session. For details about how to use database links, see "Feature Usage Guidelines > Database Links" in *Developer Guide*.

Table 7-281 V$DBLINK columns

Name	Type	Description
db_link	character varying(128)	Name of the current database link.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
owner_id	numeric	Owner ID of the current database link. If the owner of the database link is **PUBLIC**, the value of **owner_id** is **0**.
logged_on	character varying(3)	Not supported yet. Its value is **NULL**.
heterogeneous	character varying(3)	Not supported yet. Its value is **NULL**.
protocol	character varying(6)	Not supported yet. Its value is **NULL**.
open_cursors	numeric	Not supported yet. Its value is **NULL**.
in_transaction	character varying(3)	Determines whether the current database link exists in a transaction. ● **YES**: in a transaction. ● **NO**: not in a transaction.
update_sent	character varying(3)	Determines whether the current database link uses statements for updating data. ● **YES**: Statements for updating data are used. ● **NO**: Statements for updating data are not used.
commit_point_strength	numeric	Not supported yet. Its value is **NULL**.
con_id	numeric	Not supported yet. Its value is **NULL**.

7.3.12 Materialized Views

7.3.12.1 GS_MATVIEWS

GS_MATVIEWS displays information about each materialized view in the database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-282 GS_MATVIEWS columns

Name	Type	Reference	Description
schemaname	name	**nspname** in PG_NAMESPACE	Name of the schema of a materialized view.
matviewname	name	**relname** in PG_CLASS	Name of a materialized view.
matviewowner	name	**rolname** in PG_AUTHID	Owner of a materialized view.
tablespace	name	**spcname** in PG_TABLESPACE	Tablespace name of a materialized view. If the default tablespace of the database is used, the value is null.
hasindexes	Boolean	-	This column is true if a materialized view has (or has recently had) any indexes.
definition	text	-	Definition of a materialized view (a reconstructed SELECT query)

7.3.13 Other System Views

NOTE

Accessing a view in the SYS schema requires access permissions. Common users can be granted access permissions after the administrator run the **GRANT USAGE ON SCHEMA SYS TO** *username*; command.

7.3.13.1 ADM_ARGUMENTS

ADM_ARGUMENTS displays parameter information of all stored procedures and functions. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-283 ADM_ARGUMENTS columns

Name	Type	Description
owner	character varying(128)	Owner of a function or stored procedure.
object_name	character varying(128)	Name of a function or stored procedure.
package_name	character varying(128)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
object_id	oid	OID of a function or stored procedure.
overload	character varying(40)	The *n*th overloaded function of the same name.
subprogram_id	numeric	Not supported yet. Its value is **NULL**.
argument_name	character varying(128)	Parameter name.
position	numeric	Position of a parameter in the parameter list. The value is **0** for the return value of the function by default.
sequence	numeric	Sequence of a parameter, which starts from 1, with the return type before all parameters.
data_level	numeric	Nesting depth of parameters of the composite type. The value of this column is always **0** because only one row is displayed for each parameter.
data_type	character varying(64)	Data type of a parameter.
defaulted	character varying(1)	Specifies whether a parameter has a default value: • **y**: yes. • **n**: no.
default_value	text	Not supported yet. Its value is **NULL**.
default_length	numeric	Not supported yet. Its value is **NULL**.
in_out	character varying(9)	Input and output attributes of a parameter: • **in**: input parameter. • **out**: output parameter. • **in/out**: input and output parameters. • **variadic**: VARIADIC parameter.
data_length	numeric	Not supported yet. Its value is **NULL**.
data_precision	numeric	Not supported yet. Its value is **NULL**.
data_scale	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
radix	numeric	Radix of a number, which is **10** when the data type is smallint, integer, bigint, numeric, or float. For other data types, set this column to **NULL**.
character_set_name	character varying(44)	Not supported yet. Its value is **NULL**.
type_owner	character varying(128)	Owner of a data type.
type_name	character varying(128)	Parameter type name. Only the user-defined type is displayed.
type_subname	character varying(128)	Not supported yet. Its value is **NULL**.
type_link	character varying(128)	Not supported yet. Its value is **NULL**.
type_object_type	character varying(7)	The parameter type of the data type described by the **type_owner**, **type_name**, and **type_subname** columns: • **table**: The parameter is of the table type. • **view**: The parameter is of the view type. • For other data types, the value is **NULL**.
pls_type	character varying(128)	Name of the PL/SQL type for parameters of the number type. Otherwise, this column is empty.
char_length	numeric	Not supported yet. Its value is **NULL**.
char_used	character varying(1)	Not supported. • **b**: varchar, nvarchar2, bpchar, or char. • **NULL**: other types.
origin_con_id	character varying(256)	Not supported. The value is **0**.

7.3.13.2 ADM_COL_COMMENTS

ADM_COL_COMMENTS displays information about table column comments in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-284 ADM_COL_COMMENTS columns

Name	Type	Description
owner	character varying(128)	Table owner.
table_name	character varying(128)	Table name.
column_name	character varying(128)	Column name.
comments	text	Comments.
origin_con_id	numeric	Not supported. The value is **0**.
schema	character varying(64)	Name of the namespace to which the column belongs.

7.3.13.3 ADM_COLL_TYPES

ADM_COLL_TYPES displays information about all collection types. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-285 ADM_COLL_TYPES columns

Name	Type	Description
owner	character varying(128)	Owner of a collection.
type_name	character varying(128)	Name of a collection.
coll_type	character varying(128)	Description of a collection.
upper_bound	numeric	Not supported yet. Its value is **NULL**.
elem_type_mod	character varying(7)	Not supported yet. Its value is **NULL**.
elem_type_owner	character varying(128)	Owner of the element type on which the collection is based. This parameter is mainly used for user-defined types.
elem_type_name	character varying(128)	Name of the data type or user-defined type on which the collection is based.
length	numeric	Not supported yet. Its value is **NULL**.
precision	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
scale	numeric	Not supported yet. Its value is **NULL**.
character_set_name	character varying(44)	Not supported yet. Its value is **NULL**.
elem_storage	character varying(7)	Not supported yet. Its value is **NULL**.
nulls_stored	character varying(3)	Not supported yet. Its value is **NULL**.
char_used	character varying(1)	Not supported yet. Its value is **NULL**.

7.3.13.4 ADM_CONS_COLUMNS

ADM_CONS_COLUMNS displays information about constraint columns in database tables. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-286 ADM_CONS_COLUMNS columns

Name	Type	Description
owner	character varying(64)	Constraint creator.
constraint_name	character varying(64)	Constraint name.
table_name	character varying(64)	Name of a constraint-related table.
column_name	character varying(64)	Name of a constraint-related column.
position	smallint	Position of a column in a table.

7.3.13.5 ADM_CONSTRAINTS

ADM_CONSTRAINTS displays information about table constraints in database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-287 ADM_CONSTRAINTS columns

Name	Type	Description
constraint_name	character varying(64)	Constraint name.
constraint_type	text	Constraint type. • **c**: CHECK constraint. • **f**: FOREIGN KEY constraint. • **p**: PRIMARY KEY constraint. • **u**: UNIQUE constraint.
table_name	character varying(64)	Name of a constraint-related table.
index_owner	character varying(64)	Owner of a constraint-related index (only for the UNIQUE constraint and PRIMARY KEY constraint).
index_name	character varying(64)	Name of a constraint-related index (only for the UNIQUE constraint and PRIMARY KEY constraint).
owner	character varying(64)	Constraint creator.
status	character varying(8)	Constraint status.
generated	character varying(14)	Not supported yet. Its value is **NULL**.
search_condition	text	Not supported yet. Its value is **NULL**.
search_condition_vc	character varying(4000)	Not supported yet. Its value is **NULL**.
r_owner	character varying(128)	Not supported yet. Its value is **NULL**.
r_constraint_name	character varying(128)	Not supported yet. Its value is **NULL**.
delete_rule	character varying(9)	Not supported yet. Its value is **NULL**.
con_deferrable	character varying(14)	Not supported yet. Its value is **NULL**.
deferred	character varying(9)	Not supported yet. Its value is **NULL**.
validated	character varying(13)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
bad	character varying(3)	Not supported yet. Its value is **NULL**.
rely	character varying(4)	Not supported yet. Its value is **NULL**.
last_change	date	Not supported yet. Its value is **NULL**.
invalid	character varying(7)	Not supported yet. Its value is **NULL**.
view_related	character varying(14)	Not supported yet. Its value is **NULL**.
origin_con_id	character varying(256)	Not supported yet. Its value is **NULL**.

7.3.13.6 ADM_DATA_FILES

ADM_DATA_FILES displays the description of database files. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-288 ADM_DATA_FILES columns

Name	Type	Description
tablespace_name	name	Name of the tablespace to which a file belongs
bytes	double precision	Length of a file, in bytes

7.3.13.7 ADM_DEPENDENCIES

ADM_DEPENDENCIES displays the dependency relationships between types, tables, views, stored procedures, functions, and triggers in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-289 ADM_DEPENDENCIES columns

Name	Type	Description
owner	name	Object owner.
name	name	Object name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
type	character varying(18)	Object type.
referenced_owner	name	Owner of the referenced object.
referenced_name	name	Name of the referenced object.
referenced_type	character varying(18)	Type of the referenced object.
referenced_link_name	character varying(128)	Not supported yet. Its value is **NULL**.
dependency_type	character varying(4)	Not supported yet. Its value is **NULL**.

7.3.13.8 ADM_DIRECTORIES

ADM_DIRECTORIES stores information about all directory objects in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-290 ADM_DIRECTORIES columns

Name	Type	Description
owner	oid	Directory owner.
directory_name	name	Directory name.
directory_path	text	OS path name of a directory.
origin_con_id	character varying(256)	Not supported yet. Its value is **NULL**.

7.3.13.9 ADM_HIST_SNAPSHOT

ADM_HIST_SNAPSHOT records the WDR snapshot data stored in the current system. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas. This view can be accessed only when the GUC parameter **enable_wdr_snapshot** is set to **on**. To access the PG_CATALOG.ADM_HIST_SNAPSHOT and SYS.ADM_HIST_SNAPSHOT views, you must have the permission to access the **snapshot schema**, **snapshot**, and **tables_snap_timestamp** tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-291 ADM_HIST_SNAPSHOT columns

Name	Type	Description
snap_id	bigint	Unique snapshot ID.
dbid	oid	Database ID of the snapshot.
instance_number	oid	Not supported. The value is the same as that of **DBID**.
startup_time	timestamp(3) without time zone	Instance start time.
begin_interval_time	timestamp without time zone	Start time of the snapshot interval (the end time of the last snapshot).
end_interval_time	timestamp without time zone	End time of the snapshot interval. Actual time when the snapshot is taken (that is, the end time of the snapshot).
flush_elapsed	interval	Amount of time to perform the snapshot.
snap_level	numeric	Not supported yet. Its value is **NULL**.
error_count	numeric	Not supported yet. Its value is **NULL**.
snap_flag	numeric	Not supported yet. Its value is **NULL**.
snap_timezone	interval day to second(0)	Time offset between the snapshot time zone and the coordinated universal time (UTC).
begin_interval_time_tz	timestamp with time zone	Start time of the snapshot interval with the time zone (end time of the last snapshot).
end_interval_time_tz	timestamp with time zone	End time of the snapshot interval. Actual time when the snapshot is taken (that is, the end time of the snapshot), with the time zone.
con_id	numeric	Not supported. The value is **0**.

7.3.13.10 ADM_HIST_SQL_PLAN

ADM_HIST_SQL_PLAN displays plan information collected by the current user by running the EXPLAIN PLAN statement. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-292 ADM_HIST_SQL_PLAN columns

Name	Type	Description
dbid	text	Database ID
sql_id	character varying(30)	Session that inserts the data. Its value consists of a worker thread start timestamp and a worker thread ID. Values are constrained by **NOT NULL**.
plan_hash_value	bigint	Query ID.
id	integer	Number assigned to each step in the execution plan
operation	character varying(30)	Operation description
options	character varying(255)	Operation action
object_node	character varying(128)	Not supported yet. Its value is **NULL**.
object#	numeric	Not supported yet. Its value is **NULL**.
object_owner	name	Object number of a table or an index
object_name	name	Name of an operated object. It is defined by users.
object_alias	character varying(261)	Not supported yet. Its value is **NULL**.
object_type	character varying(30)	Object type
optimizer	character varying(20)	Not supported yet. Its value is **NULL**.
parent_id	numeric	Not supported yet. Its value is **NULL**.
depth	numeric	Not supported yet. Its value is **NULL**.
position	numeric	Not supported yet. Its value is **NULL**.
search_columns	numeric	Not supported yet. Its value is **NULL**.
cost	double precision	Execution cost estimated by the optimizer for an operator
cardinality	double precision	Cardinality estimated by the optimizer for an operator to access table records.
bytes	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
other_tag	character varying(35)	Not supported yet. Its value is **NULL**.
partition_start	character varying(64)	Not supported yet. Its value is **NULL**.
partition_stop	character varying(64)	Not supported yet. Its value is **NULL**.
partition_id	numeric	Not supported yet. Its value is **NULL**.
other	character varying(4000)	Not supported yet. Its value is **NULL**.
distribution	character varying(20)	Not supported yet. Its value is **NULL**.
cpu_cost	numeric	Not supported yet. Its value is **NULL**.
io_cost	numeric	Not supported yet. Its value is **NULL**.
temp_space	numeric	Not supported yet. Its value is **NULL**.
access_predicates	character varying(4000)	Not supported yet. Its value is **NULL**.
filter_predicates	character varying(4000)	Not supported yet. Its value is **NULL**.
projection	character varying(4000)	Returned column information
time	numeric	Not supported yet. Its value is **NULL**.
qblock_name	character varying(128)	Not supported yet. Its value is **NULL**.
remarks	character varying(4000)	Not supported yet. Its value is **NULL**.
timestamp	date	Not supported yet. Its value is **NULL**.
other_xml	clob	Not supported yet. Its value is **NULL**.
con_dbid	text	Database ID of a container, which is currently set to a value the same as that of **dbid**.
con_id	numeric	Container ID. Containers are not supported. Its value is **0**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.11 ADM_HIST_SQLSTAT

ADM_HIST_SQLSTAT displays information about statements executed on the current node. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

After the WDR snapshot function is enabled (that is, the GUC parameter **enable_wdr_snapshot** is set to **on**), users can view the data in this view.

Table 7-293 ADM_HIST_SQLSTAT columns

Name	Type	Description
instance_number	integer	Instance ID of a snapshot.
plan_hash_value	integer	ID of the normalized SQL statement.
module	integer	Name of the module that is executing when the SQL statement is first parsed.
apwait_delta	integer	Delta value of the application wait time.
sql_id	bigint	Query ID.
snap_id	bigint	Unique snapshot ID.
elapsed_time_delta	bigint	Valid DB time, which is accumulated if multiple threads are involved (unit: μs).
cpu_time_delta	bigint	CPU time (unit: μs).
executions_delta	bigint	Increment in the number of executions that have occurred on this object since it was brought into the cache.
iowait_delta	bigint	I/O time (unit: μs).
rows_processed_delta	bigint	Number of rows in the result set returned by the SELECT statement.
parsing_schema_name	character varying	Not supported yet. Its value is **NULL**.
disk_reads_delta	bigint	Not supported yet. Its value is **NULL**.
buffer_reads_delta	bigint	Not supported yet. Its value is **NULL**.
clwait_delta	bigint	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.12 ADM_HIST_SQLTEXT

ADM_HIST_SQLTEXT displays information about statements executed on the current node. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

After the WDR snapshot function is enabled (that is, the GUC parameter **enable_wdr_snapshot** is set to **on**), users can view the data in this view.

Table 7-294 ADM_HIST_SQLTEXT columns

Name	Type	Description
dbid	integer	Database ID.
sql_id	bigint	Query ID.
sql_text	clob	Text corresponding to the query.
command_type	integer	Not supported. The value is **0**.
con_dbid	integer	Database ID of a container, which is currently set to a value the same as that of **dbid**.
con_id	integer	Container ID. Containers are not supported. The value is **0**.

7.3.13.13 ADM_IND_COLUMNS

ADM_IND_COLUMNS displays information about index columns in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-295 ADM_IND_COLUMNS columns

Name	Type	Description
index_owner	character varying(64)	Index owner.
index_name	character varying(64)	Index name.
table_owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.
column_name	name	Column name.
column_position	smallint	Position of a column in an index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
column_length	numeric	Column length. For the variable-length type, its value is **NULL**.
char_length	numeric	Maximum length of a column, in bytes.
descend	character varying(4)	Columns sorted in descending (**DESC**) or ascending (**ASC**) order.
collated_column_id	numeric	Not supported yet. Its value is **NULL**.

7.3.13.14 ADM_IND_EXPRESSIONS

ADM_IND_EXPRESSIONS displays information about expression indexes in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-296 ADM_IND_EXPRESSIONS columns

Name	Type	Description
table_owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.
index_owner	character varying(64)	Index owner.
index_name	character varying(64)	Index name.
column_expression	text	Function-based index expression of a specified column.
column_position	smallint	Position of a column in an index.

7.3.13.15 ADM_INDEXES

ADM_INDEXES displays all indexes in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-297 ADM_INDEXES columns

Name	Type	Description
owner	character varying(64)	Index owner
index_name	character varying(64)	Index name
table_name	character varying(64)	Name of the table corresponding to the index
uniqueness	text	Specifies whether the index is unique. • **UNIQUE**: unique index. • **NONUNIQUE**: non-unique index.
partitioned	character(3)	Specifies whether the index has the property of partitioned tables. • **Yes**: The index has the property of a partitioned table. • **No**: The index does not have the property of a partitioned table.
generated	character varying(1)	Specifies whether the name of the index is generated by the system. • **y**: The index name is generated by the system. • **n**: The index name is not generated by the system.
index_type	character varying(27)	Index type • **NORMAL**: Index attributes are simple references, and the expression tree is empty. • **FUNCTION-BASED NORMAL**: Expression trees are used for index attributes that are not simple column references.
table_owner	character varying(128)	Owner of the index object
table_type	character(11)	Type of an index object • **TABLE**: The index object is of the table type.
tablespace_name	character varying(30)	Name of the tablespace that contains the index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
status	character varying(8)	Status of a non-partitioned index. • **VALID**: Non-partitioned indexes can be used for query. • **UNUSABLE**: The non-partitioned index is unavailable. • **N/A**: The index has the property of a partitioned table.
compression	character varying(13)	Not supported yet. Its value is **NULL**.
prefix_length	numeric	Not supported yet. Its value is **NULL**.
ini_trans	numeric	Not supported yet. Its value is **NULL**.
max_trans	numeric	Not supported yet. Its value is **NULL**.
initial_extent	numeric	Not supported yet. Its value is **NULL**.
next_extent	numeric	Not supported yet. Its value is **NULL**.
min_extents	numeric	Not supported yet. Its value is **NULL**.
max_extents	numeric	Not supported yet. Its value is **NULL**.
pct_increase	numeric	Not supported yet. Its value is **NULL**.
pct_threshold	numeric	Not supported yet. Its value is **NULL**.
include_column	numeric	Not supported yet. Its value is **NULL**.
freelists	numeric	Not supported yet. Its value is **NULL**.
freelist_groups	numeric	Not supported yet. Its value is **NULL**.
pct_free	numeric	Not supported yet. Its value is **NULL**.
logging	character varying(3)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
blevel	numeric	Not supported yet. Its value is **NULL**.
leaf_blocks	numeric	Not supported yet. Its value is **NULL**.
distinct_keys	numeric	Not supported yet. Its value is **NULL**.
avg_leaf_blocks_per_key	numeric	Not supported yet. Its value is **NULL**.
avg_data_blocks_per_key	numeric	Not supported yet. Its value is **NULL**.
clustering_factor	numeric	Not supported yet. Its value is **NULL**.
num_rows	numeric	Not supported yet. Its value is **NULL**.
sample_size	numeric	Not supported yet. Its value is **NULL**.
last_analyzed	date	Not supported yet. Its value is **NULL**.
degree	character varying(40)	Not supported yet. Its value is **NULL**.
instances	character varying(40)	Not supported yet. Its value is **NULL**.
temporary	character varying(1)	Not supported yet. Its value is **NULL**.
secondary	character varying(1)	Not supported yet. Its value is **NULL**.
buffer_pool	character varying(7)	Not supported yet. Its value is **NULL**.
flash_cache	character varying(7)	Not supported yet. Its value is **NULL**.
cell_flash_cache	character varying(7)	Not supported yet. Its value is **NULL**.
user_stats	character varying(3)	Not supported yet. Its value is **NULL**.
duration	character varying(15)	Not supported yet. Its value is **NULL**.
pct_direct_access	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
ityp_owner	character varying(128)	Not supported yet. Its value is **NULL**.
ityp_name	character varying(128)	Not supported yet. Its value is **NULL**.
parameters	character varying(1000)	Not supported yet. Its value is **NULL**.
global_stats	character varying(3)	Not supported yet. Its value is **NULL**.
domidx_status	character varying(12)	Not supported yet. Its value is **NULL**.
domidx_opstatus	character varying(6)	Not supported yet. Its value is **NULL**.
funcidx_status	character varying(8)	Not supported yet. Its value is **NULL**.
join_index	character varying(3)	Not supported yet. Its value is **NULL**.
iot_redundant_pkey_elim	character varying(3)	Not supported yet. Its value is **NULL**.
dropped	character varying(3)	Not supported yet. Its value is **NULL**.
visibility	character varying(9)	Specifies whether the index is visible to the optimizer. • **VISIBLE**: The index is visible to the optimizer. • **INVISIBLE**: The index is visible to the optimizer.
domidx_management	character varying(14)	Not supported yet. Its value is **NULL**.
segment_created	character varying(3)	Not supported yet. Its value is **NULL**.
orphaned_entries	character varying(3)	Not supported yet. Its value is **NULL**.
indexing	character varying(7)	Not supported yet. Its value is **NULL**.
auto	character varying(3)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.16 ADM_OBJECTS

ADM_OBJECTS displays all database objects in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-298 ADM_OBJECTS columns

Name	Type	Description
owner	name	Object owner.
object_name	name	Object name.
object_id	oid	OID of the object.
object_type	name	Object type. For example, table, schema, and index.
namespace	oid	Namespace containing the object.
temporary	character(1)	Specifies whether an object is a temporary object.
status	character varying(7)	Object status. • **valid**: The object is valid. • **invalid**: The object is invalid.
subobject_name	name	Subobject name of an object.
generated	character(1)	Specifies whether an object name is generated by the system.
created	timestamp with time zone	Creation time of an object.
last_ddl_time	timestamp with time zone	Last time when an object was modified.
default_collation	character varying(100)	Default collation of objects.
data_object_id	numeric	Not supported yet. Its value is **NULL**.
timestamp	character varying(19)	Not supported yet. Its value is **NULL**.
secondary	character varying(1)	Not supported yet. Its value is **NULL**.
edition_name	character varying(128)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
sharing	character varying(18)	Not supported yet. Its value is **NULL**.
editionable	character varying(1)	Not supported yet. Its value is **NULL**.
oracle_maintained	character varying(1)	Not supported yet. Its value is **NULL**.
application	character varying(1)	Not supported yet. Its value is **NULL**.
duplicated	character varying(1)	Not supported yet. Its value is **NULL**.
sharded	character varying(1)	Not supported yet. Its value is **NULL**.
created_appid	numeric	Not supported yet. Its value is **NULL**.
modified_appid	numeric	Not supported yet. Its value is **NULL**.
created_vsnid	numeric	Not supported yet. Its value is **NULL**.
modified_vsnid	numeric	Not supported yet. Its value is **NULL**.

NOTICE

For details on the value ranges of **created** and **last_ddl_time**, see **PG_OBJECT**.

7.3.13.17 ADM_PROCEDURES

ADM_PROCEDURES displays information about all stored procedures, functions, and triggers in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-299 ADM_PROCEDURES columns

Name	Type	Description
owner	character varying(64)	Owner of a stored procedure, function, trigger, or package.
object_name	character varying(64)	Name of a stored procedure, function, or trigger. This column will be a package name if the object is a function in a package or a stored procedure.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
procedure_name	character varying(128)	This column will be the name of a function or stored procedure in a package if **object_name** is a package name. Otherwise, the column is empty.
object_id	oid	OID of a stored procedure, function, trigger, or package.
subprogram_id	numeric	Location of a function or stored procedure in a package.
overload	character varying(40)	*n*th overloaded function of the name.
object_type	character varying(13)	Object type.
aggregate	character varying(3)	Specifies whether the function is an aggregate function: • **yes** • **no**
pipelined	character varying(3)	Not supported. The value is **no**.
impltypeowner	character varying(128)	Owner of an implementation type.
impltypename	character varying(128)	Name of an implementation type.
parallel	character varying(3)	Not supported. The value is **no**.
interface	character varying(3)	Not supported. The value is **no**.
deterministic	character varying(3)	Not supported. The value is **no**.
authid	character varying(12)	Permission type. • **definer**: The creator permission is used. • **current_user**: The caller permission is used. This column conflicts with reserved keywords. Therefore, add the view name when calling this column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
result_cache	character varying(3)	Not supported yet. Its value is **NULL**.
origin_con_id	character varying(256)	Not supported. The value is **0**.
polymorphic	character varying(5)	Not supported yet. Its value is **NULL**.
argument_number	smallint	Number of input parameters in the stored procedure.

7.3.13.18 ADM_SCHEDULER_JOB_ARGS

ADM_SCHEDULER_JOB_ARGS displays parameters related to all jobs in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-300 ADM_SCHEDULER_JOB_ARGS columns

Name	Type	Description
owner	character varying(128)	Owner of the job to which the parameter belongs.
job_name	character varying(128)	Name of the job to which the parameter belongs.
argument_name	character varying(128)	Parameter name.
argument_position	numeric	Position of the parameter in the parameter list.
argument_type	character varying(257)	Data type of a parameter, which can be user-defined.
value	character varying(4000)	Parameter value.
anydata_value	character varying(4000)	Not supported yet. Its value is **NULL**.
out_argument	character varying(5)	Reserved column. The value is **NULL**.

7.3.13.19 ADM_SCHEDULER_JOBS

ADM_SCHEDULER_JOBS displays information about all DBE_SCHEDULER scheduler jobs in the database. By default, only users with the SYSADMIN

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-301 ADM_SCHEDULER_JOBS columns

Name	Type	Description
owner	name	Owner of a scheduler job.
job_name	text	Name of a scheduler job.
job_subname	character varying(128)	Not supported yet. Its value is **NULL**.
job_style	text	Action mode of a scheduler job. It is specified during creation. Its value can only be **"REGULAR"**. If this parameter is not specified, the value is **NULL**.
job_creator	name	Creator of a scheduler job.
client_id	character varying(65)	Not supported yet. Its value is **NULL**.
global_uid	character varying(33)	Not supported yet. Its value is **NULL**.
program_owner	character varying(4000)	Owner of a program referenced by a scheduler job.
program_name	text	Name of the program referenced by a scheduler job.
job_type	character varying(16)	Inline program type of a scheduled task. The options are as follows: • **PLSQL_BLOCK**: fast anonymous stored procedure. • **STORED_PROCEDURE**: stored procedure that is saved. • **EXTERNAL_SCRIPT**: external script.
job_action	text	Program content of a scheduler job.
number_of_arguments	text	Number of parameters of a scheduler job. The value ranges from 0 to 255.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
schedule_owner	character varying(4000)	Not supported yet. Its value is **NULL**.
schedule_name	text	Name of the schedule referenced by a scheduler job.
schedule_type	character varying(12)	Not supported yet. Its value is **NULL**.
start_date	timestamp without time zone	Start time of a scheduler job.
repeat_interval	text	Period of a scheduler job.
event_queue_owner	character varying(128)	Not supported yet. Its value is **NULL**.
event_queue_name	character varying(128)	Not supported yet. Its value is **NULL**.
event_queue_agent	character varying(523)	Not supported yet. Its value is **NULL**.
event_condition	character varying(4000)	Not supported yet. Its value is **NULL**.
event_rule	character varying(261)	Not supported yet. Its value is **NULL**.
file_watcher_owner	character varying(261)	Not supported yet. Its value is **NULL**.
file_watcher_name	character varying(261)	Not supported yet. Its value is **NULL**.
end_date	timestamp without time zone	End time of a scheduler job.
job_class	text	Name of the scheduler job class to which a scheduler job belongs.
enabled	boolean	Status of a scheduler job.
auto_drop	text	Status of the automatic deletion function of a scheduler job.
restart_on_recovery	character varying(5)	Not supported yet. Its value is **NULL**.
restart_on_failure	character varying(5)	Not supported yet. Its value is **NULL**.
state	"char"	Status of a scheduler job.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
job_priority	numeric	Not supported yet. Its value is **NULL**.
run_count	numeric	Not supported yet. Its value is **NULL**.
uptime_run_count	numeric	Not supported yet. Its value is **NULL**.
max_runs	numeric	Not supported yet. Its value is **NULL**.
failure_count	smallint	Number of scheduler job failures.
uptime_failure_count	numeric	Not supported yet. Its value is **NULL**.
max_failures	numeric	Maximum number of failures allowed before the status of a scheduler job is marked as broken.
retry_count	numeric	Not supported yet. Its value is **NULL**.
last_start_date	timestamp without time zone	Last time when a scheduler job was started.
last_run_duration	interval day to second(6)	Last execution duration of a scheduler job.
next_run_date	timestamp without time zone	Next execution time of a scheduler job.
schedule_limit	interval day to second(0)	Not supported yet. Its value is **NULL**.
max_run_duration	interval day to second(0)	Not supported yet. Its value is **NULL**.
logging_level	character varying(11)	Not supported yet. Its value is **NULL**.
store_output	character varying(5)	Specifies whether to store the output information of all scheduler jobs.
stop_on_window_close	character varying(5)	Not supported yet. Its value is **NULL**.
instance_stickiness	character varying(5)	Not supported yet. Its value is **NULL**.
raise_events	character varying(4000)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
system	character varying(5)	Not supported yet. Its value is **NULL**.
job_weight	numeric	Not supported yet. Its value is **NULL**.
nls_env	character varying(4000)	Not supported yet. Its value is **NULL**.
source	character varying(128)	Not supported yet. Its value is **NULL**.
number_of_destinations	numeric	Not supported yet. Its value is **NULL**.
destination_owner	character varying(261)	Not supported yet. Its value is **NULL**.
destination	text	Target name of a scheduler job. If the value of **destination** is **CCN**, the task is a CCN job. For details, see **DBE_SCHEDULER**.
credential_owner	character varying(128)	Not supported yet. Its value is **NULL**.
credential_name	text	Certificate name of a scheduler job.
instance_id	oid	OID of the current database.
deferred_drop	character varying(5)	Not supported yet. Its value is **NULL**.
allow_runs_in_restricted_mode	character varying(5)	Not supported yet. Its value is **NULL**.
comments	text	Comments of a scheduler job.
flags	numeric	Not supported yet. Its value is **NULL**.
restartable	character varying(5)	Not supported yet. Its value is **NULL**.
has_constraints	character varying(5)	Not supported yet. Its value is **NULL**.
connect_credential_owner	character varying(128)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
connect_credential_name	character varying(128)	Not supported yet. Its value is **NULL**.
fail_on_script_error	character varying(5)	Not supported yet. Its value is **NULL**.

7.3.13.20 ADM_SCHEDULER_PROGRAM_ARGS

ADM_SCHEDULER_PROGRAM_ARG displays parameters related to all programs in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-302 ADM_SCHEDULER_PROGRAM_ARGS columns

Name	Type	Description
owner	character varying(128)	Owner of the program to which the parameter belongs.
program_name	character varying(128)	Name of the program to which the parameter belongs.
argument_name	character varying(128)	Parameter name.
argument_position	numeric	Position of the parameter in the parameter list.
argument_type	character varying(257)	Data type of a parameter, which can be user-defined.
metadata_attribute	character varying(19)	Not supported yet. Its value is **NULL**.
default_value	character varying(4000)	Default parameter value.
default_anydata_value	character varying(4000)	Not supported yet. Its value is **NULL**.
out_argument	character varying(5)	Reserved column. The value is **NULL**.

7.3.13.21 ADM_SCHEDULER_PROGRAMS

ADM_SCHEDULER_PROGRAMS displays information about all programs that can be scheduled in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-303 ADM_SCHEDULER_PROGRAMS columns

Name	Type	Description
owner	name	Owner of a scheduler program.
program_name	text	Name of a scheduler program.
program_type	character varying(16)	Type of a scheduler program. The options are as follows: • **PLSQL_BLOCK**: fast anonymous stored procedure. • **STORED_PROCEDURE**: stored procedure that is saved. • **EXTERNAL_SCRIPT**: external script.
program_action	text	Action performed by a scheduler program.
number_of_arguments	numeric	Number of scheduler program parameters. The value ranges from 0 to 255.
enabled	character varying(5)	Specifies whether a scheduler program is enabled.
comments	text	Comments of a scheduler program.
detached	character varying(5)	Not supported yet. Its value is **NULL**.
schedule_limit	interval day to second(0)	Not supported yet. Its value is **NULL**.
priority	numeric	Not supported yet. Its value is **NULL**.
weight	numeric	Not supported yet. Its value is **NULL**.
max_runs	numeric	Not supported yet. Its value is **NULL**.
max_failures	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
max_run_duration	interval day to second(0)	Not supported yet. Its value is **NULL**.
has_constraints	character varying(5)	Not supported yet. Its value is **NULL**.
nls_env	character varying(4000)	Not supported yet. Its value is **NULL**.

7.3.13.22 ADM_SCHEDULER_RUNNING_JOBS

ADM_SCHEDULER_RUNNING_JOBS displays information about all **DBE_SCHEDULER** jobs that are being executed in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-304 ADM_SCHEDULER_RUNNING_JOBS columns

Name	Type	Description
owner	character varying(128)	Owner of a scheduler job.
job_name	character varying(128)	Name of a scheduler job.
job_subname	character varying(128)	Not supported yet. Its value is **NULL**.
job_style	character varying(17)	Action mode of a scheduler job. It is specified during creation. Its value can only be **"REGULAR"**. If this parameter is not specified, the value is **NULL**.
detached	character varying(5)	Not supported yet. Its value is **NULL**.
session_id	numeric	ID of the session that executes a scheduler job
slave_process_id	numeric	Not supported yet. Its value is **NULL**.
slave_os_process_id	character varying(12)	ID of the process that executes a scheduler job
running_instance	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
resource_consumer_group	character varying(32)	Not supported yet. Its value is **NULL**.
elapsed_time	interval day to second(2)	Execution duration of a scheduler job
cpu_used	interval day to second(2)	Not supported yet. Its value is **NULL**.
destination_owner	character varying(261)	Not supported yet. Its value is **NULL**.
destination	character varying(261)	Target name of a scheduler job. If the value of **destination** is **CCN**, the task is a CCN job. For details, see **DBE_SCHEDULER**.
credential_owner	character varying(128)	Not supported yet. Its value is **NULL**.
credential_name	character varying(128)	Certificate name of a scheduler job.
log_id	numeric	Not supported yet. Its value is **NULL**.

7.3.13.23 ADM_SEGMENTS

ADM_SEGMENTS displays the storage space allocated to all segments in the database. It exists in the PG_CATALOG and SYS schemas. Only system administrators can access this view. The information cannot be obtained from the system catalog. The view is empty.

Table 7-305 ADM_SEGMENTS columns

Name	Type	Description
owner	character varying(128)	Not supported yet. Its value is **NULL**.
segment_name	character varying(128)	Not supported yet. Its value is **NULL**.
partition_name	character varying(128)	Not supported yet. Its value is **NULL**.
segment_type	character varying(18)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
segment_subtype	character varying(10)	Not supported yet. Its value is **NULL**.
tablespace_name	character varying(30)	Not supported yet. Its value is **NULL**.
header_file	numeric	Not supported yet. Its value is **NULL**.
header_block	numeric	Not supported yet. Its value is **NULL**.
bytes	numeric	Not supported yet. Its value is **NULL**.
blocks	numeric	Not supported yet. Its value is **NULL**.
extents	numeric	Not supported yet. Its value is **NULL**.
initial_extent	numeric	Not supported yet. Its value is **NULL**.
next_extent	numeric	Not supported yet. Its value is **NULL**.
min_extents	numeric	Not supported yet. Its value is **NULL**.
max_extents	numeric	Not supported yet. Its value is **NULL**.
max_size	numeric	Not supported yet. Its value is **NULL**.
retention	character varying(7)	Not supported yet. Its value is **NULL**.
minretention	numeric	Not supported yet. Its value is **NULL**.
pct_increase	numeric	Not supported yet. Its value is **NULL**.
freelists	numeric	Not supported yet. Its value is **NULL**.
freelist_groups	numeric	Not supported yet. Its value is **NULL**.
relative_fno	numeric	Not supported yet. Its value is **NULL**.
buffer_pool	character varying(7)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
flash_cache	character varying(7)	Not supported yet. Its value is **NULL**.
cell_flash_cache	character varying(7)	Not supported yet. Its value is **NULL**.
inmemory	character varying(8)	Not supported yet. Its value is **NULL**.
inmemory_priority	character varying(8)	Not supported yet. Its value is **NULL**.
inmemory_distribute	character varying(15)	Not supported yet. Its value is **NULL**.
inmemory_duplicate	character varying(13)	Not supported yet. Its value is **NULL**.
inmemory_compression	character varying(17)	Not supported yet. Its value is **NULL**.
cellmemory	character varying(24)	Not supported yet. Its value is **NULL**.

7.3.13.24 ADM_SEQUENCES

ADM_SEQUENCES displays information about all sequences in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-306 ADM_SEQUENCES columns

Name	Type	Description
sequence_owner	character varying(64)	Owner of a sequence.
sequence_name	character varying(64)	Name of a sequence.
min_value	int16	Minimum value of a sequence.
max_value	int16	Maximum value of a sequence.
increment_by	int16	Increment of a sequence.
last_number	int16	Last value of the sequence allocated in a batch from the GTM at a time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
cache_size	int16	Size of the sequence disk cache.
cycle_flag	character(1)	Specifies whether a sequence is a cyclic sequence. The value can be **y** or **n**. • **y**: yes. • **n**: no.
order_flag	character(1)	Specifies whether a sequence is an ordered sequence. Value range: • **y**: yes. • **n**: no.

7.3.13.25 ADM_SOURCE

ADM_SOURCE displays the definition information about all stored procedures, functions, and triggers in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-307 ADM_SOURCE columns

Name	Type	Description
owner	name	Object owner.
name	name	Object name.
type	name	Object type. Its value can be **function**, **procedure**, or **trigger**.
line	numeric	Number of the line in the definition information.
text	text	Text source of the storage object.
origin_con_id	character varying(256)	Not supported. The value is **0**.

7.3.13.26 ADM_SYNONYMS

ADM_SYNONYMS displays all synonyms in the database. By default, only users with the SYSADMIN permission can access this view. This view exists in the PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-308 ADM_SYNONYMS columns

Name	Type	Description
owner	text	Owner of a synonym The owner of the PUBLIC synonym is PUBLIC.
schema_name	text	Name of the schema to which a synonym belongs. The name of the schema to which the PUBLIC synonym belongs is NULL.
synonym_name	text	Synonym name
table_owner	text	Owner of the associated object. Although the column is called **table_owner**, the associated object is not necessarily a table. It can be any common database object, such as a view, stored procedure, or synonym.
table_name	text	Name of the associated object. Although the column is called **table_name**, the associated object is not necessarily a table. It can be any common database object, such as a view, stored procedure, or synonym.
table_schema_name	text	Schema name of the associated object. Although the column is called **table_schema_name**, the associated object is not necessarily a table. It can be any common database object, such as a view, stored procedure, or synonym.

7.3.13.27 ADM_TAB_COL_STATISTICS

ADM_TAB_COL_STATISTICS displays column statistics and histogram information extracted from ADM_TAB_COLUMNS. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-309 ADM_TAB_COL_STATISTICS columns

Name	Type	Description
owner	character varying(128)	Table owner.
table_name	character varying(128)	Table name.
column_name	character varying(128)	Column name.
num_distinct	numeric	Number of different values in a column.
low_value	raw	Low value in a column.
high_value	raw	High value in a column.
density	numeric	• If there is a histogram on **COLUMN_NAME**, this column displays the selectivity of values in the histogram that span less than two endpoints. It does not represent the selectivity of values that span two or more endpoints. • If no histogram is available on **COLUMN_NAME**, the value of this column is **1/NUM_DISTINCT**.
num_nulls	numeric	Number of empty values in a column.
num_buckets	numeric	Number of buckets in the histogram of a column.
sample_size	numeric	Sample size used to analyze a column.
last_analyzed	date	Date when a column was last analyzed. Database restart is not supported. Otherwise, data loss will occur. **TIMESTAMP (0) WITHOUT TIME ZONE** is displayed in an O-compatible database.
global_stats	character varying(3)	Not supported yet. Its value is **NULL**.
user_stats	character varying(3)	Not supported yet. Its value is **NULL**.
notes	character varying(99)	Not supported yet. Its value is **NULL**.
avg_col_len	numeric	Average length of a column, in bytes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
histogram	character varying(15)	Specifies whether the histogram exists and the type of the histogram. • **NONE**: no histogram. • **FREQUENCY**: frequency histogram. • **EQUI-WIDTH**: equal-width histogram.
scope	character varying(7)	For statistics collected on any table other than global temporary tables, the value is SHARED (indicating that the statistics are shared among all sessions).
schema	character varying(64)	Name of the namespace to which the column belongs.

NOTE

During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of the **last_analyzed** column in the view is changed to **datea**.

7.3.13.28 ADM_TAB_COLS

ADM_TAB_COLS displays information about table and view columns. Each column of each table and view in the database has a row of data in ADM_TAB_COLS. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas. The number of rows displayed in this view is the same as that in the ADM_TAB_COLUMNS view. Only columns are different.

Table 7-310 ADM_TAB_COLS columns

Name	Type	Description
owner	character varying(64)	Owner of a table or view.
table_name	character varying(128)	Name of the table or view.
column_name	character varying(128)	Column name.
data_type	character varying(128)	Data type of a column, which can be a user-defined data type.
data_type_mod	character varying(3)	Not supported yet. Its value is **NULL**.
data_type_owner	character varying(128)	Owner of the data type of a column.
data_length	numeric	Length of a column, in bytes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
data_precision	numeric	Precision of the data type. This column is valid for the numeric data type and is **NULL** for other types.
data_scale	numeric	Number of digits to the right of the decimal point. This column is valid for the numeric data type and is **0** for other types.
nullable	character varying(1)	Specifies whether the column can be null. For PRIMARY KEY constraints and NOT NULL constraints, the value is **n**.
column_id	numeric	Sequence number of a column when a table is created.
default_length	numeric	Length of the default value of a column, in bytes.
data_default	text	Default value of a column.
num_distinct	numeric	Number of different values in a column.
low_value	raw	Minimum value in a column.
high_value	raw	Maximum value in a column.
density	numeric	Column density.
num_nulls	numeric	Number of empty values in a column.
num_buckets	numeric	Number of buckets in the histogram of a column.
last_analyzed	date	Last analysis date.
sample_size	numeric	Sample size used to analyze a column.
character_set_name	character varying(44)	Not supported yet. Its value is **NULL**.
char_col_decl_length	numeric	Declaration length of a column of the character type.
global_stats	character varying(3)	Not supported. Its value is **NO**.
user_stats	character varying(3)	Not supported. Its value is **NO**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
avg_col_len	numeric	Average length of a column, in bytes.
char_length	numeric	Length of a column (in characters). This column is valid only for the varchar, nvarchar2, bpchar, and char types.
char_used	character varying(1)	Not supported currently. ● **B**: varchar, bpchar, and char. ● **C**: nvarchar2. ● **NULL**: other types.
v80_fmt_image	character varying(3)	Not supported yet. Its value is **NULL**.
data_upgraded	character varying(3)	Not supported. Its value is **YES**.
hidden_column	character varying(3)	Not supported yet. Its value is **NULL**.
virtual_column	character varying	Specifies whether a column is a virtual column (a generated column). ● **YES**: yes. ● **NO**: no.
segment_column_id	numeric	Not supported yet. Its value is **NULL**.
internal_column_id	numeric	Internal sequence number of a column. The value is the same as that of **COLUMN_ID**.
histogram	character varying(15)	Specifies whether the histogram exists and the type of the histogram. ● **NONE**: no histogram. ● **FREQUENCY**: frequency histogram. ● **EQUI_WIDTH**: equal-width histogram.
qualified_col_name	character varying(64)	Qualified column name, which is the same as **COLUMN_NAME**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
user_generated	character varying(3)	Not supported. Its value is **YES**.
default_on_null	character varying(3)	Not supported yet. Its value is **NULL**.
identity_column	character varying(3)	Specifies whether a column is an identity column. • **yes**: It is an identity column. • **no**: It is not an identity column.
sensitive_column	character varying(3)	Not supported yet. Its value is **NULL**.
evaluation_edition	character varying(128)	Not supported yet. Its value is **NULL**.
unusable_before	character varying(128)	Not supported yet. Its value is **NULL**.
unusable_beginning	character varying(128)	Not supported yet. Its value is **NULL**.
collation	character varying(100)	Collation rule of a column. This column conflicts with reserved keywords. Therefore, add the view name when calling this column.
collated_column_id	numeric	Not supported yet. Its value is **NULL**.
schema	character varying(64)	Name of the namespace to which the column belongs.

NOTE

- During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of the **last_analyzed** column in the view is changed to **datea**.
- Currently, the CREATE TABLE syntax does not support the creation of tables with identity columns. When the ADM_TAB_COLS view is queried, the value of **identity_column** in the returned result is always **no**.

7.3.13.29 ADM_TAB_COLUMNS

ADM_TAB_COLUMNS displays information about the columns of tables and views. Each column of each table and view in the database has a row of data in ADM_TAB_COLUMNS. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-311 ADM_TAB_COLUMNS columns

Name	Type	Description
owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.
column_name	character varying(64)	Column name.
data_type	character varying(128)	Data type of a column.
data_type_mod	character varying(3)	Not supported yet. Its value is **NULL**.
data_type_owner	character varying(128)	Owner of the data type of a column.
data_length	integer	Length of a column, in bytes.
data_precision	integer	Precision of the data type. It is valid for the numeric data type and **NULL** for other types.
data_scale	integer	Number of decimal places. It is valid for the numeric data type and **0** for other data types.
nullable	bpchar	Specifies whether the column can be null. Value range: • **y**: yes. • **n**: no. For PRIMARY KEY constraints and NOT NULL constraints, the value is **n**.
column_id	integer	Sequence number of a column when a table is created.
default_length	numeric	Length of the default value of a column, in bytes. This column is left blank if no default value left.
data_default	text	Default value of a column.
num_distinct	numeric	Number of different values in a column.
low_value	raw	Minimum value in a column.
high_value	raw	Maximum value in a column.
density	numeric	Column density.
num_nulls	numeric	Number of empty values in a column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
num_buckets	numeric	Number of buckets in the histogram of a column.
last_analyzed	date	Last analysis date.
sample_size	numeric	Sample size used to analyze a column.
character_set_name	character varying(44)	Not supported yet. Its value is **NULL**.
char_col_decl_length	numeric	Declaration length of a column of the character type.
global_stats	character varying(3)	Not supported yet. Its value is **no**.
user_stats	character varying(3)	Not supported yet. Its value is **no**.
avg_col_len	numeric	Average length of a column, in bytes.
char_length	numeric	Column length (in characters), which takes effect only for the varchar, nvarchar2, bpchar, and char types.
char_used	character varying(1)	Not supported yet. • **b**: varchar, bpchar, and char. • **c**: nvarchar2. • **NULL**: other types.
v80_fmt_image	character varying(3)	Not supported yet. Its value is **NULL**.
data_upgraded	character varying(3)	Not supported yet. Its value is **yes**.
histogram	character varying(15)	Specifies whether the histogram exists and the type of the histogram. • **none**: no histogram. • **frequency**: frequency histogram. • **equi-width**: equal-width histogram.
default_on_null	character varying(3)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
identity_column	character varying(3)	Specifies whether a column is an identity column. • **yes**: It is an identity column. • **no**: It is not an identity column.
sensitive_column	character varying(3)	Not supported yet. Its value is **NULL**.
evaluation_edition	character varying(128)	Not supported yet. Its value is **NULL**.
unusable_before	character varying(128)	Not supported yet. Its value is **NULL**.
unusable_beginning	character varying(128)	Not supported yet. Its value is **NULL**.
collation	character varying(100)	Collation rule of a column. This column conflicts with reserved keywords. Therefore, add the view name when calling this column.
comments	text	Comment of a column.
schema	character varying(64)	Name of the namespace to which the column belongs.

NOTE

- During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of the **last_analyzed** column in the view is changed to **datea**.
- Currently, the CREATE TABLE syntax does not support the creation of tables with identity columns. When the ADM_TAB_COLUMNS view is queried, the value of **identity_column** in the returned result is always **no**.

7.3.13.30 ADM_TAB_COMMENTS

ADM_TAB_COMMENTS displays comments about all tables and views in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-312 ADM_TAB_COMMENTS columns

Name	Type	Description
owner	character varying(64)	Owner of a table or view.
table_name	character varying(64)	Name of a table or view.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
comments	text	Comments.
schema	character varying(64)	Name of the namespace to which the table belongs.

7.3.13.31 ADM_TAB_HISTOGRAMS

ADM_TAB_HISTOGRAMS displays the histogram information about all tables and views in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-313 ADM_TAB_HISTOGRAMS columns

Name	Type	Description
owner	character varying(128)	Table owner.
table_name	character varying(128)	Table name.
column_name	character varying(4000)	Column name.
endpoint_number	numeric	Bucket ID of the histogram.
endpoint_value	numeric	Not supported yet. Its value is **NULL**.
endpoint_actual_value	character varying(4000)	Actual value of the bucket endpoint.
endpoint_actual_value_raw	raw	Not supported yet. Its value is **NULL**.
endpoint_repeat_count	numeric	Not supported yet. Its value is **NULL**.
scope	character varying(7)	Not supported. The value is **shared**.

7.3.13.32 ADM_TAB_STATISTICS

ADM_TAB_STATISTICS displays optimizer statistics for all tables in the database. This view exists in the PG_CATALOG and SYS schemas. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-314 ADM_TAB_STATISTICS columns

Name	Type	Description
owner	character varying(128)	Object owner.
table_name	character varying(128)	Table name.
partition_name	character varying(128)	Not supported yet. Its value is **NULL**.
partition_position	numeric	Not supported yet. Its value is **NULL**.
subpartition_name	character varying(128)	Not supported yet. Its value is **NULL**.
subpartition_position	numeric	Not supported yet. Its value is **NULL**.
object_type	character varying(12)	Object type. The options are as follows: • **TABLE** • **PARTITION** • **SUBPARTITION**
num_rows	numeric	Number of rows in an object.
blocks	numeric	Not supported yet. Its value is **NULL**.
empty_blocks	numeric	Not supported yet. Its value is **NULL**.
avg_space	numeric	Not supported yet. Its value is **NULL**.
chain_cnt	numeric	Not supported yet. Its value is **NULL**.
avg_row_len	integer	Average row length, including the row overhead.
avg_space_freelist_blocks	numeric	Not supported yet. Its value is **NULL**.
num_freelist_blocks	numeric	Not supported yet. Its value is **NULL**.
avg_cached_blocks	numeric	Not supported yet. Its value is **NULL**.
avg_cache_hit_ratio	numeric	Not supported yet. Its value is **NULL**.
im_imcu_count	numeric	Not supported yet. Its value is **NULL**.
im_block_count	numeric	Not supported yet. Its value is **NULL**.
im_stat_update_time	timestamp(6) without time zone	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
scan_rate	numeric	Not supported yet. Its value is **NULL**.
sample_size	numeric	Number of samples used for analyzing a table.
last_analyzed	timestamp with time zone	Date when a table was last analyzed. Database restart is not supported. Otherwise, data loss will occur.
global_stats	character varying(3)	Not supported yet. Its value is **NULL**.
user_stats	character varying(3)	Not supported yet. Its value is **NULL**.
stattype_locked	character varying(5)	Not supported yet. Its value is **NULL**.
stale_stats	character varying(7)	Not supported yet. Its value is **NULL**.
notes	character varying(25)	Not supported yet. Its value is **NULL**.
scope	character varying(7)	Not supported. The default value is **SHARED**.

7.3.13.33 ADM_TAB_STATS_HISTORY

ADM_TAB_STATS_HISTORY provides historical statistics about all tables in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-315 ADM_TAB_STATS_HISTORY columns

Name	Type	Description
owner	character varying(128)	Object owner.
table_name	character varying(128)	Table name.
partition_name	character varying(128)	Not supported yet. Its value is **NULL**.
subpartition_name	character varying(128)	Not supported yet. Its value is **NULL**.
stats_update_time	timestamp(6) with time zone	Time when the statistics were updated. Database restart is not supported. Otherwise, data loss will occur.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.34 ADM_TABLES

ADM_TABLES displays all tables in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

Table 7-316 ADM_TABLES columns

Name	Type	Description
owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.
tablespace_name	character varying(64)	Tablespace name of the table.
dropped	character varying	Specifies whether the current table is deleted. ● **yes**: deleted. ● **no**: not deleted.
num_rows	numeric	Estimated number of rows in the table.
status	character varying(8)	Specifies whether the current table is valid. ● **valid**: The current table is valid. ● **unusable**: The current table is unavailable.
sample_size	numeric	Number of samples used for analyzing the table.
temporary	character(1)	Specifies whether the table is a temporary table. ● **y**: It is a temporary table. ● **n**: It is not a temporary table.
pct_free	numeric	Minimum percentage of free space in a block.
ini_trans	numeric	Initial number of transactions.
max_trans	numeric	Maximum number of transactions.
avg_row_len	integer	Average number of bytes in each row.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
partitioned	character varying(3)	Specifies whether the table is a partitioned table. • **yes**: It is a partitioned table. • **no**: It is not a partitioned table.
last_analyzed	timestamp with time zone	Last time when the table was analyzed. Database restart is not supported. Otherwise, data loss will occur.
row_movement	character varying(8)	Specifies whether to allow partition row movement. • **enabled**: Partition row movement is allowed. • **disabled**: Partition row movement is not allowed.
compression	character varying(8)	Specifies whether to enable table compression. • **enabled**: Table compression is enabled. • **disabled**: Table compression is disabled.
duration	character varying(15)	Time elapsed when a temporary table is processed. • **NULL**: The table is not a temporary table. • **sys$session**: The table is a temporary session table. • **sys$transaction**: The table is a temporary transaction table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
logical_replication	character varying(8)	Specifies whether to enable logical replication. • **enabled**: Logical replication is enabled. • **disabled**: Logical replication is disabled.
external	character varying(3)	Specifies whether the table is a foreign table. • **yes**: It is a foreign table. • **no**: It is not a foreign table.
logging	character varying(3)	Specifies whether to log changes made to the table. • **yes**: The database logs changes made to the table. • **no**: The database does not log changes made to the table.
default_collation	character varying(100)	Default collation of a table. • default
degree	character varying(10)	Number of instances in a scanned table.
table_lock	character varying(8)	Specifies whether to enable table-level lock. • **enabled**: Table-level lock is enabled. • **disabled**: Table-level lock is disabled.
nested	character varying(3)	Specifies whether the table is a nested table. • **yes**: It is a nested table. • **no**: It is not a nested table.
buffer_pool	character varying(7)	Default buffer pool of a table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
flash_cache	character varying(7)	Smart flash cache hint in database for a table block.
cell_flash_cache	character varying(7)	Cell flash cache hint for a table block.
skip_corrupt	character varying(8)	Specifies whether to skip corrupted blocks during table scanning. • **enabled**: Corrupted blocks will be skipped. • **disabled**: Corrupted blocks will not be skipped.
has_identity	character varying(3)	Specifies whether the table has an identity column. • **yes**: It has an identity column. • **no**: It has no identity column.
inmemory	character varying(8)	Not supported. The value is **disabled**.
segment_created	character varying(3)	Specifies whether a table segment has been created. • **yes**: A table segment has been created. • **no**: No table segment is created.
monitoring	character varying(3)	Specifies whether to track changes made to the table. • **yes**: The database tracks changes made to the table. • **no**: The database does not track changes made to the table.
cluster_name	character varying(128)	Not supported yet. Its value is **NULL**.
iot_name	character varying(128)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
pct_used	numeric	Not supported yet. Its value is **NULL**.
initial_extent	numeric	Not supported yet. Its value is **NULL**.
next_extent	numeric	Not supported yet. Its value is **NULL**.
min_extents	numeric	Not supported yet. Its value is **NULL**.
max_extents	numeric	Not supported yet. Its value is **NULL**.
pct_increase	numeric	Not supported yet. Its value is **NULL**.
freelists	numeric	Not supported yet. Its value is **NULL**.
freelist_groups	numeric	Not supported yet. Its value is **NULL**.
backed_up	character varying(1)	Not supported yet. Its value is **NULL**.
blocks	numeric	Not supported yet. Its value is **NULL**.
empty_blocks	numeric	Not supported yet. Its value is **NULL**.
avg_space	numeric	Not supported yet. Its value is **NULL**.
chain_cnt	numeric	Not supported yet. Its value is **NULL**.
avg_space_freelist_blocks	numeric	Not supported yet. Its value is **NULL**.
num_freelist_blocks	numeric	Not supported yet. Its value is **NULL**.
instances	character varying(10)	Not supported yet. Its value is **NULL**.
cache	character varying(5)	Not supported yet. Its value is **NULL**.
iot_type	character varying(12)	Not supported yet. Its value is **NULL**.
secondary	character varying(1)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
global_stats	character varying(3)	Not supported yet. Its value is **NULL**.
user_stats	character varying(3)	Not supported yet. Its value is **NULL**.
cluster_owner	character varying(30)	Not supported yet. Its value is **NULL**.
dependencies	character varying(8)	Not supported yet. Its value is **NULL**.
compression_for	character varying(30)	Not supported yet. Its value is **NULL**.
read_only	character varying(3)	Not supported yet. Its value is **NULL**.
result_cache	character varying(7)	Not supported yet. Its value is **NULL**.
clustering	character varying(3)	Not supported yet. Its value is **NULL**.
activity_tracking	character varying(23)	Not supported yet. Its value is **NULL**.
dml_timestamp	character varying(25)	Not supported yet. Its value is **NULL**.
container_data	character varying(3)	Not supported yet. Its value is **NULL**.
inmemory_priority	character varying(8)	Not supported yet. Its value is **NULL**.
inmemory_distribute	character varying(15)	Not supported yet. Its value is **NULL**.
inmemory_compression	character varying(17)	Not supported yet. Its value is **NULL**.
inmemory_duplicate	character varying(13)	Not supported yet. Its value is **NULL**.
duplicated	character varying(1)	Not supported yet. Its value is **NULL**.
sharded	character varying(1)	Not supported yet. Its value is **NULL**.
hybrid	character varying(3)	Not supported yet. Its value is **NULL**.
cellmemory	character varying(24)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
containers_default	character varying(3)	Not supported yet. Its value is **NULL**.
container_map	character varying(3)	Not supported yet. Its value is **NULL**.
extended_data_link	character varying(3)	Not supported yet. Its value is **NULL**.
extended_data_link_map	character varying(3)	Not supported yet. Its value is **NULL**.
inmemory_service	character varying(12)	Not supported yet. Its value is **NULL**.
inmemory_service_name	character varying(1000)	Not supported yet. Its value is **NULL**.
container_map_object	character varying(3)	Not supported yet. Its value is **NULL**.
memoptimize_read	character varying(8)	Not supported yet. Its value is **NULL**.
memoptimize_write	character varying(8)	Not supported yet. Its value is **NULL**.
has_sensitive_column	character varying(3)	Not supported yet. Its value is **NULL**.
admit_null	character varying(3)	Not supported yet. Its value is **NULL**.
data_link_dml_enabled	character varying(3)	Not supported yet. Its value is **NULL**.
object_id_type	character varying(16)	Not supported yet. Its value is **NULL**.
table_type_owner	character varying(128)	Not supported yet. Its value is **NULL**.
table_type	character varying(128)	Not supported yet. Its value is **NULL**.
compress_for	character varying(30)	Not supported yet. Its value is **NULL**.

7.3.13.35 ADM_TABLESPACES

ADM_TABLESPACES displays information about available tablespaces. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

and SYS schemas. The logical structure features of GaussDB are different from those of Oracle Database.

Table 7-317 ADM_TABLESPACES columns

Name	Type	Description
tablespace_name	character varying(64)	Tablespace name.
block_size	numeric	Not supported yet. Its value is **NULL**.
initial_extent	numeric	Not supported yet. Its value is **NULL**.
next_extent	numeric	Not supported yet. Its value is **NULL**.
min_extents	numeric	Not supported yet. Its value is **NULL**.
max_extents	numeric	Not supported yet. Its value is **NULL**.
max_size	numeric	Not supported yet. Its value is **NULL**.
pct_increase	numeric	Not supported yet. Its value is **NULL**.
min_extlen	numeric	Not supported yet. Its value is **NULL**.
contents	character varying(9)	Not supported yet. Its value is **NULL**.
status	character varying(9)	Not supported yet. Its value is **online**.
logging	character varying(9)	Not supported yet. Its value is **NULL**.
force_logging	character varying(3)	Not supported yet. Its value is **NULL**.
extent_management	character varying(10)	Not supported yet. Its value is **NULL**.
allocation_type	character varying(9)	Not supported yet. Its value is **NULL**.
plugged_in	character varying(3)	Not supported yet. Its value is **NULL**.
segment_space_management	character varying(6)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
def_tab_compression	character varying(8)	Not supported yet. Its value is **NULL**.
retention	character varying(11)	Not supported yet. Its value is **NULL**.
bigfile	character varying(3)	Not supported yet. Its value is **NULL**.
predicate_evaluation	character varying(7)	Not supported yet. Its value is **NULL**.
encrypted	character varying(3)	Not supported yet. Its value is **NULL**.
compress_for	character varying(30)	Not supported yet. Its value is **NULL**.
def_inmemory	character varying(8)	Not supported yet. Its value is **NULL**.
def_inmemory_priority	character varying(8)	Not supported yet. Its value is **NULL**.
def_inmemory_distribute	character varying(15)	Not supported yet. Its value is **NULL**.
def_inmemory_compression	character varying(17)	Not supported yet. Its value is **NULL**.
def_inmemory_duplicate	character varying(13)	Not supported yet. Its value is **NULL**.
shared	character varying(12)	Not supported yet. Its value is **NULL**.
def_index_compression	character varying(8)	Not supported yet. Its value is **NULL**.
index_compress_for	character varying(13)	Not supported yet. Its value is **NULL**.
def_cellmemory	character varying(14)	Not supported yet. Its value is **NULL**.
def_inmemory_service	character varying(12)	Not supported yet. Its value is **NULL**.
def_inmemory_service_name	character varying(1000)	Not supported yet. Its value is **NULL**.
lost_write_protect	character varying(7)	Not supported yet. Its value is **NULL**.
chunk_tablespace	character varying(1)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.36 ADM_TRIGGERS

ADM_TRIGGERS displays information about triggers in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-318 ADM_TRIGGERS columns

Name	Type	Description
owner	character varying(128)	Trigger owner.
trigger_name	character varying(64)	Trigger name.
trigger_type	character varying	Time when a trigger is triggered. Value range: **before statement**, **before each row**, **after statement**, **after each row**, and **instead of**.
triggering_event	character varying	Event that triggers the trigger. Value range: **update**, **insert**, **delete**, and **truncate**.
table_owner	character varying(64)	Owner of the table that defines a trigger.
base_object_type	character varying(18)	Defines the basic object of a trigger. Value range: **table** and **view**.
table_name	character varying(64)	Name of the table or view that defines a trigger.
column_name	character varying(4000)	Not supported yet. Its value is **NULL**.
referencing_name	character varying(422)	Not supported. Its value is **referencing new as new old as old**.
when_clause	character varying(4000)	Content of the WHEN clause. **trigger_body** can be executed only when the value is **true**.
status	character varying(64)	Trigger status. • **O**: The trigger is enabled in origin or local mode. • **D**: The trigger is disabled. • **R**: The trigger is enabled in replica mode. • **A**: The trigger is always enabled.
description	character varying(4000)	Trigger description. It is used to create the trigger creation statement again.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
action_type	character varying(11)	Action type of a trigger, which only supports **call**.
trigger_body	text	Statement executed when a trigger is triggered.
crossedition	character varying(7)	Not supported. Its value is **no**.
before_statement	character varying(3)	Not supported. Its value is **no**.
before_row	character varying(3)	Not supported. Its value is **no**.
after_row	character varying(3)	Not supported. Its value is **no**.
after_statement	character varying(3)	Not supported. Its value is **no**.
instead_of_row	character varying(3)	Not supported. Its value is **no**.
fire_once	character varying(3)	Not supported yet. Its value is **NULL**.
apply_server_only	character varying(3)	Not supported yet. Its value is **NULL**.

7.3.13.37 ADM_TYPE_ATTRS

ADM_TYPE_ATTRS displays the attributes of the current database object type. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas.

Table 7-319 ADM_TYPE_ATTRS columns

Name	Type	Description
owner	oid	Owner of the type
type_name	name	Data type name
attr_name	name	Column name

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
attr_type_mod	integer	Type-specific data supplied at the table creation time (for example, the maximum length of a **varchar** column). This column is used as the third parameter when passing to type-specific input functions and length coercion functions. The value will generally be **-1** for types that do not need ATTTYPMOD.
attr_type_owner	oid	Owner of an attribute of this type
attr_type_name	name	Data type attribute name. Currently, this column records the type name after conversion.
length	smallint	Number of bytes in the internal representation of the type for a fixed-size type. It is a negative number for a variable-length type. ● The value **-1** indicates a "varlena" type (one that has a length word). ● The value **-2** indicates a NULL-terminated C string.
precision	integer	Precision of the numeric type
scale	integer	Range of the numeric type
character_set_name	character(1)	Character set name of an attribute. ● **C**: CHAR_CS. ● **N**: NCHAR_CS.
attr_no	smallint	Attribute number
inherited	character(1)	Specifies whether the attribute is inherited from the super type (**y** or **n**).
attr_length	integer	Type-specific data supplied at the table creation time (for example, the maximum length of a **varchar** column). The raw type is not recorded due to kernel implementation.

7.3.13.38 ADM_TYPES

ADM_TYPES describes all object types in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-320 ADM_TYPES columns

Name	Type	Description
owner	character varying(128)	Owner of the type.
type_name	character varying(128)	Type name.
type_oid	raw	Type identifier (OID).
typecode	character varying(128)	Type code.
attributes	numeric	Number of attributes in a type.
methods	numeric	Not supported. The value is **0**.
predefined	character varying(3)	Specifies whether the type is a predefined type.
incomplete	character varying(3)	Specifies whether the type is an incomplete type.
final	character varying(3)	Not supported yet. Its value is **NULL**.
instantiable	character varying(3)	Not supported yet. Its value is **NULL**.
persistable	character varying(3)	Not supported yet. Its value is **NULL**.
supertype_owner	character varying(128)	Not supported yet. Its value is **NULL**.
supertype_name	character varying(128)	Not supported yet. Its value is **NULL**.
local_attributes	numeric	Not supported yet. Its value is **NULL**.
local_methods	numeric	Not supported yet. Its value is **NULL**.
typeid	raw	Not supported yet. Its value is **NULL**.

7.3.13.39 ADM_VIEWS

ADM_VIEWS displays views in the database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-321 ADM_VIEWS columns

Name	Type	Description
owner	character varying(64)	Owner of the view.
view_name	character varying(64)	Name of the view.
text	text	Text of the view.
text_length	integer	Text length of the view.
text_vc	character varying(4000)	View creation statement. This column may truncate the view text. The BEQUEATH clause will not appear as part of the **TEXT_VC** column in this view.
type_text_length	numeric	Not supported yet. Its value is **NULL**.
type_text	character varying(4000)	Not supported yet. Its value is **NULL**.
oid_text_length	numeric	Not supported yet. Its value is **NULL**.
oid_text	character varying(4000)	Not supported yet. Its value is **NULL**.
view_type_owner	character varying(128)	Not supported yet. Its value is **NULL**.
view_type	character varying(128)	Not supported yet. Its value is **NULL**.
superview_name	character varying(128)	Not supported yet. Its value is **NULL**.
editioning_view	character varying(1)	Not supported yet. Its value is **NULL**.
read_only	character varying(1)	Not supported yet. Its value is **NULL**.
container_data	character varying(1)	Not supported yet. Its value is **NULL**.
bequeath	character varying(12)	Not supported yet. Its value is **NULL**.
origin_con_id	character varying(256)	Not supported yet. Its value is **NULL**.
default_collation	character varying(100)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
containers_default	character varying(3)	Not supported yet. Its value is **NULL**.
container_map	character varying(3)	Not supported yet. Its value is **NULL**.
extended_data_link	character varying(3)	Not supported yet. Its value is **NULL**.
extended_data_link_map	character varying(3)	Not supported yet. Its value is **NULL**.
has_sensitive_column	character varying(3)	Not supported yet. Its value is **NULL**.
admit_null	character varying(3)	Not supported yet. Its value is **NULL**.
pdb_local_only	character varying(3)	Not supported yet. Its value is **NULL**.

7.3.13.40 DB_ARGUMENTS

DB_ARGUMENTS displays parameter information about stored procedures and functions accessible to the current user. This view exists in both PG_CATALOG and SYS schemas. This view is accessible to all users and displays all information accessible to the current user.

Table 7-322 DB_ARGUMENTS columns

Name	Type	Description
owner	character varying(128)	Owner of a function or stored procedure.
object_name	character varying(128)	Name of a function or stored procedure.
package_name	character varying(128)	Not supported yet. Its value is **NULL**.
object_id	oid	OID of a function or stored procedure.
overload	character varying(40)	The *n*th overloaded function of the same name.
subprogram_id	numeric	Not supported yet. Its value is **NULL**.
argument_name	character varying(128)	Parameter name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
position	numeric	Position of the parameter in the parameter list. The value is **0** for the return value of the function by default.
sequence	numeric	Sequence of a parameter, which starts from 1, with the return type before all parameters.
data_level	numeric	Nesting depth of parameters of the composite type. The value of this column is always **0** because only one row is displayed for each parameter.
data_type	character varying(64)	Data type of a parameter.
defaulted	character varying(1)	Specifies whether a parameter has a default value: • **y**: yes. • **n**: no.
default_value	text	Not supported yet. Its value is **NULL**.
default_length	numeric	Not supported yet. Its value is **NULL**.
in_out	character varying(9)	Input and output attributes of a parameter: The options are as follows: • **in**: input parameter. • **out**: output parameter. • **in/out**: input and output parameters. • **variadic**: VARIADIC parameter.
data_length	numeric	Not supported yet. Its value is **NULL**.
data_precision	numeric	Not supported yet. Its value is **NULL**.
data_scale	numeric	Not supported yet. Its value is **NULL**.
radix	numeric	Radix of a numeric parameter. Its value is **10** for smallint, integer, bigint, numeric, and float, and **NULL** for other types.
character_set_name	character varying(44)	Not supported yet. Its value is **NULL**.
type_owner	character varying(128)	Owner of the data type.
type_name	character varying(128)	Parameter type name. Only the user-defined type is displayed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
type_subname	character varying(128)	Not supported yet. Its value is **NULL**.
type_link	character varying(128)	Not supported yet. Its value is **NULL**.
type_object_type	character varying(7)	Types of types of the **type_owner**, **type_name**, and **type_subname** columns: • **table**: The parameter is of the table type. • **view**: The parameter is of the view type. • **NULL**: The parameter is not of the table or view type.
pls_type	character varying(128)	Name of the PL/SQL type for parameters of the number type. Otherwise, this column is empty.
char_length	numeric	Not supported yet. Its value is **NULL**.
char_used	character varying(1)	Not supported. • The value **b** is for the varchar, nvarchar2, bpchar, or char type. • The value **NULL** is for other data types.
origin_con_id	character varying(256)	Not supported. The value is **0**.

7.3.13.41 DB_ALL_TABLES

DB_ALL_TABLES displays tables or views accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

Table 7-323 DB_ALL_TABLES columns

Name	Type	Description
owner	name	Owner of a table or view.
table_name	name	Name of a table or view.
tablespace_name	name	Tablespace where a table or view is located.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.42 DB_COL_COMMENTS

DB_COL_COMMENTS displays comment information about table columns accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

Table 7-324 DB_COL_COMMENTS columns

Name	Type	Description
owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.
column_name	character varying(64)	Column name.
comments	text	Comments.
schema	character varying(64)	Name of the namespace to which the column belongs.

7.3.13.43 DB_COLL_TYPES

DB_COLL_TYPES displays information about collection types that can be accessed by the current user. By default, they are accessible to all users. This view exists in the PG_CATALOG and SYS schemas.

Table 7-325 DB_COLL_TYPES columns

Name	Type	Description
owner	character varying(128)	Owner of a collection.
type_name	character varying(128)	Name of a collection.
coll_type	character varying(128)	Description of a collection.
upper_bound	numeric	Not supported yet. Its value is **NULL**.
elem_type_mod	character varying(7)	Type modifier of an element.
elem_type_owner	character varying(128)	Owner of the element type on which the collection is based. This parameter is mainly used for user-defined types.
elem_type_name	character varying(128)	Name of the data type or user-defined type on which the collection is based.
length	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
precision	numeric	Not supported yet. Its value is **NULL**.
scale	numeric	Not supported yet. Its value is **NULL**.
character_set_name	character varying(44)	Not supported yet. Its value is **NULL**.
elem_storage	character varying(7)	Not supported yet. Its value is **NULL**.
nulls_stored	character varying(3)	Not supported yet. Its value is **NULL**.
char_used	character varying(1)	Not supported yet. Its value is **NULL**.

7.3.13.44 DB_CONS_COLUMNS

DB_CONS_COLUMNS displays information about constraint columns accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

Table 7-326 DB_CONS_COLUMNS columns

Name	Type	Description
constraint_name	character varying(64)	Constraint name.
table_name	character varying(64)	Name of a constraint-related table.
column_name	character varying(64)	Name of a constraint-related column.
owner	character varying(64)	Constraint creator.
position	smallint	Position of a column in a table.

7.3.13.45 DB_CONSTRAINTS

DB_CONSTRAINTS displays information about constraints accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

Table 7-327 DB_CONSTRAINTS columns

Name	Type	Description
constraint_name	character varying(64)	Constraint name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
constraint_type	text	Constraint type. • **c**: CHECK constraint. • **f**: FOREIGN KEY constraint. • **p**: PRIMARY KEY constraint. • **u**: UNIQUE constraint.
table_name	character varying(64)	Name of a constraint-related table.
index_owner	character varying(64)	Owner of a constraint-related index (only for the UNIQUE constraint and PRIMARY KEY constraint).
index_name	character varying(64)	Name of a constraint-related index (only for the UNIQUE constraint and PRIMARY KEY constraint).
owner	character varying(64)	Constraint creator.

7.3.13.46 DB_DEPENDENCIES

DB_DEPENDENCIES displays the dependency relationship between types, tables, views, stored procedures, functions, and triggers accessible to the current user. It is accessible to all users. This view exists in the PG_CATALOG and SYS schemas.

Table 7-328 DB_DEPENDENCIES columns

Name	Type	Description
owner	name	Object owner.
name	name	Object name.
type	character varying(18)	Object type.
referenced_owner	name	Owner of the referenced object.
referenced_name	name	Name of the referenced object.
referenced_type	character varying(18)	Type of the referenced object.
referenced_link_name	character varying(128)	Not supported yet. Its value is **NULL**.
dependency_type	character varying(4)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.47 DB_IND_COLUMNS

DB_IND_COLUMNS displays all index columns accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

Table 7-329 DB_IND_COLUMNS columns

Name	Type	Description
index_owner	character varying(64)	Index owner.
index_name	character varying(64)	Index name.
table_owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.
column_name	name	Column name.
column_position	smallint	Position of a column in an index.
column_length	numeric	Column length.
char_length	numeric	Length of a column (character type).
descend	character varying(4)	Sorting mode of a column. The options are as follows: • **DESC**: descending order. • **ASC**: ascending order.
collated_column_id	numeric	Internal serial number of the language sorting columns.

7.3.13.48 DB_IND_EXPRESSIONS

DB_IND_EXPRESSIONS displays information about expression indexes accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

Table 7-330 DB_IND_EXPRESSIONS columns

Name	Type	Description
index_owner	character varying(64)	Index owner.
index_name	character varying(64)	Index name.
table_owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
column_expression	text	Function-based index expression of a specified column.
column_position	smallint	Position of a column in an index.

7.3.13.49 DB_INDEXES

DB_INDEXES displays information about indexes accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

Table 7-331 DB_INDEXES columns

Name	Type	Description
owner	character varying(64)	Index owner
index_name	character varying(64)	Index name
table_name	character varying(64)	Name of the table corresponding to the index
uniqueness	text	Specifies whether the index is unique. • **UNIQUE**: unique index. • **NONUNIQUE**: non-unique index.
partitioned	character(3)	Specifies whether the index has the property of partitioned tables. • **Yes**: The index has the property of a partitioned table. • **No**: The index does not have the property of a partitioned table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
generated	character varying(1)	Specifies whether the name of the index is generated by the system. • **y**: The index name is generated by the system. • **n**: The index name is not generated by the system.
index_type	character varying(27)	Index type • **NORMAL**: Index attributes are simple references, and the expression tree is empty. • **FUNCTION-BASED NORMAL**: Expression trees are used for index attributes that are not simple column references.
table_owner	character varying(128)	Owner of the index object
table_type	character(11)	Type of an index object • **TABLE**: The index object is of the table type.
tablespace_name	character varying(30)	Name of the tablespace that contains the index.
status	character varying(8)	Status of a non-partitioned index. • **VALID**: Non-partitioned indexes can be used for query. • **UNUSABLE**: The non-partitioned index is unavailable. • **N/A**: The index has the property of a partitioned table.
compression	character varying(13)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
prefix_length	numeric	Not supported yet. Its value is **NULL**.
ini_trans	numeric	Not supported yet. Its value is **NULL**.
max_trans	numeric	Not supported yet. Its value is **NULL**.
initial_extent	numeric	Not supported yet. Its value is **NULL**.
next_extent	numeric	Not supported yet. Its value is **NULL**.
min_extents	numeric	Not supported yet. Its value is **NULL**.
max_extents	numeric	Not supported yet. Its value is **NULL**.
pct_increase	numeric	Not supported yet. Its value is **NULL**.
pct_threshold	numeric	Not supported yet. Its value is **NULL**.
include_column	numeric	Not supported yet. Its value is **NULL**.
freelists	numeric	Not supported yet. Its value is **NULL**.
freelist_groups	numeric	Not supported yet. Its value is **NULL**.
pct_free	numeric	Not supported yet. Its value is **NULL**.
logging	character varying(3)	Not supported yet. Its value is **NULL**.
blevel	numeric	Not supported yet. Its value is **NULL**.
leaf_blocks	numeric	Not supported yet. Its value is **NULL**.
distinct_keys	numeric	Not supported yet. Its value is **NULL**.
avg_leaf_blocks_per_key	numeric	Not supported yet. Its value is **NULL**.
avg_data_blocks_per_key	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
clustering_factor	numeric	Not supported yet. Its value is **NULL**.
num_rows	numeric	Not supported yet. Its value is **NULL**.
sample_size	numeric	Not supported yet. Its value is **NULL**.
last_analyzed	date	Not supported yet. Its value is **NULL**.
degree	character varying(40)	Not supported yet. Its value is **NULL**.
instances	character varying(40)	Not supported yet. Its value is **NULL**.
temporary	character varying(1)	Not supported yet. Its value is **NULL**.
secondary	character varying(1)	Not supported yet. Its value is **NULL**.
buffer_pool	character varying(7)	Not supported yet. Its value is **NULL**.
flash_cache	character varying(7)	Not supported yet. Its value is **NULL**.
cell_flash_cache	character varying(7)	Not supported yet. Its value is **NULL**.
user_stats	character varying(3)	Not supported yet. Its value is **NULL**.
duration	character varying(15)	Not supported yet. Its value is **NULL**.
pct_direct_access	numeric	Not supported yet. Its value is **NULL**.
ityp_owner	character varying(128)	Not supported yet. Its value is **NULL**.
ityp_name	character varying(128)	Not supported yet. Its value is **NULL**.
parameters	character varying(1000)	Not supported yet. Its value is **NULL**.
global_stats	character varying(3)	Not supported yet. Its value is **NULL**.
domidx_status	character varying(12)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
domidx_opstatus	character varying(6)	Not supported yet. Its value is **NULL**.
funcidx_status	character varying(8)	Not supported yet. Its value is **NULL**.
join_index	character varying(3)	Not supported yet. Its value is **NULL**.
iot_redundant_pkey_elim	character varying(3)	Not supported yet. Its value is **NULL**.
dropped	character varying(3)	Not supported yet. Its value is **NULL**.
visibility	character varying(9)	Specifies whether the index is visible to the optimizer. • **VISIBLE**: The index is visible to the optimizer. • **INVISIBLE**: The index is visible to the optimizer.
domidx_management	character varying(14)	Not supported yet. Its value is **NULL**.
segment_created	character varying(3)	Not supported yet. Its value is **NULL**.
orphaned_entries	character varying(3)	Not supported yet. Its value is **NULL**.
indexing	character varying(7)	Not supported yet. Its value is **NULL**.
auto	character varying(3)	Not supported yet. Its value is **NULL**.

7.3.13.50 DB_OBJECTS

DB_OBJECTS displays all database objects accessible to the current user. This view exists in both PG_CATALOG and SYS schemas.

Table 7-332 DB_OBJECTS columns

Name	Type	Description
owner	name	Object owner.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
object_name	name	Object name.
object_id	oid	Object OID.
object_type	name	Object class.
namespace	oid	ID of the namespace where an object resides.
temporary	character(1)	Specifies whether an object is a temporary object.
status	character varying(7)	Object status. • **valid**: The object is valid. • **invalid**: The object is invalid.
subobject_name	name	Subobject name of an object.
generated	character(1)	Specifies whether an object name is generated by the system.
created	timestamp with time zone	Creation time of an object.
last_ddl_time	timestamp with time zone	Last modification time of an object.
default_collation	character varying(100)	Default collation of objects.
data_object_id	numeric	Not supported yet. Its value is **NULL**.
timestamp	character varying(19)	Not supported yet. Its value is **NULL**.
secondary	character varying(1)	Not supported yet. Its value is **NULL**.
edition_name	character varying(128)	Not supported yet. Its value is **NULL**.
sharing	character varying(18)	Not supported yet. Its value is **NULL**.
editionable	character varying(1)	Not supported yet. Its value is **NULL**.
oracle_maintained	character varying(1)	Not supported yet. Its value is **NULL**.
application	character varying(1)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
duplicated	character varying(1)	Not supported yet. Its value is **NULL**.
sharded	character varying(1)	Not supported yet. Its value is **NULL**.
created_appid	numeric	Not supported yet. Its value is **NULL**.
modified_appid	numeric	Not supported yet. Its value is **NULL**.
created_vsnid	numeric	Not supported yet. Its value is **NULL**.
modified_vsnid	numeric	Not supported yet. Its value is **NULL**.

NOTICE

For details about the value ranges of **created** and **last_ddl_time**, see **PG_OBJECT**.

7.3.13.51 DB_PROCEDURES

DB_PROCEDURES displays information about all stored procedures or functions accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

Table 7-333 DB_PROCEDURES columns

Name	Type	Description
owner	name	Object owner.
object_name	name	Object name.

7.3.13.52 DB_SCHEDULER_JOB_ARGS

DB_SCHEDULER_JOB_ARG displays the parameters related to the tasks accessible to the current user. This view is accessible to all users and displays all information accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

Table 7-334 DB_SCHEDULER_JOB_ARGS columns

Name	Type	Description
owner	character varying(128)	Owner of the job to which the parameter belongs.
job_name	character varying(128)	Name of the job to which the parameter belongs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
argument_name	character varying(128)	Parameter name.
argument_position	numeric	Position of the parameter in the parameter list.
argument_type	character varying(257)	Data type of a parameter, which can be user-defined.
value	character varying(4000)	Parameter value.
anydata_value	character varying(4000)	Not supported yet. Its value is **NULL**.
out_argument	character varying(5)	Reserved column. The value is **NULL**.

7.3.13.53 DB_SCHEDULER_PROGRAM_ARGS

DB_SCHEDULER_PROGRAM_ARG displays the parameters related to the programs accessible to the current user. This view is accessible to all users and displays all information accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

Table 7-335 DB_SCHEDULER_PROGRAM_ARGS columns

Name	Type	Description
owner	character varying(128)	Owner of the program to which the parameter belongs.
program_name	character varying(128)	Name of the program to which the parameter belongs.
argument_name	character varying(128)	Parameter name.
argument_position	numeric	Position of the parameter in the parameter list.
argument_type	character varying(257)	Data type of a parameter, which can be user-defined.
metadata_attribute	character varying(19)	Not supported yet. Its value is **NULL**.
default_value	character varying(4000)	Default parameter value.
anydata_default_value	character varying(4000)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
out_argument	character varying(5)	Reserved column. The value is **NULL**.

7.3.13.54 DB_SEQUENCES

DB_SEQUENCES displays all sequences accessible to the current user. This view exists in both PG_CATALOG and SYS schemas.

Table 7-336 DB_SEQUENCES columns

Name	Type	Description
sequence_owner	name	Owner of a sequence.
sequence_name	name	Name of the sequence.
min_value	int16	Minimum value of the sequence.
max_value	int16	Maximum value of the sequence.
increment_by	int16	Value by which the sequence is incremented.
cycle_flag	character(1)	Specifies whether a sequence is a cyclic sequence. Value range: • **y**: yes. • **n**: no.
order_flag	character(1)	Specifies whether a sequence is an ordered sequence. Value range: • **y**: yes. • **n**: no.
cache_size	int16	Size of the sequence disk cache.
last_number	int16	Last value of the sequence allocated in a batch from the GTM at a time.
scale_flag	character varying(1)	Specifies whether a sequence is a scalable sequence. Not supported yet. Its value is **NULL**.
extend_flag	character varying(1)	Specifies whether the value generated by a scalable sequence exceeds the maximum or minimum value of the sequence. Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
sharded_flag	character varying(1)	Specifies whether a sequence is a shard sequence. Not supported yet. Its value is **NULL**.
session_flag	character varying(1)	Specifies whether a sequence is a private session. Not supported yet. Its value is **NULL**.
keep_value	character varying(1)	Specifies whether to retain the sequence value during replay after a failure. Not supported yet. Its value is **NULL**.

7.3.13.55 DB_SOURCE

DB_SOURCE displays the definition information about stored procedures, functions, and triggers accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

Table 7-337 DB_SOURCE columns

Name	Type	Description
owner	name	Object owner.
name	name	Object name.
type	name	Object type. The value can be **function**, **procedure**, or **trigger**.
line	numeric	Number of the line in the definition information.
text	text	Text source of the storage object.
origin_con_id	character varying(256)	Not supported. The value is **0**.

7.3.13.56 DB_SYNONYMS

DB_SYNONYMS displays all synonyms accessible to the current user.

Table 7-338 DB_SYNONYMS columns

Name	Type	Description
owner	text	Owner of a synonym. The owner of the PUBLIC synonym is PUBLIC.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
schema_name	text	Name of the schema to which the synonym belongs. The name of the schema to which the PUBLIC synonym belongs is NULL.
synonym_name	text	Synonym name.
table_owner	text	Owner of the associated object. Although the column is called **table_owner**, the associated object is not necessarily a table. It can be any common database object, such as a view, stored procedure, or synonym.
table_name	text	Name of the associated object. Although the column is called **table_name**, the associated object is not necessarily a table. It can be any common database object, such as a view, stored procedure, or synonym.
table_schema_name	text	Schema name of the associated object. Although the column is called **table_schema_name**, the associated object is not necessarily a table. It can be any common database object, such as a view, stored procedure, or synonym.
db_link	character varying(128)	Reserved column. The value is **NULL**.
origin_con_id	character varying(256)	Not supported. The value is **0**.

7.3.13.57 DB_TAB_COL_STATISTICS

DB_TAB_COL_STATISTICS displays column statistics and histogram information extracted from DB_TAB_COLUMNS. All users can access this view. This view exists in both PG_CATALOG and SYS schemas.

Table 7-339 DB_TAB_COL_STATISTICS columns

Name	Type	Description
owner	character varying(128)	Table owner.
table_name	character varying(128)	Table name.
column_name	character varying(128)	Column name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
num_distinct	numeric	Number of different values in a column.
low_value	raw	Low value in a column.
high_value	raw	High value in a column.
density	numeric	• If there is a histogram on **COLUMN_NAME**, this column displays the selectivity of values in the histogram that span less than two endpoints. It does not represent the selectivity of values that span two or more endpoints. • If no histogram is available on **COLUMN_NAME**, the value of this column is **1/ NUM_DISTINCT**.
num_nulls	numeric	Number of empty values in a column.
num_buckets	numeric	Number of buckets in the histogram of a column.
sample_size	numeric	Sample size used to analyze a column.
last_analyzed	date	Date when a column was last analyzed. Database restart is not supported. Otherwise, data loss will occur. **TIMESTAMP (0) WITHOUT TIME ZONE** is displayed in an O-compatible database.
global_stats	character varying(3)	Not supported yet. Its value is **NULL**.
user_stats	character varying(3)	Not supported yet. Its value is **NULL**.
notes	character varying(99)	Not supported yet. Its value is **NULL**.
avg_col_len	numeric	Average length of a column, in bytes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
histogram	character varying(15)	Specifies whether the histogram exists and the type of the histogram. • **NONE**: no histogram. • **FREQUENCY**: frequency histogram. • **EQUI-WIDTH**: equal-width histogram.
scope	character varying(7)	For statistics collected on any table other than global temporary tables, the value is SHARED (indicating that the statistics are shared among all sessions).
schema	character varying(64)	Name of the namespace to which the column belongs.

NOTE

During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of the **last_analyzed** column in the view is changed to **datea**.

7.3.13.58 DB_TAB_COLUMNS

DB_TAB_COLUMNS displays description information about columns of tables and views accessible to the current user. This view exists in both PG_CATALOG and SYS schemas. This view is accessible to all users and displays all information accessible to the current user.

Table 7-340 DB_TAB_COLUMNS columns

Name	Type	Description
owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.
column_name	character varying(64)	Column name.
data_type	character varying(128)	Data type of a column.
data_type_mod	character varying(3)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
data_type_owner	character varying(128)	Owner of the data type of a column.
data_length	integer	Length of a column, in bytes.
data_precision	integer	Precision of the data type. This column is valid for the numeric data type and is **NULL** for other types.
data_scale	integer	Number of digits to the right of the decimal point. This column is valid for the numeric data type and is **0** for other types.
nullable	bpchar	Specifies whether the column can be null. Value range: • **y**: yes. • **n**: no. For PRIMARY KEY constraints and NOT NULL constraints, the value is **n**.
column_id	integer	Sequence number of a column when a table is created.
default_length	numeric	Length of the default value of a column, in bytes.
data_default	text	Default value of a column.
num_distinct	numeric	Number of different values in a column.
low_value	raw	Minimum value in a column.
high_value	raw	Maximum value in a column.
density	numeric	Column density.
num_nulls	numeric	Number of empty values in a column.
num_buckets	numeric	Number of buckets in the histogram of a column.
last_analyzed	date	Last analysis date.
sample_size	numeric	Sample size used to analyze a column.
character_set_name	character varying(44)	Not supported yet. Its value is **NULL**.
char_col_decl_length	numeric	Declaration length of a column of the character type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
global_stats	character varying(3)	Not supported. The value is **NO**.
user_stats	character varying(3)	Not supported. The value is **NO**.
avg_col_len	numeric	Average length of a column, in bytes.
char_length	numeric	Column length (in characters) which is valid only for varchar, nvarchar2, bpchar, and char types.
char_used	character varying(1)	Not supported. Set it to **b** for the varchar, bpchar, or char data type. Set it to **c** for the nvarchar2 data type. For other data types, set it to **NULL**.
v80_fmt_image	character varying(3)	Not supported yet. Its value is **NULL**.
data_upgraded	character varying(3)	Not supported. The value is **YES**.
histogram	character varying(15)	Specifies whether the histogram exists and the type of the histogram. • **none**: no histogram. • **frequency**: frequency histogram. • **equi_width**: equal-width histogram.
default_on_null	character varying(3)	Not supported yet. Its value is **NULL**.
identity_column	character varying(3)	Specifies whether a column is an identity column. • **yes**: It is an identity column. • **no**: It is not an identity column.
evaluation_edition	character varying(128)	Not supported yet. Its value is **NULL**.
unusable_before	character varying(128)	Not supported yet. Its value is **NULL**.
unusable_beginning	character varying(128)	Not supported yet. Its value is **NULL**.
collation	character varying(100)	Collation rule of a column. This column conflicts with reserved keywords. Therefore, add the view name when calling this column.
comments	text	Comments.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
schema	character varying(64)	Name of the namespace to which the column belongs.

NOTE

- During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of the **last_analyzed** column in the view is changed to **datea**.
- Currently, the CREATE TABLE syntax does not support the creation of tables with identity columns. When the DB_TAB_COLUMNS view is queried, the value of **identity_column** in the returned result is always **no**.

7.3.13.59 DB_TAB_COMMENTS

DB_TAB_COMMENTS displays comments about all tables and views accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

Table 7-341 DB_TAB_COMMENTS columns

Name	Type	Description
owner	character varying(128)	Owner of a table or view.
table_name	character varying(128)	Name of a table or view.
table_type	character varying(11)	Object type.
comments	text	Comments.
origin_con_id	numeric	Not supported. The value is **0**.
schema	character varying(64)	Name of the namespace to which the table belongs.

7.3.13.60 DB_TAB_HISTOGRAMS

DB_TAB_HISTOGRAMS displays histogram statistics about the tables and views accessible to the current user, that is, the distribution of data in each column of the table. All users can access this view. This view exists in both PG_CATALOG and SYS schemas.

Table 7-342 DB_TAB_HISTOGRAMS columns

Name	Type	Description
owner	character varying(128)	Table owner.
table_name	character varying(128)	Table name.
column_name	character varying(4000)	Column name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
endpoint_number	numeric	Bucket ID of the histogram.
endpoint_value	numeric	Not supported yet. Its value is **NULL**.
endpoint_actual_value	character varying(4000)	Actual value of the bucket endpoint.
endpoint_actual_value_raw	raw	Not supported yet. Its value is **NULL**.
endpoint_repeat_count	numeric	Not supported yet. Its value is **NULL**.
scope	character varying(7)	Not supported. The value is **shared**.

7.3.13.61 DB_TAB_MODIFICATIONS

DB_TAB_MODIFICATIONS displays statistics about modifications to tables accessible to the current user since the last statistics collection on the tables. Currently, this view displays only the statistics data of tables on which INSERT, DELETE, and UPDATE operations have been performed. All users can access this view. This view exists in both PG_CATALOG and SYS schemas.

Table 7-343 DB_TAB_MODIFICATIONS columns

Name	Type	Description
table_owner	character varying(128)	Table owner.
table_name	character varying(128)	Table name.
partition_name	character varying(128)	Partition name.
subpartition_name	character varying(128)	Subpartition name.
inserts	numeric	Approximate number of insertions since the last statistics collection.
updates	numeric	Approximate number of updates since the last statistics collection.
deletes	numeric	Approximate number of deletions since the last statistics collection.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
timestamp	date	Last modification time. Currently, the modification time of a partitioned table is not supported. The value is **NULL**. • **TIMESTAMP (0) WITHOUT TIME ZONE** is displayed in an O-compatible database.
truncated	character varying(3)	Not supported yet. Its value is **NULL**.
drop_segments	numeric	Not supported yet. Its value is **NULL**.
schema_name	character varying(128)	Name of the schema to which the table belongs.

NOTE

During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of the **timestamp** column in the view is changed to **datea**.

7.3.13.62 DB_TAB_STATS_HISTORY

DB_TAB_STATS_HISTORY records the tables, partitions, or subpartitions involved in table statistics and the time when table statistics are collected. All users can access this view. This view exists in the PG_CATALOG and SYS schemas.

Table 7-344 DB_TAB_STATS_HISTORY columns

Name	Type	Description
owner	character varying(128)	Object owner.
table_name	character varying(128)	Table name.
partition_name	character varying(128)	Not supported yet. Its value is **NULL**.
subpartition_name	character varying(128)	Not supported yet. Its value is **NULL**.
stats_update_time	timestamp(6) with time zone	Time when statistics are updated. Database restart is not supported. Otherwise, data loss will occur.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.63 DB_TABLES

DB_TABLES displays all tables accessible to the current user. This view exists in both PG_CATALOG and SYS schemas.

Table 7-345 DB_TABLES columns

Name	Type	Description
owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.
tablespace_name	character varying(64)	Tablespace name of the table.
dropped	character varying	Specifies whether the current table is deleted. • **yes**: deleted. • **no**: not deleted.
num_rows	numeric	Estimated number of rows in the table.
status	character varying(8)	Specifies whether the current table is valid. • **valid**: The current table is valid. • **unusable**: The current table is unavailable.
sample_size	numeric	Number of samples used for analyzing the table.
temporary	character(1)	Specifies whether the table is a temporary table. • **y**: The table is a temporary table. • **n**: The table is not a temporary table.
pct_free	numeric	Minimum percentage of free space in a block.
ini_trans	numeric	Initial number of transactions.
max_trans	numeric	Maximum number of transactions.
avg_row_len	integer	Average number of bytes in each row.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
partitioned	character varying(3)	Specifies whether a table is a partitioned table. • **yes**: The table is a partitioned table. • **no**: The table is not a partitioned table.
last_analyzed	timestamp with time zone	Last time when the table was analyzed. Database restart is not supported. Otherwise, data loss will occur.
row_movement	character varying(8)	Specifies whether to allow partition row movement. • **enabled**: The partition row movement is allowed. • **disabled**: The partition row movement is not allowed.
compression	character varying(8)	Specifies whether to enable table compression. • **enabled**: Table compression is enabled. • **disabled**: Table compression is disabled.
duration	character varying(15)	Time elapsed when a temporary table is processed. • **NULL**: The table is not a temporary table. • **sys$session**: The table is a temporary session table. • **sys$transaction**: The table is a temporary transaction table.
logical_replication	character varying(8)	Specifies whether logical replication is enabled for a table. • **enabled**: Logical replication is enabled. • **disabled**: Logical replication is disabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
external	character varying(3)	Specifies whether the table is a foreign table. • **yes**: The table is a foreign table. • **no**: The table is not a foreign table.
logging	character varying(3)	Specifies whether to record logs for table changes. • **yes**: Logs are recorded for table changes. • **no**: Logs are not recorded for table changes.
default_collation	character varying(100)	Default collation of a table.
degree	character varying(10)	Number of instances in a scanned table.
table_lock	character varying(8)	Specifies whether to enable a table lock. • **enabled**: The table lock is enabled. • **disabled**: The table lock is disabled.
nested	character varying(3)	Specifies whether a table is a nested table. • **yes**: The table is a nested table. • **no**: The table is not a nested table.
buffer_pool	character varying(7)	Default buffer pool of a table.
flash_cache	character varying(7)	Smart flash cache hint in database for a table block.
cell_flash_cache	character varying(7)	Cell flash cache hint for a table block.
skip_corrupt	character varying(8)	Specifies whether to skip corrupted blocks during table scanning. • **enabled**: The corrupted block is skipped. • **disabled**: The corrupted block is not skipped.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
has_identity	character varying(3)	Specifies whether the table has an identity column. • **yes**: It has an identity column. • **no**: It has no identity column.
inmemory	character varying(8)	Not supported. The value is **disabled**.
segment_created	character varying(3)	Specifies whether a table segment has been created. • **yes**: The table segment has been created. • **no**: The table segment is not created.
monitoring	character varying(3)	Specifies whether to monitor the modification of a table. • **yes**: The modification of the table is monitored. • **no**: The modification of the table is not monitored.
cluster_name	character varying(128)	Not supported yet. Its value is **NULL**.
iot_name	character varying(128)	Not supported yet. Its value is **NULL**.
pct_used	numeric	Not supported yet. Its value is **NULL**.
initial_extent	numeric	Not supported yet. Its value is **NULL**.
next_extent	numeric	Not supported yet. Its value is **NULL**.
min_extents	numeric	Not supported yet. Its value is **NULL**.
max_extents	numeric	Not supported yet. Its value is **NULL**.
pct_increase	numeric	Not supported yet. Its value is **NULL**.
freelists	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
freelist_groups	numeric	Not supported yet. Its value is **NULL**.
backed_up	character varying(1)	Not supported yet. Its value is **NULL**.
blocks	numeric	Not supported yet. Its value is **NULL**.
empty_blocks	numeric	Not supported yet. Its value is **NULL**.
avg_space	numeric	Not supported yet. Its value is **NULL**.
chain_cnt	numeric	Not supported yet. Its value is **NULL**.
avg_space_freelist_blocks	numeric	Not supported yet. Its value is **NULL**.
num_freelist_blocks	numeric	Not supported yet. Its value is **NULL**.
instances	character varying(10)	Not supported yet. Its value is **NULL**.
cache	character varying(5)	Not supported yet. Its value is **NULL**.
iot_type	character varying(12)	Not supported yet. Its value is **NULL**.
secondary	character varying(1)	Not supported yet. Its value is **NULL**.
global_stats	character varying(3)	Not supported yet. Its value is **NULL**.
user_stats	character varying(3)	Not supported yet. Its value is **NULL**.
cluster_owner	character varying(30)	Not supported yet. Its value is **NULL**.
dependencies	character varying(8)	Not supported yet. Its value is **NULL**.
compression_for	character varying(30)	Not supported yet. Its value is **NULL**.
read_only	character varying(3)	Not supported yet. Its value is **NULL**.
result_cache	character varying(7)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
clustering	character varying(3)	Not supported yet. Its value is **NULL**.
activity_tracking	character varying(23)	Not supported yet. Its value is **NULL**.
dml_timestamp	character varying(25)	Not supported yet. Its value is **NULL**.
container_data	character varying(3)	Not supported yet. Its value is **NULL**.
inmemory_priority	character varying(8)	Not supported yet. Its value is **NULL**.
inmemory_distribute	character varying(15)	Not supported yet. Its value is **NULL**.
inmemory_compression	character varying(17)	Not supported yet. Its value is **NULL**.
inmemory_duplicate	character varying(13)	Not supported yet. Its value is **NULL**.
duplicated	character varying(1)	Not supported yet. Its value is **NULL**.
sharded	character varying(1)	Not supported yet. Its value is **NULL**.
hybrid	character varying(3)	Not supported yet. Its value is **NULL**.
cellmemory	character varying(24)	Not supported yet. Its value is **NULL**.
containers_default	character varying(3)	Not supported yet. Its value is **NULL**.
container_map	character varying(3)	Not supported yet. Its value is **NULL**.
extended_data_link	character varying(3)	Not supported yet. Its value is **NULL**.
extended_data_link_map	character varying(3)	Not supported yet. Its value is **NULL**.
inmemory_service	character varying(12)	Not supported yet. Its value is **NULL**.
inmemory_service_name	character varying(1000)	Not supported yet. Its value is **NULL**.
container_map_object	character varying(3)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
memoptimize_read	character varying(8)	Not supported yet. Its value is **NULL**.
memoptimize_write	character varying(8)	Not supported yet. Its value is **NULL**.
has_sensitive_column	character varying(3)	Not supported yet. Its value is **NULL**.
admit_null	character varying(3)	Not supported yet. Its value is **NULL**.
data_link_dml_enabled	character varying(3)	Not supported yet. Its value is **NULL**.
object_id_type	character varying(16)	Not supported yet. Its value is **NULL**.
table_type_owner	character varying(128)	Not supported yet. Its value is **NULL**.
table_type	character varying(128)	Not supported yet. Its value is **NULL**.
compress_for	character varying(30)	Not supported yet. Its value is **NULL**.

7.3.13.64 DB_TRIGGERS

DB_TRIGGERS displays information about triggers accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

Table 7-346 DB_TRIGGERS columns

Name	Type	Description
trigger_name	character varying(64)	Trigger name.
table_name	character varying(64)	Relational table name.
table_owner	character varying(64)	Role name.
status	character varying(64)	Trigger status. • **O**: The trigger is enabled in origin or local mode. • **D**: The trigger is disabled. • **R**: The trigger is enabled in replica mode. • **A**: The trigger is always enabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.65 DB_TYPES

DB_TYPES displays all object types accessible to the current user. All users can access this view. This view exists in the PG_CATALOG and SYS schemas.

Table 7-347 DB_TYPES columns

Name	Type	Description
owner	character varying(128)	Owner of the type.
type_name	character varying(128)	Type name.
type_oid	raw	Type identifier (OID).
typecode	character varying(128)	Type code.
attributes	numeric	Number of attributes in a type.
methods	numeric	Not supported. The value is **0**.
predefined	character varying(3)	Specifies whether the type is a predefined type.
incomplete	character varying(3)	Specifies whether the type is an incomplete type.
final	character varying(3)	Not supported yet. Its value is **NULL**.
instantiable	character varying(3)	Not supported yet. Its value is **NULL**.
persistable	character varying(3)	Not supported yet. Its value is **NULL**.
supertype_owner	character varying(128)	Not supported yet. Its value is **NULL**.
supertype_name	character varying(128)	Not supported yet. Its value is **NULL**.
local_attributes	numeric	Not supported yet. Its value is **NULL**.
local_methods	numeric	Not supported yet. Its value is **NULL**.
typeid	raw	Not supported yet. Its value is **NULL**.

7.3.13.66 DB_VIEWS

DB_VIEWS displays the description about all views accessible to the current user. This view exists in the PG_CATALOG and SYS schemas.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-348 DB_VIEWS columns

Name	Type	Description
owner	name	Owner of the view.
view_name	name	Name of the view.
text	text	Text of the view.
text_length	integer	Text length of the view.
text_vc	character varying(4000)	View creation statement. This column may truncate the view text. The BEQUEATH clause will not appear as part of the **TEXT_VC** column in this view.
type_text_length	numeric	Not supported yet. Its value is **NULL**.
type_text	character varying(4000)	Not supported yet. Its value is **NULL**.
oid_text_length	numeric	Not supported yet. Its value is **NULL**.
oid_text	character varying(4000)	Not supported yet. Its value is **NULL**.
view_type_owner	character varying(128)	Not supported yet. Its value is **NULL**.
view_type	character varying(128)	Not supported yet. Its value is **NULL**.
superview_name	character varying(128)	Not supported yet. Its value is **NULL**.
editioning_view	character varying(1)	Not supported yet. Its value is **NULL**.
read_only	character varying(1)	Not supported yet. Its value is **NULL**.
container_data	character varying(1)	Not supported yet. Its value is **NULL**.
bequeath	character varying(12)	Not supported yet. Its value is **NULL**.
origin_con_id	character varying(256)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
default_collation	character varying(100)	Not supported yet. Its value is **NULL**.
containers_default	character varying(3)	Not supported yet. Its value is **NULL**.
container_map	character varying(3)	Not supported yet. Its value is **NULL**.
extended_data_link	character varying(3)	Not supported yet. Its value is **NULL**.
extended_data_link_map	character varying(3)	Not supported yet. Its value is **NULL**.
has_sensitive_column	character varying(3)	Not supported yet. Its value is **NULL**.
admit_null	character varying(3)	Not supported yet. Its value is **NULL**.
pdb_local_only	character varying(3)	Not supported yet. Its value is **NULL**.

7.3.13.67 DICT

DICT displays the description of data dictionary tables and system views in the database. This view exists in the PG_CATALOG and SYS schemas and all users can access this view.

Table 7-349 DICT columns

Name	Type	Description
table_name	character varying(64)	Object name.
comments	character varying(4000)	Comment on an object.

7.3.13.68 DICTIONARY

DICTIONARY displays the description of data dictionary tables and system views in the database. This view exists in the PG_CATALOG and SYS schemas and all users can access this view.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-350 DICTIONARY columns

Name	Type	Description
table_name	character varying(64)	Object name.
comments	character varying(4000)	Comment on an object.

7.3.13.69 DUAL

The DUAL view is a virtual view created by the system. It has one row and one column and is mainly used to execute queries that do not require actual data tables. It is accessible to all users. This view exists in the PG_CATALOG and SYS schemas.

Table 7-351 DUAL columns

Name	Type	Description
dummy	text	Fixed content "X".

7.3.13.70 DV_SESSIONS

DV_SESSIONS displays information about all active background threads. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized.

Table 7-352 DV_SESSIONS columns

Name	Type	Description
sid	bigint	OID of the active background thread of the current session.
serial#	integer	Sequence number of the background thread of the current activity, which is **0** in GaussDB.
user#	oid	OID of the user logged in to the background thread. The OID is **0** if the background thread is a global auxiliary thread.
username	name	Name of the user logged in to the background process. **username** is null if the background thread is a global auxiliary thread. **application_name** can be identified by associating with **pg_stat_get_activity()**. Example: `SELECT s.*,a.application_name FROM DV_SESSIONS AS s LEFT JOIN pg_stat_get_activity(NULL) AS a ON s.sid=a.sessionid;`

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.71 DV_SESSION_LONGOPS

DV_SESSION_LONGOPS displays the progress of ongoing operations. Users can access this view only after being authorized.

Table 7-353 DV_SESSION_LONGOPS columns

Name	Type	Description
sid	bigint	OID of the running background thread.
serial#	integer	Sequence number of the running background thread, which is **0** in GaussDB.
sofar	integer	Completed workload, which is null in GaussDB.
totalwork	integer	Total workload, which is null in GaussDB.

7.3.13.72 GET_GLOBAL_PREPARED_XACTS

GET_GLOBAL_PREPARED_XACTS displays prepared transactions on all nodes globally.

Table 7-354 GET_GLOBAL_PREPARED_XACTS columns

Name	Type	Description
transaction	xid	XID of a prepared transaction
gid	text	GID of the prepared transaction
prepared	timestamp with time zone	Prepared time of the prepared transaction.
owner	name	Owner of the prepared transaction.
database	name	Database to which the prepared transaction belongs.
node_name	text	Name of the node where the prepared transaction resides

7.3.13.73 GLOBAL_BAD_BLOCK_INFO

GLOBAL_BAD_BLOCK_INFO is executed on the CN to collect statistics on damaged data pages of all instances. The queried result displays basic information about damaged pages. The execution result on the DN is empty. Based on the information, you can use the page detection and repair function in **Data Damage Detection and Repair Functions** to perform further repair operations. By default, only the initial user, users with the SYSADMIN permission, users with the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

OPRADMIN permission in the O&M mode, and monitoring users can view the information. Other users can view the information only after being granted permissions.

Table 7-355 GLOBAL_BAD_BLOCK_INFO columns

Name	Type	Description
node_name	text	Node information of the damaged page.
spc_node	oid	ID of the tablespace corresponding to the damaged page.
db_node	oid	ID of the database corresponding to the damaged page.
rel_node	oid	relfilenode of the relation corresponding to the damaged page.
bucket_node	integer	Bucket node of the damaged page. This column is used to specify whether a table is a segment-page table during repair. ● **-1**: non-segment page table. ● Non-zero value: segment-page table.
block_num	oid	Number of the damaged page.
fork_num	integer	File forknum of the damaged page.
file_path	text	Relative path of the current damaged page. The logical path instead of the actual file is displayed for a segment-page table.
check_time	timestamp with time zone	Time when a fault is detected on the damaged page.
repair_time	timestamp with time zone	Time when the damaged page is repaired.

7.3.13.74 GLOBAL_CLEAR_BAD_BLOCK_INFO

GLOBAL_CLEAR_BAD_BLOCK_INFO is executed on the CN to clear information about repaired pages in all instances. The execution result on DNs is empty. By default, only the initial user, users with the SYSADMIN permission, users with the OPRADMIN permission in the O&M mode, and monitoring users can view the information. Other users can view the information only after being granted permissions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-356 GLOBAL_BAD_BLOCK_INFO columns

Name	Type	Description
node_name	text	Information about the node for clearing repaired pages.
result	boolean	Execution result of clearing repaired pages of the current instance.

7.3.13.75 GLOBAL_SQL_PATCH

GLOBAL_SQL_PATCH stores information about all SQL patches. This view is available only in the PG_CATALOG schema.

Table 7-357 GLOBAL_SQL_PATCH columns

Name	Type	Description
node_name	text	Name of the node where the SQL PATCH is located.
patch_name	name	Name of the SQL PATCH.
unique_sql_id	bigint	Global unique query ID.
owner	oid	ID of the user created using the SQL PATCH.
enable	Boolean	Specifies whether the SQL patch takes effect.
status	"char"	SQL patch status (reserved column).
abort	Boolean	Specifies whether a hint is discarded.
hint_string	text	Hint text.
description	text	SQL patch description.
parent_unique_sql_id	bigint	Globally unique ID of the outer statement of the SQL statement for which the SQL patch takes effect. • Its value is **0** for statements outside the stored procedure. • For a statement within a stored procedure, the value is the globally unique ID of the statement that calls the stored procedure.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
original_query	text	SQL text.

7.3.13.76 GLOBAL_STAT_HOTKEYS_INFO

GLOBAL_STAT_HOTKEYS_INFO queries the statistics of hotspot keys in the entire cluster. The query results are sorted by **count** in descending order.

Table 7-358 GLOBAL_STAT_HOTKEYS_INFO columns

Name	Type	Description
database_name	text	Name of the database where the hot key is located.
schema_name	text	Name of the mode where the hotspot key is located.
table_name	text	Name of the table where the hotspot key is located.
key_value	text	Value of the hotspot key
hash_value	bigint	Hash value of the hot key in the database. For a list or range distribution table, the value of this column is **0**.
count	numeric	Frequency of accessing the hotspot key.

7.3.13.77 GLOBAL_WAL_SENDER_STATUS

GLOBAL_WAL_SENDER_STATUS displays the redo log transfer and replay status of the primary DN in the current cluster. Only users with the SYSADMIN or MONADMIN permission can query this view.

Table 7-359 GLOBAL_WAL_SENDER_STATUS columns

Name	Type	Description
nodename	text	Name of the primary node
source_ip	text	IP address of the primary node
source_port	integer	Port of the primary node
dest_ip	text	IP address of the standby node
dest_port	integer	Port of the standby node
sender_pid	integer	PID of the sending thread

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
local_role	text	Type of the primary node • **UNKNOWN_MODE**: The status is unknown. • **NORMAL_MODE**: single-host node type. • **PRIMARY_MODE**: The node is a primary node. • **STANDBY_MODE**: The node is a standby node. • **CASCADE_STANDBY_MODE**: The node is a cascaded standby node. • **PENDING_MODE**: The node is in the arbitration phase. • **RECOVERY_MODE**: The node is in the restoration phase. • **STANDBY_CLUSTER_MODE**: The node is a standby cluster node. • **MAIN_STANDBY_MODE**: main standby cluster node. **NOTE** The expected values of the primary node type are **NORMAL_MODE**, **PRIMARY_MODE**, **PENDING_MODE**, and **RECOVERY_MODE**. If other node types are displayed, contact Huawei technical support.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
peer_role	text	Type of the standby node • **UNKNOWN_MODE**: The status is unknown. • **NORMAL_MODE**: single-host node type. • **PRIMARY_MODE**: The node is a primary node. • **STANDBY_MODE**: The node is a standby node. • **CASCADE_STANDBY_MODE**: The node is a cascaded standby node. • **PENDING_MODE**: The node is in the arbitration phase. • **RECOVERY_MODE**: The node is in the restoration phase. • **STANDBY_CLUSTER_MODE**: The node is a standby cluster node. • **MAIN_STANDBY_MODE**: main standby cluster node. **NOTE** The expected values of the standby node type are **NORMAL_MODE**, **STANDBY_MODE**, **CASCADE_STANDBY_MODE**, **PENDING_MODE**, **RECOVERY_MODE**, **STANDBY_CLUSTER_MODE**, and **MAIN_STANDBY_MODE**. If other node types are displayed, contact Huawei technical support.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
peer_state	text	Status of the standby node • **UNKNOWN_STATE**: The node status is unknown. • **NORMAL_STATE**: The node is started properly. • **NEEDREPAIR_STATE**: The current node needs to be repaired. • **STARTING_STATE**: The node is being started. • **WAITING_STATE**: The node is waiting to be upgraded. • **DEMOTING_STATE**: The node is being downgraded. • **PROMOTING_STATE**: The standby node is being upgraded to the primary node. • **BUILDING_STATE**: The standby node fails to be started and needs to be rebuilt. • **CATCHUP_STATE**: The standby node is catching up with the primary node. • **COREDUMP_STATE**: The node breaks down.
state	text	WAL sender status • **WALSNDSTATE_STARTUP**: startup state. • **WALSNDSTATE_BACKUP**: backup state. • **WALSNDSTATE_CATCHUP**: catch-up state. • **WALSNDSTATE_STREAMING**: streaming replication state.
sender_sent_location	text	Sending position of the primary node
sender_write_location	text	Writing position of the primary node
sender_flush_location	text	Location where the master node writes the cache to the disk.
sender_replay_location	text	Redo position of the primary node
receiver_received_location	text	Receiving position of the standby node
receiver_write_location	text	Writing position of the standby node

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
receiver_flush_location	text	Location where the standby node writes the cache to the disk.
receiver_replay_location	text	Redo location of the standby node

7.3.13.78 GS_ALL_CONTROL_GROUP_INFO

GS_ALL_CONTROL_GROUP_INFO displays all Cgroup information in a database.

Table 7-360 GS_ALL_CONTROL_GROUP_INFO columns

Name	Type	Description
name	text	Cgroup name.
type	text	Cgroup type. • **GROUP_NONE**: no group. • **GROUP_TOP**: top group. • **GROUP_CLASS**: Class Cgroup, which does not control any thread. • **GROUP_BAKWD**: backend thread Cgroup. • **GROUP_DEFWD**: default Cgroup, which controls only the query threads at this level. • **GROUP_TSWD**: time-sharing Cgroup of each user, which controls the query threads at the bottom layer.
gid	bigint	Cgroup ID.
classgid	bigint	ID of the **Class** Cgroup to which a **Workload** Cgroup belongs.
class	text	**Class** Cgroup.
workload	text	Workload Cgroup.
shares	bigint	CPU quota allocated to the Cgroup.
limits	bigint	Limit of CPUs allocated to the Cgroup.
wdlevel	bigint	**Workload** Cgroup level.
cpucores	text	Usage of CPU cores in the Cgroup.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.79 GS_ALL_PREPARED_STATEMENTS

GS_ALL_PREPARED_STATEMENTS displays information about prepared statements that are available in all sessions. By default, this view is accessible only to SYSADMINs.

Table 7-361 GS_ALL_PREPARED_STATEMENTS columns

Name	Type	Description
pid	bigint	Background thread ID. **NOTE** In thread pool mode, **pid** indicates the ID of the thread bound to the current session. When a session is executed on different threads, **pid** changes accordingly. In thread pool mode, statements are associated with **sessionid** but not **pid**. You are advised to use **sessionid** for join query.
sessionid	bigint	Current session ID.
global_sessionid	text	Global session ID.
name	text	Prepared statement identifier.
statement	text	Query string for creating this prepared statement. For prepared statements created through SQL, this is the PREPARE statement submitted by the client. For prepared statements created through the frontend/backend protocol, this is the text of the prepared statement itself. For the plan cache generated by stream PBE, CNs record the preceding content, whereas DNs record the statement executed for the first time.
prepare_time	timestamp with time zone	Timestamp when the prepared statement is created.
parameter_types	regtype[]	Expected parameter type of the prepared statement in the form of a regtype array. The OID corresponding to this array can be obtained by converting the regtype value. For the plan cache generated by stream PBE, DNs leave this column blank.
from_sql	boolean	• **True** if the prepared statement was created through the PREPARE statement. • **False** if the statement was prepared through the frontend/backend protocol.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
is_deleted	boolean	Specifies whether the plan cache is cleared. The GUC parameter **max_plancache_memory_ratio** is used to set the clearance threshold. • **TRUE** if the plan cache is cleared. • **FALSE** if the plan cache is not cleared.
size	bigint	Size of the memory occupied by the plan cache, in bytes.
valid	boolean	Expiration status of the plan cache. • **TRUE** if the status is valid. • **FALSE** if the status is invalidated.
is_reused	boolean	Specifies whether the cache is reused. • **TRUE** if the cache is reused. • **FALSE** if the cache is not reused.

7.3.13.80 GS_AUTONOMOUS_TRANSACTION_DETAIL

GS_AUTONOMOUS_TRANSACTION_DETAIL displays the autonomous transaction session to which the user belongs and the relationships of parent transactions and sub-transactions in all sessions on the current CN. If the current user has the MONADMIN or SYSADMIN permission, they can view all autonomous transaction sessions and the relationships of parent transactions and sub-transactions on the current CN. Calling on DNs is not allowed.

Table 7-362 GS_AUTONOMOUS_TRANSACTION_DETAIL columns

Name	Type	Description
autonomous_transaction_session_id	bigint	Session ID of an autonomous transaction.
autonomous_transaction_thread_id	bigint	Thread ID of an autonomous transaction.
parent_autonomous_transaction_session_id	bigint	Parent session ID of an autonomous transaction.

7.3.13.81 GS_BUCKET_LOCKS

GS_BUCKET_LOCKS displays information about locks held by open transactions. Only CNs in distributed mode can be queried.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-363 GS_BUCKET_LOCKS columns

Name	Type	Reference	Description
node_name	text	-	Node name.
database	oid	**oid** in **PG_DATABASE**	OID of the database in which the locked target exists.
virtualtransaction	text	-	Virtual ID of the transaction holding or awaiting this lock.
pid	bigint	-	Logical ID of the server thread holding or awaiting this lock. The value is **NULL** if the lock is held by a prepared transaction.
sessionid	bigint	-	ID of the session that holds or waits for the lock.
mode	text	-	Lock mode held or desired by this thread. The options are **AccessShareLock** and **AccessExclusiveLock**.
granted	boolean	-	• The value is **TRUE** if the lock is a held lock. • The value is **FALSE** if the lock is an awaited lock.
fastpath	boolean	-	The value is **TRUE** if the lock is obtained through fast-path. The value is **FALSE** if the lock is obtained through database-level lock information or two-phase commit.
global_sessionid	text	-	Global session ID.
bucketlist	text	-	List of locked buckets.
bucket_group	smallint	-	Bucket group to which the locked bucket belongs. The value is fixed to **0** and is not supported in the current version.

7.3.13.82 GS_GET_CONTROL_GROUP_INFO

GS_GET_CONTROL_GROUP_INFO displays information about all Cgroups. By default, only users with the SYSADMIN permission can access this view.

Table 7-364 GS_GET_CONTROL_GROUP_INFO columns

Name	Type	Description
group_name	text	Name of a Cgroup.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
group_type	text	Type of the Cgroup. • **GROUP_NONE**: no group. • **GROUP_TOP**: top group. • **GROUP_CLASS**: class Cgroup, which does not control any thread. • **GROUP_BAKWD**: backend thread Cgroup. • **GROUP_DEFWD**: default Cgroup, which controls only the query threads at this level. • **GROUP_TSWD**: time-sharing Cgroup of each user, which controls the query threads at the bottom layer.
gid	bigint	Cgroup ID.
classgid	bigint	ID of the **Class** Cgroup to which a **Workload** Cgroup belongs.
class	text	**Class** Cgroup.
group_workload	text	Workload Cgroup.
shares	bigint	CPU quota allocated to the Cgroup.
limits	bigint	Limit of CPUs allocated to the Cgroup.
wdlevel	bigint	Workload Cgroup level.
cpucores	text	Usage of CPU cores in the Cgroup.
nodegroup	text	Node group name.
group_kind	text	Node group type. The options are as follows: • **i**: installation node group • **n**: node group in a common cluster. • **e**: elastic cluster

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.83 GS_GLC_MEMORY_DETAIL

GS_GLC_MEMORY_DETAIL displays the memory usage of the global PL/pgSQL cache in all databases. This view is available only when **enable_global_plsqlcache** is set to **on**.

Table 7-365 GS_GLC_MEMORY_DETAIL columns

Name	Type	Description
contextname	text	Name of a memory object.
database	text	Database to which the memory object belongs. **"pkg_bucket"** and **"func_bucket"** are displayed as **"NULL"**.
schema	text	Schema to which the memory object belongs. **"pkg_bucket"** and **"func_bucket"** are displayed as **"NULL"**.
type	text	Object type: ● **"pkg_bucket"**: indicates that the object is the parent node of a package object. ● **"func_bucket"**: indicates that the object is the parent node of a function or stored procedure. ● **"pkg"**: indicates that the object is a package object. ● **"func"**: indicates that the object is a function or stored procedure.
status	text	Status of the cached object. **"valid"** indicates that the cached object is available, and **'invalid'** indicates that the cached object is unavailable. The **"pkg_bucket"** and **"func_bucket"** objects have no status and are displayed as **"NULL"**.
location	text	Current location of a cached object. It is displayed as **"in_global_hash_table"** in the cache hash table and **"in_global_expired_list"** in the invalid linked list. The **"pkg_bucket"** and **"func_bucket"** objects are displayed as **"NULL"**.
env	bigint	Environment parameter when an object is created, that is, the value of **behavior_compat_flags**. The **"pkg_bucket"** and **"func_bucket"** objects are displayed as **0**.
usedsize	bigint	Size of a cached object.
usecount	bigint	Number of objects that are referencing the global cache. When no object references the global compilation product, the number is 0.
oid	integer	OID of a memory object.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
isspec	boolean	Specifies whether the memory object is a packet header.
searchpath	text	The memory object accesses the schema and other environment variables of the compilation product.
invalidreason	text	Reason why a memory object becomes invalid.

7.3.13.84 GS_GLOBAL_ARCHIVE_STATUS

GS_GLOBAL_ARCHIVE_STATUS describes the archiving progress of CNs and all shards, including the shard name (**node_name**), archiving location (**restart_lsn**), name of the primary or standby node (**archive_node**), and current log location (**current_xlog_location**). To query this view, you need to enable the archiving functionality of the database and query the view from the CN. Only users with the SYSADMIN or MONADMIN permission can query this view. For details about the columns, see **Table 7-366**.

Table 7-366 GS_GLOBAL_ARCHIVE_STATUS columns

Name	Type	Description
node_name	text	Shard name.
restart_lsn	text	Archiving location.
archive_node	text	Name of the primary or standby node where archiving is performed.
current_xlog_location	text	Current log position.

7.3.13.85 GS_GSC_MEMORY_DETAIL

GS_GSC_MEMORY_DETAIL displays the global SysCache memory usage of the current process on the current node. The data is displayed only when Global SysCache is enabled. For details about the columns, see **Table 7-367**.

The query is separated by the database memory context. Therefore, some memory statistics are missing. The memory context corresponding to the missing memory statistics is **GlobalSysDBCache**.

Table 7-367 GS_GSC_MEMORY_DETAIL columns

Name	Type	Description
db_id	text	Database ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
totalsize	numeric	Total size of the shared memory, in bytes.
freesize	numeric	Remaining size of the shared memory, in bytes.
usedsize	numeric	Used size of the shared memory, in bytes.

7.3.13.86 GS_GTT_ATTACHED_PIDS

GS_GTT_ATTACHED_PIDS is used to check which sessions are using global temporary tables on DNs by calling the gs_gtt_attached_pid() function.

Table 7-368 GS_GTT_ATTACHED_PIDS columns

Name	Type	Description
schemaname	name	Schema name.
tablename	name	Name of a global temporary table.
relid	oid	OID of a global temporary table.
pids	bigint[]	Session ID.
cn_session_ids	text[]	List of global session IDs on the CN in distributed mode.

7.3.13.87 GS_GLOBAL_AUTONOMOUS_TRANSACTION_DETAIL

GS_GLOBAL_AUTONOMOUS_TRANSACTION_DETAIL displays the autonomous transaction session to which the user belongs and the relationships of parent transactions and sub-transactions in all sessions on each CN. If the current user has the MONADMIN or SYSADMIN permission, they can view all autonomous transaction sessions and the relationships of parent transactions and sub-transactions on each CN. Calling on DNs is not allowed.

NOTE

When the GUC parameter **enable_nonsysadmin_execute_direct** is set to **off**, this view can be called only by users with the MONADMIN or SYSADMIN permissions.

Table 7-369 GS_GLOBAL_AUTONOMOUS_TRANSACTION_DETAIL columns

Name	Type	Description
node_name	text	CN name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
autonomous_transaction_session_id	bigint	Session ID of an autonomous transaction.
autonomous_transaction_thread_id	bigint	Thread ID of an autonomous transaction.
parent_autonomous_transaction_session_id	bigint	Parent session ID of an autonomous transaction.

7.3.13.88 GS_GLOBAL_GTT_ATTACHED_PIDS

GS_GLOBAL_GTT_ATTACHED_PIDS displays which sessions are using global temporary tables on DNs on a CN.

Table 7-370 GS_GLOBAL_GTT_ATTACHED_PIDS columns

Name	Type	Description
node_name	text	Node name.
schemaname	name	Schema name.
tablename	name	Name of a global temporary table.
relid	oid	OID of a global temporary table.
pids	bigint[]	Session ID.
cn_session_ids	text[]	List of global session IDs on the CN in distributed mode.

7.3.13.89 GS_GLOBAL_DIST_GTT_REMAIN_SESSIONS

GS_GLOBAL_DIST_GTT_REMAIN_SESSIONS displays all session information that remains on DNs on a CN.

Table 7-371 GS_GTT_ATTACHED_PIDS columns

Name	Type	Description
dn_name	text	DN name.
cn_name	text	CN name.
cn_timeline	integer	Timeline of the CN.
cn_sequence	integer	Session sequence of the CN.
cn_session_id	bigint	Session ID of the CN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.90 GS_LOGICALSTANDBY_SKIP

GS_LOGICALSTANDBY_SKIP displays the defined skip rules. When **skiptype** is set to **1** or **3** (for details, see **GS_SQLAPPLY_SKIP**), the following columns are obtained from the system catalog GS_SQLAPPLY_SKIP to create or update the system view. The view is empty and reserved in distributed scenarios.

Table 7-372 GS_LOGICALSTANDBY_SKIP columns

Name	Type	Description
createtime	timestamp with time zone	SKIP definition time.
error	text	Specifies whether it is triggered only if an error occurs.
dump	"char"	Dump type.
stmt	text	SKIP statement or KEYWORD.
dbname	text	Database name.
schemaname	text	Schema name.
objectname	text	Object name.

7.3.13.91 GS_LOGICALSTANDBY_SKIP_TRANSACTION

GS_LOGICALSTANDBY_SKIP_TRANSACTION displays the defined SKIP rules. When **skiptype** is set to **2** (for details, see **GS_SQLAPPLY_SKIP**), the following columns are obtained from the system catalog GS_SQLAPPLY_SKIP to create or update the system view. The view is empty and reserved in distributed scenarios.

Table 7-373 GS_LOGICALSTANDBY_SKIP_TRANSACTION columns

Name	Type	Attribute
createtime	timestamp with time zone	SKIP definition time.
dump	"char"	Dump type.
commitcsn	text	Transaction CSN.
xid	text	Transaction ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.92 GS_LSC_MEMORY_DETAIL

GS_LSC_MEMORY_DETAIL displays the memory usage of the local system cache of all threads based on the MemoryContext node. The data is displayed only when Global SysCache is enabled. For details about the columns, see **Table 7-374**.

Table 7-374 GS_LSC_MEMORY_DETAIL columns

Name	Type	Description
threadid	text	Thread start time + thread ID (string: *timestamp.sessionid*).
tid	bigint	Thread ID.
thrdtype	text	Thread type. It can be any thread type in the system, such as postgresql and wlmmonitor.
contextname	text	Name of the memory context.
level	smallint	Hierarchy of the memory context.
parent	text	Name of the parent memory context.
totalsize	bigint	Total size of the memory context, in bytes.
freesize	bigint	Total size of released memory in the memory context, in bytes.
usedsize	bigint	Total size of used memory in the memory context, in bytes.

7.3.13.93 GS_MY_PLAN_TRACE

GS_MY_PLAN_TRACE is a view of the GS_PLAN_TRACE system catalog. This view displays the plan traces of the current user. For details about the columns, see **Table 7-375**. The plan trace feature is not supported in the distributed scenario. Therefore, no data is displayed in this view in the distributed scenario.

Table 7-375 GS_MY_PLAN_TRACE columns

Name	Type	Description
query_id	text	Unique ID of the current request.
query	text	SQL statement of the current request. The value of this column cannot exceed the value of **track_activity_query_size**.
unique_sql_id	bigint	Unique ID of the SQL statement of the current request.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
plan	text	Query plan text corresponding to the SQL statement of the current request. The size of this column cannot exceed 10 KB.
plan_trace	text	Details about the query plan generation process corresponding to the SQL statement of the current request. The value of this column cannot exceed 300 MB.
modifydate	timestamp with time zone	Time when the current plan trace is updated (that is, time when the plan trace is created).

7.3.13.94 GS_SESSION_ALL_SETTINGS

GS_SESSION_ALL_SETTINGS displays the full GUC parameter settings of all sessions on the local node. Only SYSADMIN and MONADMIN can query this view. For details about the columns, see **Table 7-376**.

Table 7-376 GS_SESSION_ALL_SETTINGS columns

Name	Type	Description
sessionid	bigint	Session ID.
pid	bigint	Backend thread ID.
name	text	Parameter name.
setting	text	Current parameter value.
unit	text	Implicit unit of a parameter.

7.3.13.95 GS_SQL_COUNT

GS_SQL_COUNT displays statistics about five types of running statements (SELECT, INSERT, UPDATE, DELETE, and MERGE INTO) on the current node of the database. For details about the columns, see **Table 7-377**.

- When a common user queries the GS_SQL_COUNT view, statistics about the current node of the user are displayed. When an administrator queries the GS_SQL_COUNT view, statistics about the current node of all users are displayed.
- When the cluster or node is restarted, the count is cleared and then restarted.
- The system counts when a node receives a query, including a query inside the cluster. For example, when a CN receives a query and distributes multiple queries to DNs, the queries are counted accordingly on the DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-377 GS_SQL_COUNT columns

Name	Type	Description
node_name	name	Node name.
user_name	name	Username.
select_count	bigint	Statistical result of the SELECT statement.
update_count	bigint	Statistical result of the UPDATE statement.
insert_count	bigint	Statistical result of the INSERT statement.
delete_count	bigint	Statistical result of the DELETE statement.
mergeinto_count	bigint	Statistical result of the MERGE INTO statement.
ddl_count	bigint	Number of DDL statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including TRUNCATE, CLUSTER, VACUUM, ANALYZE, CREATE (excluding CREATE ROLE and CREATE USER), DROP (excluding DROP ROLE and DROP USER), ALTER (excluding ALTER ROLE, ALTER USER, ALTER DEFAULT PRIVILEGES, and ALTER SESSION), REINDEX, COMMENT, BARRIER, SECURITY LABEL ON, PURGE, TIMECAPSULE, REFRESH, IMPDP, and EXPDP statements.
dml_count	bigint	Number of DML statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including SELECT, UPDATE, INSERT, DELETE, MERGE, PREPARE, EXPLAIN, LOCK, COPY, ANONYMOUS, DELTA, EXECUTE, MOVE, FETCH, CLOSE, DEALLOCATE, DECLARE, CURSOR, and LOAD statements.
dcl_count	bigint	Number of DCL statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including SHOW, CREATE ROLE, CREATE USER, DROP ROLE, DROP USER, ALTER ROLE, ALTER USER, ALTER DEFAULT PRIVILEGES, GRANT, REVOKE, REASSIGN, SET, RESET, and ALTER SESSION statements.
total_select_elapse	bigint	Total response time of SELECT statements (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
avg_select_elapse	bigint	Average response time of SELECT statements (unit: μs).
max_select_elapse	bigint	Maximum response time of SELECT statements (unit: μs).
min_select_elapse	bigint	Minimum response time of SELECT statements (unit: μs).
total_update_elapse	bigint	Total response time of UPDATE statements (unit: μs).
avg_update_elapse	bigint	Average response time of UPDATE statements (unit: μs).
max_update_elapse	bigint	Maximum response time of UPDATE statements (unit: μs).
min_update_elapse	bigint	Minimum response time of UPDATE statements (unit: μs).
total_insert_elapse	bigint	Total response time of INSERT statements (unit: μs).
avg_insert_elapse	bigint	Average response time of INSERT statements (unit: μs).
max_insert_elapse	bigint	Maximum response time of INSERT statements (unit: μs).
min_insert_elapse	bigint	Minimum response time of INSERT statements (unit: μs).
total_delete_elapse	bigint	Total response time of DELETE statements (unit: μs).
avg_delete_elapse	bigint	Average response time of DELETE statements (unit: μs).
max_delete_elapse	bigint	Maximum response time of DELETE statements (unit: μs).
min_delete_elapse	bigint	Minimum response time of DELETE statements (unit: μs).
dbid	oid	ID of the database to which the five types of statements (SELECT, INSERT, UPDATE, DELETE, and MERGE INTO) belong.
user_dml_count	bigint	Number of DML statements executed by users. The value of **user_dml_count** is equal to the value of **dml_count** minus the value of **bg_dml_count**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
bg_dml_count	bigint	Number of DML statements executed by background threads (excluding WORKER and THREADPOOL_WORKER) in the database in all scenarios. **NOTE** The DML statements executed by the current stored procedure cannot be counted.

7.3.13.96 GS_SYSTEM_OBJECT_VERSION_DETAILS

GS_SYSTEM_OBJECT_VERSION_DETAILS displays the version information about system objects and details about the corresponding system objects.

The system object types in this system view include views, functions, tables, indexes, types, rules, and type casting relationships (CAST).

Table 7-378 GS_SYSTEM_OBJECTS_VERSION columns

Name	Type	Description
object_oid	oid	Identifier of a system object. For different types of system objects, the relationships are as follows: Unique identifier of a view, table, or index in **PG_CLASS**, or unique identifier of a type, function, rule, or type cast in **PG_TYPE**, **PG_PROC**, **PG_REWRITE**, or **PG_CAST**.
object_namespace	oid	OID of the schema of the system object. If the system object is a type cast, this column is empty.
object_name	character varying	Name of the system object. If the system object is a type cast, the value of this column is in the format of "*source data type* (*OID of the source data type*) AS *target data type* (*OID of the target data type*)".
object_schema	name	Name of the schema of the system object.
object_args	text	If the system object is a function, this column is the data type array of the function parameters, including only input parameters (including **INOUT** parameters). The display format is "[arg1,..., argn]". If the system object is of another type, this column is empty.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
object_type	text	Type of a system object. • **view** • **function** • **table**: ordinary table. • **index** • **composite type** • **domain**: domain type. • **rule** • **cast**: type cast.
object_version	bigint	Version number of the system object. **0**: initial version number. **-1**: version number of a deleted object. Other version numbers are used for subsequent object changes.

7.3.13.97 GS_THRDSTREAM_WORKER_INFO

GS_THRDSTREAM_WORKER_INFO displays information about stream thread pool reuse. The query can be performed only on DNs running in thread pool mode. Only users with the PUBLIC permission can access this view.

Table 7-379 GS_THRDSTREAM_WORKER_INFO columns

Name	Type	Description
node_name	text	Instance name.
group_id	integer	Group ID of the thread pool.
db_info	text	Dynamic database statistics about the running stream thread pool of the current group. The information includes: • **total stream reusing db**: number of databases that are being used or reused in the stream thread pool. • **free session db**: number of databases that do not have running sessions in the stream thread pool.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
stream_info	text	Dynamic stream thread statistics about the running stream thread pool of the current group. The information includes: • **actual**: number of actually started stream threads. • **idle**: number of idle stream threads. • **using**: number of stream threads in use. • **init session threads**: number of stream threads for initializing a session. • **reuse session threads**: number of stream threads that reuse sessions.

7.3.13.98 GS_TOTAL_NODEGROUP_MEMORY_DETAIL

GS_TOTAL_NODEGROUP_MEMORY_DETAIL returns the memory usage (in MB) of the current node group of the database. If the GUC parameter **enable_memory_limit** is set to **off**, this view is unavailable. For details about the columns, see **Table 7-380**.

Table 7-380 GS_TOTAL_NODEGROUP_MEMORY_DETAIL columns

Name	Type	Description
ngname	text	Node group name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
memorytype	text	Memory type. The value must be one of the following: • **ng_total_memory**: total memory size of the node group. • **ng_used_memory**: memory used by the node group. • **ng_estimate_memory**: estimated memory used by the node group. • **ng_foreignrp_memsize**: total memory of the external resource pool of the node group. • **ng_foreignrp_usedsize**: memory used by the external resource pool of the node group. • **ng_foreignrp_peaksize**: peak memory used by the external resource pool of the node group. • **ng_foreignrp_mempct**: percentage of the external resource pool of the node group to the total memory of the node group. • **ng_foreignrp_estmsize**: estimated memory used by the external resource pool of the node group.
memorymbytes	integer	Size of allocated memory-typed memory.

7.3.13.99 GS_WORKLOAD_RULE_STAT

GS_WORKLOAD_RULE_STAT displays information about SQL concurrency control rules. Only the SYSADMIN can access the system view. For details about the columns, see **Table 7-381**.

Table 7-381 GS_WORKLOAD_RULE_STAT columns

Name	Type	Description
rule_id	bigint	Concurrency control rule ID.
rule_name	name	Name of a concurrency control rule, which is used to search for the concurrency control rule.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
databases	name[]	List of databases on which the concurrency control rules take effect. If the value is **NULL**, the concurrency control rules take effect for all databases.
rule_type	text	Concurrency control rule type. Currently, only **"sqlid"**, **"select"**, **"insert"**, **"update"**, **"delete"**, **"merge"**, **"resource"**, and **"client"** are supported. Other values are invalid.
start_time	timestamp with time zone	Start time of the concurrency control rules. The value **NULL** indicates that the rules take effect from now on.
end_time	timestamp with time zone	End time of the concurrency control rules. The value **NULL** indicates that the rules are always effective.
max_workload	bigint	Maximum number of concurrent rule settings.
option_val	text[]	Parameter values of a concurrency control rule, including the SQL IDs, keyword lists, resource limitations, client IP addresses, and application names. For details, see **gs_add_workload_rule(rule_type, rule_name, databases, start_time, end_time, max_workload, option_val)**.
is_valid	Boolean	Determines whether a concurrency control rule takes effect. If it times out, the value is set to **false**.
validate_count	bigint	Number of SQL statements intercepted by the concurrency control rule.
node_names	text[]	List of nodes on which the concurrency control rules take effect. This parameter is reserved and does not take effect currently.
user_names	text[]	List of users for which the concurrency control rules take effect. This parameter is reserved and does not take effect currently.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.100 GV_INSTANCE

GV_INSTANCE displays information about the current database instance. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-382**.

Table 7-382 GV_INSTANCE columns

Name	Type	Description
inst_id	oid	OID of the current database.
instance_number	oid	OID of the current database.
instance_name	character varying(16)	Name of the current database.
host_name	character varying(64)	Not supported yet. Its value is **NULL**.
version	character varying(17)	Not supported yet. Its value is **NULL**.
version_legacy	character varying(17)	Not supported yet. Its value is **NULL**.
version_full	character varying(17)	Not supported yet. Its value is **NULL**.
startup_time	date	Not supported yet. Its value is **NULL**.
status	character varying(12)	Not supported yet. Its value is **NULL**.
parallel	character varying(3)	Not supported yet. Its value is **NULL**.
thread#	numeric	Not supported yet. Its value is **NULL**.
archiver	character varying(7)	Not supported yet. Its value is **NULL**.
log_switch_wait	character varying(15)	Not supported yet. Its value is **NULL**.
logins	character varying(10)	Not supported yet. Its value is **NULL**.
shutdown_pending	character varying(3)	Not supported yet. Its value is **NULL**.
database_status	character varying(17)	Not supported yet. Its value is **NULL**.
instance_role	character varying(18)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
active_state	character varying(9)	Not supported yet. Its value is **NULL**.
blocked	character varying(3)	Not supported yet. Its value is **NULL**.
con_id	numeric	Not supported yet. Its value is **NULL**.
instance_mode	character varying(11)	Not supported yet. Its value is **NULL**.
edition	character varying(7)	Not supported yet. Its value is **NULL**.
family	character varying(80)	Not supported yet. Its value is **NULL**.
database_type	character varying(15)	Not supported yet. Its value is **NULL**.

7.3.13.101 GV_SESSION

GV_SESSION displays information about all current sessions. Only administrators can access this view. Common users can access the view only after being authorized. This view exists in both PG_CATALOG and SYS schemas. When the thread pool is enabled (**enable_thread_pool** is set to **on**), all session information is displayed. When the thread pool is disabled (**enable_thread_pool** is set to **off**), the sessions connected by users are not displayed. For details about the columns, see **Table 7-383**.

Table 7-383 GV_SESSION columns

Name	Type	Description
inst_id	numeric	Not supported. Set it to **NULL**.
saddr	raw	Not supported. Set it to **NULL**.
sid	bigint	Session ID.
serial#	integer	Sequence number of the active background thread, which is **0** in GaussDB.
audsid	numeric	Not supported. Set it to **NULL**.
paddr	raw	Not supported. Set it to **NULL**.
schema#	numeric	Not supported. Set it to **NULL**.
schemaname	name	Name of the user logged in to the background.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
user#	oid	OID of the user logged in to the background thread. (**0** if the background thread is a global auxiliary thread).
username	name	Name of the user logged in to the background process. **username** is empty if the background thread is a global auxiliary thread.
command	numeric	Not supported. Set it to **NULL**.
ownerid	numeric	Not supported. Set it to **NULL**.
taddr	character varying(16)	Not supported. Set it to **NULL**.
lockwait	character varying(16)	Not supported. Set it to **NULL**.
machine	text	Host name of the connected client, as reported by a reverse DNS lookup of **client_addr**. This column will be non-null only for IP connections and only when **log_hostname** is enabled.
sql_id	bigint	ID of a query.
client_info	text	Client information
event	text	Queuing status of a statement. The value can be: • **waiting in queue**: The statement is in the queue. • **Empty**: The statement is running.
sql_exec_start	timestamp with time zone	Time when the currently active query was started, or time when the last query was started if the value of **state** is not **active**.
program	text	Name of the application connected to the background.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
status	text	Overall status of this background. The value can be: • **active**: The background is executing a query. • **idle**: The background is waiting for a new client command. • **idle in transaction**: The background is in a transaction, but there is no statement being executed in the transaction. • **idle in transaction (aborted)**: The background is in a transaction, but there are statements failed in the transaction. • **fastpath function call**: The background is executing a fast-path function. • **disabled**: This state is reported if **track_activities** is disabled in this background.
server	character varying(9)	Not supported. Set it to **NULL**.
pdml_status	character varying(8)	Specifies whether to enable a DML parallel execution in the current session.
port	numeric	Port number of the current session.
process	character varying(24)	Process ID of the current session.
logon_time	date	Login time of the current session. **TIMESTAMP (0) WITHOUT TIME ZONE** is displayed in an O-compatible database.
last_call_et	integer	Duration when the status of the current session changes last time.
osuser	text	OS username of the server.
terminal	character varying(30)	Not supported. Set it to **NULL**.
type	character varying(10)	Not supported. Set it to **NULL**.
sql_address	raw	Not supported. Set it to **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
sql_hash_value	numeric	Not supported. Set it to **NULL**.
sql_child_number	numeric	Not supported. Set it to **NULL**.
sql_exec_id	numeric	Not supported. Set it to **NULL**.
prev_sql_addr	raw	Not supported. Set it to **NULL**.
prev_hash_value	numeric	Not supported. Set it to **NULL**.
prev_sql_id	character varying(13)	Not supported. Set it to **NULL**.
prev_child_number	numeric	Not supported. Set it to **NULL**.
prev_exec_start	date	Not supported. Set it to **NULL**. • **TIMESTAMP (0) WITHOUT TIME ZONE** is displayed in an O-compatible database.
prev_exec_id	numeric	Not supported. Set it to **NULL**.
plsql_entry_object_id	numeric	Not supported. Set it to **NULL**.
plsql_entry_subprogram_id	numeric	Not supported. Set it to **NULL**.
plsql_object_id	numeric	Not supported. Set it to **NULL**.
plsql_subprogram_id	numeric	Not supported. Set it to **NULL**.
module	text	Name of the running module, which is set by calling DBE_APPLICATION_INFO.SET_MODULE.
module_hash	numeric	Not supported. Set it to **NULL**.
action	text	Name of the current operation in the current module, which is set by calling DBE_APPLICATION_INFO.SET_MODULE or DBE_APPLICATION_INFO.SET_ACTION.
action_hash	numeric	Not supported. Set it to **NULL**.
fixed_table_sequence	numeric	Not supported. Set it to **NULL**.
row_wait_obj#	numeric	Not supported. Set it to **NULL**.
row_wait_file#	numeric	Not supported. Set it to **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
row_wait_block#	numeric	Not supported. Set it to **NULL**.
row_wait_row#	numeric	Not supported. Set it to **NULL**.
top_level_call#	numeric	Not supported. Set it to **NULL**.
pdml_enabled	character varying(3)	Not supported. Set it to **NULL**.
failover_type	character varying(13)	Not supported. Set it to **NULL**.
failover_method	character varying(10)	Not supported. Set it to **NULL**.
failed_over	character varying(3)	Not supported. Set it to **NULL**.
resource_consumer_group	character varying(32)	Not supported. Set it to **NULL**.
pddl_status	character varying(8)	Not supported. Set it to **NULL**.
pq_status	character varying(8)	Not supported. Set it to **NULL**.
current_queue_duration	numeric	Not supported. Set it to **NULL**.
client_identifier	character varying(64)	Not supported. Set it to **NULL**.
blocking_session_status	character varying(11)	Not supported. Set it to **NULL**.
blocking_instance	numeric	Not supported. Set it to **NULL**.
blocking_session	numeric	Not supported. Set it to **NULL**.
final_blocking_session_status	character varying(11)	Not supported. Set it to **NULL**.
final_blocking_instance	numeric	Not supported. Set it to **NULL**.
final_blocking_session	numeric	Not supported. Set it to **NULL**.
seq#	numeric	Not supported. Set it to **NULL**.
event#	numeric	Not supported. Set it to **NULL**.
p1text	character varying(64)	Not supported. Set it to **NULL**.
p1	numeric	Not supported. Set it to **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
p1raw	raw	Not supported. Set it to **NULL**.
p2text	character varying(64)	Not supported. Set it to **NULL**.
p2	numeric	Not supported. Set it to **NULL**.
p2raw	raw	Not supported. Set it to **NULL**.
p3text	character varying(64)	Not supported. Set it to **NULL**.
p3	numeric	Not supported. Set it to **NULL**.
p3raw	raw	Not supported. Set it to **NULL**.
wait_class_id	numeric	Not supported. Set it to **NULL**.
wait_class#	numeric	Not supported. Set it to **NULL**.
wait_class	character varying(64)	Not supported. Set it to **NULL**.
wait_time	numeric	Not supported. Set it to **NULL**.
seconds_in_wait	numeric	Not supported. Set it to **NULL**.
state	character varying(19)	Not supported. Set it to **NULL**.
wait_time_micro	numeric	Not supported. Set it to **NULL**.
time_remaining_micro	numeric	Not supported. Set it to **NULL**.
time_since_last_wait_micro	numeric	Not supported. Set it to **NULL**.
service_name	character varying(64)	Not supported. Set it to **NULL**.
sql_trace	character varying(8)	Not supported. Set it to **NULL**.
sql_trace_waits	character varying(5)	Not supported. Set it to **NULL**.
sql_trace_binds	character varying(5)	Not supported. Set it to **NULL**.
sql_trace_plan_stats	character varying(10)	Not supported. Set it to **NULL**.
session_edition_id	numeric	Not supported. Set it to **NULL**.
creator_addr	raw	Not supported. Set it to **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
creator_serial#	numeric	Not supported. Set it to **NULL**.
ecid	character varying(64)	Not supported. Set it to **NULL**.
sql_translation_profile_id	numeric	Not supported. Set it to **NULL**.
pga_tunable_mem	numeric	Not supported. Set it to **NULL**.
shard_ddl_status	character varying(8)	Not supported. Set it to **NULL**.
con_id	numeric	Not supported. Set it to **NULL**.
external_name	character varying(1024)	Not supported. Set it to **NULL**.
plsql_debugger_connected	character varying(5)	Not supported. Set it to **NULL**.

NOTE

During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of the **logon_time** column in the view is changed to **datea**.

7.3.13.102 MPP_TABLES

MPP_TABLES displays information about tables in PGXC_CLASS. For details about the columns, see **Table 7-384**.

Table 7-384 MPP_TABLES columns

Name	Type	Description
schemaname	name	Name of the schema that contains a table.
tablename	name	Table name.
tableowner	name	Table owner.
tablespace	name	Tablespace containing the table.
pgroup	name	Name of the node.
nodeoids	oidvector_extend	List of distributed table node OIDs.

7.3.13.103 MY_AUDIT_TRAIL

MY_AUDIT_TRAIL displays the standard audit trail entries related to the current user. The GaussDB audit information is mainly obtained through the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

gs_query_audit() function. This view exists in both the PG_CATALOG and SYS schemas. Only users with the AUDITADMIN attribute and the SELECT permission on MY_AUDIT_TRAIL can view audit information. If separation of duties is disabled, users with the SYSADMIN attribute can also view audit information. The **action_name** column of GaussDB is different from that of Oracle Database in terms of the audit actions. The type of the **transactionid** column is the same as that in Oracle Database. In GaussDB, the **sql_text** column is the parsed SQL statement, which is not completely the same as the executed SQL statement. For details about the columns, see **Table 7-385**.

CAUTION

In a distributed system, the gs_query_audit() function can only be used to query the audit information of the current node.

Table 7-385 MY_AUDIT_TRAIL columns

Name	Type	Description
os_username	character varying(255)	Not supported yet. Its value is **NULL**.
username	character varying(128)	Name of the user whose operation is audited, not the user ID.
userhost	character varying(128)	Not supported yet. Its value is **NULL**.
terminal	character varying(255)	Not supported yet. Its value is **NULL**.
timestamp	date	Date and time when an audit trail entry is created in the local database session time zone (user login date and time of the entry created by the audit session).
owner	character varying(128)	Creator of the object affected by the operation.
obj_name	character varying(128)	Name of the object affected by the operation.
action	numeric	Not supported yet. Its value is **NULL**.
action_name	character varying(28)	Action type corresponding to the code in the **action** column.
new_owner	character varying(128)	Not supported yet. Its value is **NULL**.
new_name	character varying(128)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
obj_privilege	character varying(32)	Not supported yet. Its value is **NULL**.
sys_privilege	character varying(40)	Not supported yet. Its value is **NULL**.
admin_option	character varying(1)	Not supported yet. Its value is **NULL**.
grantee	character varying(128)	Not supported yet. Its value is **NULL**.
audit_option	character varying(40)	Not supported yet. Its value is **NULL**.
ses_actions	character varying(19)	Not supported yet. Its value is **NULL**.
logoff_time	date	Not supported yet. Its value is **NULL**.
logoff_lread	numeric	Not supported yet. Its value is **NULL**.
logoff_pread	numeric	Not supported yet. Its value is **NULL**.
logoff_lwrite	numeric	Not supported yet. Its value is **NULL**.
logoff_dlock	character varying(40)	Not supported yet. Its value is **NULL**.
comment_text	character varying(4000)	Not supported yet. Its value is **NULL**.
sessionid	numeric	Not supported yet. Its value is **NULL**.
entryid	numeric	Not supported yet. Its value is **NULL**.
statementid	numeric	Not supported yet. Its value is **NULL**.
returncode	numeric	Not supported yet. Its value is **NULL**.
priv_used	character varying(40)	Not supported yet. Its value is **NULL**.
client_id	character varying(128)	Not supported yet. Its value is **NULL**.
econtext_id	character varying(64)	Not supported yet. Its value is **NULL**.
session_cpu	numeric	Not supported yet. Its value is **NULL**.
extended_timestamp	timestamp(6) with time zone	Time zone of the timestamp when an audit trail entry is created (user login date and time of the entry created by the audit session in UTC).
proxy_sessionid	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
global_uid	character varying(32)	Not supported yet. Its value is **NULL**.
instance_number	numeric	Not supported yet. Its value is **NULL**.
os_process	character varying(16)	Not supported yet. Its value is **NULL**.
transactionid	text	Identifier of the transaction that accesses or modifies an object.
scn	numeric	Not supported yet. Its value is **NULL**.
sql_bind	nvarchar2(2000)	Not supported yet. Its value is **NULL**.
sql_text	nvarchar2	SQL text of the query.
obj_edition_name	character varying(128)	Not supported yet. Its value is **NULL**.
dbid	numeric	Not supported yet. Its value is **NULL**.
rls_info	clob	Not supported yet. Its value is **NULL**.
current_user	character varying(128)	Not supported yet. Its value is **NULL**.

NOTE

During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of the **timestamp** column in the view is changed to **datea**.

7.3.13.104 MY_COL_COMMENTS

MY_COL_COMMENTS displays column comments of the table accessible to the current user. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-386**.

Table 7-386 MY_COL_COMMENTS columns

Name	Type	Description
owner	character varying(128)	Table owner.
table_name	character varying(128)	Table name.
column_name	character varying(128)	Column name.
comments	text	Comments.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
origin_con_id	numeric	Not supported. The value is **0**.
schema	character varying(64)	Name of the namespace to which the column belongs.

7.3.13.105 MY_COLL_TYPES

MY_COLL_TYPES displays information about types of collections created by the current user. By default, it is accessible to all users. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-387**.

Table 7-387 MY_COLL_TYPES columns

Name	Type	Description
owner	character varying(128)	Owner of a collection.
type_name	character varying(128)	Name of a collection.
coll_type	character varying(128)	Description of a collection.
upper_bound	numeric	Not supported yet. Its value is **NULL**.
elem_type_mod	character varying(7)	Not supported yet. Its value is **NULL**.
elem_type_owner	character varying(128)	Owner of the element type on which the collection is based. This parameter is mainly used for user-defined types.
elem_type_name	character varying(128)	Name of the data type or user-defined type on which the collection is based.
length	numeric	Not supported yet. Its value is **NULL**.
precision	numeric	Not supported yet. Its value is **NULL**.
scale	numeric	Not supported yet. Its value is **NULL**.
character_set_name	character varying(44)	Not supported yet. Its value is **NULL**.
elem_storage	character varying(7)	Not supported yet. Its value is **NULL**.
nulls_stored	character varying(3)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
char_used	character varying(1)	Not supported yet. Its value is **NULL**.

7.3.13.106 MY_CONS_COLUMNS

MY_CONS_COLUMNS displays information about PRIMARY KEY constraint columns in tables accessible to the current user. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-388**.

Table 7-388 MY_CONS_COLUMNS columns

Name	Type	Description
table_name	character varying(64)	Name of a constraint-related table.
column_name	character varying(64)	Name of a constraint-related column.
constraint_name	character varying(64)	Constraint name.
owner	character varying(64)	Constraint creator.
position	smallint	Position of a column in a table.

7.3.13.107 MY_CONSTRAINTS

MY_CONSTRAINTS displays table constraint information accessible to the current user. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-389**.

Table 7-389 MY_CONSTRAINTS columns

Name	Type	Description
constraint_name	character varying(64)	Constraint name.
constraint_type	text	Constraint type. • **c**: CHECK constraint. • **f**: FOREIGN KEY constraint. • **p**: PRIMARY KEY constraint. • **u**: UNIQUE constraint.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
table_name	character varying(64)	Name of a constraint-related table.
index_owner	character varying(64)	Owner of a constraint-related index (only for the UNIQUE constraint and PRIMARY KEY constraint).
index_name	character varying(64)	Name of a constraint-related index (only for the UNIQUE constraint and PRIMARY KEY constraint).
owner	character varying(64)	Constraint creator.

7.3.13.108 MY_DEPENDENCIES

MY_DEPENDENCIES displays the dependencies between objects that are accessible to the current user. This view exists in the PG_CATALOG and SYS schemas and all users can access this view. For details about the columns, see **Table 7-390**.

Table 7-390 MY_DEPENDENCIES columns

Name	Type	Description
name	name	Object name.
type	character varying(18)	Object type.
referenced_owner	name	Owner of the referenced object.
referenced_name	name	Name of the referenced object.
referenced_type	character varying(18)	Type of the referenced object.
referenced_link_name	character varying(128)	Name of the link to the parent object (if remote).
dependency_type	character varying(4)	Specifies whether the dependency is **REF**. The default value is **HARD**, indicating hard reference.

7.3.13.109 MY_IND_COLUMNS

MY_IND_COLUMNS displays column information about all indexes accessible to the current user. It is accessible to all users. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-391**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-391 MY_IND_COLUMNS columns

Name	Type	Description
index_owner	character varying(64)	Index owner.
index_name	character varying(64)	Index name.
table_owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.
column_name	name	Column name.
column_position	smallint	Position of a column in an index.
column_length	numeric	Length of the column. For the variable-length type, its value is **NULL**.
char_length	numeric	Maximum length of a column, in bytes.
descend	character varying(4)	Specifies whether columns are sorted in descending (**DESC**) or ascending (**ASC**) order.
collated_column_id	numeric	Not supported yet. Its value is **NULL**.

7.3.13.110 MY_IND_EXPRESSIONS

MY_IND_EXPRESSIONS displays information about function-based expression indexes accessible to the current user. It is accessible to all users. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-392**.

Table 7-392 MY_IND_EXPRESSIONS columns

Name	Type	Description
table_owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.
index_owner	character varying(64)	Index owner.
index_name	character varying(64)	Index name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
column_expression	text	Function-based index expression of a specified column.
column_position	smallint	Position of a column in an index.

7.3.13.111 MY_INDEXES

MY_INDEXES displays index information about the current user. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-393**.

Table 7-393 MY_INDEXES columns

Name	Type	Description
owner	character varying(64)	Index owner
index_name	character varying(64)	Index name
table_name	character varying(64)	Name of the table corresponding to the index
uniqueness	text	Specifies whether the index is a unique index. • **UNIQUE**: unique index. • **NONUNIQUE**: non-unique index.
partitioned	character(3)	Specifies whether the index has the property of partitioned tables. • **Yes**: The index has the property of a partitioned table. • **No**: The index does not have the property of a partitioned table.
generated	character varying(1)	Specifies whether the index name is generated by the system. • **y**: The index name is generated by the system. • **n**: The index name is not generated by the system.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
index_type	character varying(27)	Index type • **NORMAL**: Index attributes are simple references, and the expression tree is empty. • **FUNCTION-BASED NORMAL**: Expression trees are used for index attributes that are not simple column references.
table_owner	character varying(128)	Owner of the index object
table_type	character(11)	Type of an index object • **TABLE**: The index object is of the table type.
tablespace_name	character varying(30)	Name of the tablespace that contains the index.
status	character varying(8)	Status of a non-partitioned index. • **VALID**: Non-partitioned indexes can be used for query. • **UNUSABLE**: The non-partitioned index is unavailable. • **N/A**: The index has the property of a partitioned table.
compression	character varying(13)	Not supported yet. Its value is **NULL**.
prefix_length	numeric	Not supported yet. Its value is **NULL**.
ini_trans	numeric	Not supported yet. Its value is **NULL**.
max_trans	numeric	Not supported yet. Its value is **NULL**.
initial_extent	numeric	Not supported yet. Its value is **NULL**.
next_extent	numeric	Not supported yet. Its value is **NULL**.
min_extents	numeric	Not supported yet. Its value is **NULL**.
max_extents	numeric	Not supported yet. Its value is **NULL**.
pct_increase	numeric	Not supported yet. Its value is **NULL**.
pct_threshold	numeric	Not supported yet. Its value is **NULL**.
include_column	numeric	Not supported yet. Its value is **NULL**.
freelists	numeric	Not supported yet. Its value is **NULL**.
freelist_groups	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
pct_free	numeric	Not supported yet. Its value is **NULL**.
logging	character varying(3)	Not supported yet. Its value is **NULL**.
blevel	numeric	Not supported yet. Its value is **NULL**.
leaf_blocks	numeric	Not supported yet. Its value is **NULL**.
distinct_keys	numeric	Not supported yet. Its value is **NULL**.
avg_leaf_blocks_per_key	numeric	Not supported yet. Its value is **NULL**.
avg_data_blocks_per_key	numeric	Not supported yet. Its value is **NULL**.
clustering_factor	numeric	Not supported yet. Its value is **NULL**.
num_rows	numeric	Not supported yet. Its value is **NULL**.
sample_size	numeric	Not supported yet. Its value is **NULL**.
last_analyzed	date	Not supported yet. Its value is **NULL**.
degree	character varying(40)	Not supported yet. Its value is **NULL**.
instances	character varying(40)	Not supported yet. Its value is **NULL**.
temporary	character varying(1)	Not supported yet. Its value is **NULL**.
secondary	character varying(1)	Not supported yet. Its value is **NULL**.
buffer_pool	character varying(7)	Not supported yet. Its value is **NULL**.
flash_cache	character varying(7)	Not supported yet. Its value is **NULL**.
cell_flash_cache	character varying(7)	Not supported yet. Its value is **NULL**.
user_stats	character varying(3)	Not supported yet. Its value is **NULL**.
duration	character varying(15)	Not supported yet. Its value is **NULL**.
pct_direct_access	numeric	Not supported yet. Its value is **NULL**.
ityp_owner	character varying(128)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
ityp_name	character varying(128)	Not supported yet. Its value is **NULL**.
parameters	character varying(1000)	Not supported yet. Its value is **NULL**.
global_stats	character varying(3)	Not supported yet. Its value is **NULL**.
domidx_status	character varying(12)	Not supported yet. Its value is **NULL**.
domidx_opstatus	character varying(6)	Not supported yet. Its value is **NULL**.
funcidx_status	character varying(8)	Not supported yet. Its value is **NULL**.
join_index	character varying(3)	Not supported yet. Its value is **NULL**.
iot_redundant_pkey_elim	character varying(3)	Not supported yet. Its value is **NULL**.
dropped	character varying(3)	Not supported yet. Its value is **NULL**.
visibility	character varying(9)	Specifies whether the index is visible to the optimizer. • **VISIBLE**: The index is visible to the optimizer. • **INVISIBLE**: The index is visible to the optimizer.
domidx_management	character varying(14)	Not supported yet. Its value is **NULL**.
segment_created	character varying(3)	Not supported yet. Its value is **NULL**.
orphaned_entries	character varying(3)	Not supported yet. Its value is **NULL**.
indexing	character varying(7)	Not supported yet. Its value is **NULL**.
auto	character varying(3)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.112 MY_JOBS

MY_JOBS displays details about the scheduled tasks owned by the current user. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-394**.

Table 7-394 MY_JOBS columns

Name	Type	Description
job	bigint	Job ID.
log_user	name	Username of the job creator.
priv_user	name	Username of the job executor.
dbname	name	Name of the database where the job is created.
schema_user	name	Default schema name of a scheduler job.
start_date	timestamp without time zone	Time when a job starts to be executed for the first time.
start_suc	text	Start time of the first successful job execution.
last_date	timestamp without time zone	Start time of the last job execution.
last_suc	text	Start time of the last successful job execution.
last_sec	text	Start time of the last successful job execution. Compatibility is supported.
this_date	timestamp without time zone	Start time of the ongoing job execution.
this_suc	text	Start time of the ongoing job execution.
this_sec	text	Start time of an ongoing job execution. Compatibility is supported.
next_date	timestamp without time zone	Schedule time of the next job execution.
next_suc	text	Schedule time of the next job execution.
next_sec	text	Schedule time of the next job execution. Compatibility is supported.
total_time	numeric	Latest execution duration of a task.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
broken	text	If the value of the **status** column is **d**, the value of **broken** is **y**. Otherwise, the value of **broken** is **n**.
status	"char"	Status of the current job. The value can be **r**, **s**, **f**, or **d**. The default value is **r**. • **r**: running • **s**: successful execution • **f**: failed execution • **d**: canceling an execution
interval	text	Time expression used to calculate the next time the job will be executed. If this parameter is set to **null**, the job will be executed once only.
failures	smallint	Number of times the job has started and failed. If a job fails to be executed for 16 consecutive times, no more attempt will be made on it.
what	text	Executable job.
nls_env	character varying(4000)	Not supported yet. Its value is **NULL**.
misc_env	raw	Not supported yet. Its value is **NULL**.
instance	numeric	Not supported yet. Its value is **NULL**.

7.3.13.113 MY_OBJECTS

MY_OBJECTS displays database objects accessible to the current user. This view exists in both PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-395**.

Table 7-395 MY_OBJECTS columns

Name	Type	Description
object_name	name	Object name.
object_id	oid	Object OID.
object_type	name	Object type, including table, index, sequence, view, and global secondary index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
namespace	oid	Namespace that an object belongs to.
temporary	character(1)	Specifies whether an object is a temporary object.
status	character varying(7)	Object status.
subobject_name	name	Subobject name of an object.
generated	character(1)	Specifies whether an object name is generated by the system.
created	timestamp with time zone	Creation time of an object.
last_ddl_time	timestamp with time zone	Last modification time of an object.
default_collation	character varying(100)	Default collation of objects.
data_object_id	numeric	Not supported yet. Its value is **NULL**.
timestamp	character varying(19)	Not supported yet. Its value is **NULL**.
secondary	character varying(1)	Not supported yet. Its value is **NULL**.
edition_name	character varying(128)	Not supported yet. Its value is **NULL**.
sharing	character varying(18)	Not supported yet. Its value is **NULL**.
editionable	character varying(1)	Not supported yet. Its value is **NULL**.
oracle_maintained	character varying(1)	Not supported yet. Its value is **NULL**.
application	character varying(1)	Not supported yet. Its value is **NULL**.
duplicated	character varying(1)	Not supported yet. Its value is **NULL**.
sharded	character varying(1)	Not supported yet. Its value is **NULL**.
created_appid	numeric	Not supported yet. Its value is **NULL**.
modified_appid	numeric	Not supported yet. Its value is **NULL**.
created_vsnid	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
modified_vsnid	numeric	Not supported yet. Its value is **NULL**.

NOTICE

For details on the value ranges of **created** and **last_ddl_time**, see **PG_OBJECT**.

7.3.13.114 MY_PROCEDURES

MY_PROCEDURES displays information about stored procedures, functions, or triggers owned by the current user. This view exists in both PG_CATALOG and SYS schemas. This view can be accessed by all users. Only the information about the current user can be viewed. For details about the columns, see **Table 7-396**.

Table 7-396 MY_PROCEDURES columns

Name	Type	Description
owner	character varying(64)	Owner of a stored procedure, function, trigger, or package.
object_name	character varying(64)	Name of a stored procedure, function, or trigger. This column will be a package name if the object is a function in a package or a stored procedure.
procedure_name	character varying(128)	This column will be the name of a function or stored procedure in a package if **object_name** is a package name. Otherwise, the column is empty.
object_id	oid	OID of a stored procedure, function, trigger, or package.
subprogram_id	numeric	Location of a function or stored procedure in a package.
overload	character varying(40)	The *n*th overloaded function of the same name.
object_type	character varying(13)	Object type.
aggregate	character varying(3)	Specifies whether the function is an aggregate function: • **yes** • **no**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
pipelined	character varying(3)	Not supported. The value is **no**.
impltypeowner	character varying(128)	Owner of an implementation type.
impltypename	character varying(128)	Name of an implementation type.
parallel	character varying(3)	Not supported. The value is **no**.
interface	character varying(3)	Not supported. The value is **no**.
deterministic	character varying(3)	Not supported. The value is **no**.
authid	character varying(12)	Specifies whether to use the creator permission or caller permission: • **definer**: The creator permission is used. • **current_user**: The caller permission is used. This column conflicts with reserved keywords. Therefore, add the view name when calling this column.
result_cache	character varying(3)	Not supported yet. Its value is **NULL**.
origin_con_id	character varying(256)	Not supported. The value is **0**.
polymorphic	character varying(5)	Not supported yet. Its value is **NULL**.
argument_number	smallint	Number of input parameters in the stored procedure.

7.3.13.115 MY_SCHEDULER_JOB_ARGS

MY_SCHEDULER_JOB_ARG displays the parameters related to the jobs owned by the current user. This view can be accessed by all users. Only the information about the current user can be viewed. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-397**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-397 MY_SCHEDULER_JOB_ARGS columns

Name	Type	Description
job_name	character varying(128)	Name of the job to which the parameter belongs.
argument_name	character varying(128)	Parameter name.
argument_position	numeric	Position of the parameter in the parameter list.
argument_type	character varying(257)	Data type of a parameter, which can be user-defined.
value	character varying(4000)	Parameter value.
anydata_value	character varying(4000)	Not supported yet. Its value is **NULL**.
out_argument	character varying(5)	Reserved column. The value is **NULL**.

7.3.13.116 MY_SCHEDULER_JOBS

MY_SCHEDULER_JOBS displays information about all DBE_SCHEDULER scheduled tasks owned by the current user in the database. By default, this view exists in the PG_CATALOG and SYS schemas and all users can access this view. For details about the columns, see **Table 7-398**.

Table 7-398 MY_SCHEDULER_JOBS columns

Name	Type	Description
job_name	text	Name of a scheduler job.
job_subname	character varying(128)	Not supported yet. Its value is **NULL**.
job_style	text	Style of a scheduled job, which is specified during creation. Its value can only be **"REGULAR"**. If this parameter is not specified, the value is **NULL**.
job_creator	name	Creator of a scheduler job.
client_id	character varying(65)	Not supported yet. Its value is **NULL**.
global_uid	character varying(33)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
program_owner	character varying(4000)	Owner of a program referenced by a scheduler job.
program_name	text	Name of the program referenced by a scheduler job.
job_type	character varying(16)	Type of the inline program of a scheduler job.
job_action	text	Program content of a scheduler job.
number_of_arguments	text	Number of parameters of a scheduler job. The value ranges from 0 to 255.
schedule_owner	character varying(4000)	Not supported yet. Its value is **NULL**.
schedule_name	text	Name of the schedule referenced by a scheduler job.
schedule_type	character varying(12)	Not supported yet. Its value is **NULL**.
start_date	timestamp without time zone	Start time of a scheduler job.
repeat_interval	text	Period of a scheduler job.
event_queue_owner	character varying(128)	Not supported yet. Its value is **NULL**.
event_queue_name	character varying(128)	Not supported yet. Its value is **NULL**.
event_queue_agent	character varying(523)	Not supported yet. Its value is **NULL**.
event_condition	character varying(4000)	Not supported yet. Its value is **NULL**.
event_rule	character varying(261)	Not supported yet. Its value is **NULL**.
file_watcher_owner	character varying(261)	Not supported yet. Its value is **NULL**.
file_watcher_name	character varying(261)	Not supported yet. Its value is **NULL**.
end_date	timestamp without time zone	End time of a scheduler job.
job_class	text	Name of the scheduler job class to which a scheduler job belongs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
enabled	boolean	Status of a scheduler job.
auto_drop	text	Status of the automatic deletion function of a scheduler job.
restart_on_recovery	character varying(5)	Not supported yet. Its value is **NULL**.
restart_on_failure	character varying(5)	Not supported yet. Its value is **NULL**.
state	"char"	Status of a scheduler job.
job_priority	numeric	Not supported yet. Its value is **NULL**.
run_count	numeric	Not supported yet. Its value is **NULL**.
uptime_run_count	numeric	Not supported yet. Its value is **NULL**.
max_runs	numeric	Not supported yet. Its value is **NULL**.
failure_count	smallint	Number of scheduler job failures.
uptime_failure_count	numeric	Not supported yet. Its value is **NULL**.
max_failures	numeric	Maximum number of failures allowed before the status of a scheduler job is marked as broken.
retry_count	numeric	Not supported yet. Its value is **NULL**.
last_start_date	timestamp without time zone	Last time when a scheduler job was started.
last_run_duration	interval day to second(6)	Last execution duration of a scheduler job.
next_run_date	timestamp without time zone	Next execution time of a scheduler job.
schedule_limit	interval day to second(0)	Not supported yet. Its value is **NULL**.
max_run_duration	interval day to second(0)	Not supported yet. Its value is **NULL**.
logging_level	character varying(11)	Not supported yet. Its value is **NULL**.
store_output	character varying(5)	Specifies whether to store the output information of all scheduler jobs.
stop_on_window_close	character varying(5)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
instance_stickiness	character varying(5)	Not supported yet. Its value is **NULL**.
raise_events	character varying(4000)	Not supported yet. Its value is **NULL**.
system	character varying(5)	Not supported yet. Its value is **NULL**.
job_weight	numeric	Not supported yet. Its value is **NULL**.
nls_env	character varying(4000)	Not supported yet. Its value is **NULL**.
source	character varying(128)	Not supported yet. Its value is **NULL**.
number_of_destinations	numeric	Not supported yet. Its value is **NULL**.
destination_owner	character varying(261)	Not supported yet. Its value is **NULL**.
destination	text	Target name of a scheduler job. If the value of **destination** is **CCN**, the task is a CCN job. For details, see **DBE_SCHEDULER**.
credential_owner	character varying(128)	Not supported yet. Its value is **NULL**.
credential_name	text	Certificate name of a scheduler job.
instance_id	oid	OID of the current database.
deferred_drop	character varying(5)	Not supported yet. Its value is **NULL**.
allow_runs_in_restricted_mode	character varying(5)	Not supported yet. Its value is **NULL**.
comments	text	Comments of a scheduler job.
flags	numeric	Not supported yet. Its value is **NULL**.
restartable	character varying(5)	Not supported yet. Its value is **NULL**.
has_constraints	character varying(5)	Not supported yet. Its value is **NULL**.
connect_credential_owner	character varying(128)	Not supported yet. Its value is **NULL**.
connect_credential_name	character varying(128)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
fail_on_script_error	character varying(5)	Not supported yet. Its value is **NULL**.

7.3.13.117 MY_SCHEDULER_PROGRAM_ARGS

MY_SCHEDULER_PROGRAM_ARG displays the parameters related to the programs owned by the current user. This view can be accessed by all users. Only the information about the current user can be viewed. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-399**.

Table 7-399 MY_SCHEDULER_PROGRAM_ARGS columns

Name	Type	Description
program_name	character varying(128)	Name of the program to which the parameter belongs.
argument_name	character varying(128)	Parameter name.
argument_position	numeric	Position of the parameter in the parameter list.
argument_type	character varying(257)	Data type of a parameter, which can be user-defined.
metadata_attribute	character varying(19)	Not supported yet. Its value is **NULL**.
default_value	character varying(4000)	Default parameter value.
default_anydata_value	character varying(4000)	Not supported yet. Its value is **NULL**.
out_argument	character varying(5)	Reserved column. The value is **NULL**.

7.3.13.118 MY_SEQUENCES

MY_SEQUENCES displays the sequence information about the current user. This view exists in both PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-400**.

Table 7-400 MY_SEQUENCES columns

Name	Type	Description
sequence_owner	name	Owner of a sequence.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
sequence_name	name	Name of a sequence.
min_value	int16	Minimum value of a sequence.
max_value	int16	Maximum value of a sequence.
increment_by	int16	Value by which the sequence is incremented.
cycle_flag	character(1)	Specifies whether a sequence is a cyclic sequence. The values are as follows: • **y**: yes. • **n**: no.
order_flag	character(1)	Specifies whether a sequence is an ordered sequence. The values are as follows: • **y**: yes. • **n**: no.
cache_size	int16	Size of the sequence disk cache.
last_number	int16	Last value of the sequence allocated in a batch from the GTM at a time.
scale_flag	character varying(1)	Specifies whether a sequence is a scalable sequence. This parameter is not supported. Set it to **NULL**.
extend_flag	character varying(1)	Specifies whether the value generated by a scalable sequence exceeds the maximum or minimum value of the sequence. Not supported yet. Its value is **NULL**.
sharded_flag	character varying(1)	Specifies whether a sequence is a shard sequence. Not supported yet. Its value is **NULL**.
session_flag	character varying(1)	Specifies whether a sequence has a private session. This is not supported. The value is **NULL**.
keep_value	character varying(1)	Specifies whether to retain the sequence value during replay after a failure. Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.119 MY_SOURCE

MY_SOURCE displays the definition information about stored procedures, functions, and triggers accessible to the current user. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-401**.

Table 7-401 MY_SOURCE columns

Name	Type	Description
owner	name	Object owner.
name	name	Object name.
type	name	Object type. The value can be **function**, **procedure**, or **trigger**.
line	numeric	Number of the line in the definition information.
text	text	Text source of the storage object.
origin_con_id	character varying(256)	Not supported. The value is **0**.

7.3.13.120 MY_SYNONYMS

MY_SYNONYMS displays synonyms in the current schema. This view can be accessed by all users. Only the information about the current user can be viewed. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-402**.

Table 7-402 MY_SYNONYMS columns

Name	Type	Description
schema_name	text	Name of the schema to which the synonym belongs.
synonym_name	text	Synonym name.
table_owner	text	Owner of the associated object. Although the column is called **table_owner**, the associated object is not necessarily a table. It can be any common database object, such as a view, stored procedure, or synonym.
table_name	text	Name of the associated object. Although the column is called **table_name**, the associated object is not necessarily a table. It can be any common database object, such as a view, stored procedure, or synonym.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
table_schema_name	text	Schema name of the associated object. Although the column is called **table_schema_name**, the associated object is not necessarily a table. It can be any common database object, such as a view, stored procedure, or synonym.
db_link	character varying(128)	Reserved column. The value is **NULL**.
origin_con_id	character varying(256)	Not supported. The value is **0**.

7.3.13.121 MY_TAB_COL_STATISTICS

MY_TAB_COL_STATISTICS displays column statistics and histogram information extracted from MY_TAB_COLUMNS. All users can access this view. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-403**.

Table 7-403 MY_TAB_COL_STATISTICS columns

Name	Type	Description
table_name	character varying(128)	Table name.
column_name	character varying(128)	Column name.
num_distinct	numeric	Number of different values in a column.
low_value	raw	Low value in a column.
high_value	raw	High value in a column.
density	numeric	• If there is a histogram on **COLUMN_NAME**, this column displays the selectivity of values in the histogram that span less than two endpoints. It does not represent the selectivity of values that span two or more endpoints. • If no histogram is available on **COLUMN_NAME**, the value of this column is **1/NUM_DISTINCT**.
num_nulls	numeric	Number of empty values in a column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
num_buckets	numeric	Number of buckets in the histogram of a column.
last_analyzed	date	Date when a column was last analyzed. After the database is restarted, data loss will occur.
sample_size	numeric	Sample size used to analyze a column.
global_stats	character varying(3)	Not supported yet. Its value is **NULL**.
user_stats	character varying(3)	Not supported yet. Its value is **NULL**.
notes	character varying(99)	Not supported yet. Its value is **NULL**.
avg_col_len	numeric	Average length of a column, in bytes.
histogram	character varying(15)	Specifies whether the histogram exists and the type of the histogram if it exists. • **none**: no histogram. • **frequency**: frequency histogram. • **equi-width**: equal-width histogram.
scope	character varying(7)	For statistics collected on any table other than global temporary tables, the value is **shared** (indicating that the statistics are shared among all sessions).
schema	character varying(64)	Name of the namespace to which the column belongs.

NOTE

During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of the **last_analyzed** column in the view is changed to **datea**.

7.3.13.122 MY_TAB_COLUMNS

MY_TAB_COLUMNS displays the columns of the tables and views owned by the current user. This view exists in the PG_CATALOG and SYS schemas. All users can access this view. Only the information about the user is displayed. For details about the columns, see **Table 7-404**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-404 MY_TAB_COLUMNS columns

Name	Type	Description
owner	character varying(64)	Table owner.
table_name	character varying(64)	Table name.
column_name	character varying(64)	Column name.
data_type	character varying(128)	Data type of a column.
data_type_mod	character varying(3)	Not supported yet. Its value is **NULL**.
data_type_owner	character varying(128)	Owner of the data type of a column.
data_length	integer	Length of a column, in bytes.
data_precision	integer	Precision of the data type. It takes effect on the numeric type. For other types, the value is **NULL**.
data_scale	integer	Number of decimal places. It takes effect on the numeric type. For other types, the value is **0**.
nullable	bpchar	Specifies whether a column can be empty (**n** for a PRIMARY KEY constraint or NOT NULL constraint).
column_id	integer	Sequence number of a column when a table is created.
default_length	numeric	Length of the default value of a column, in bytes.
data_default	text	Default value of a column.
num_distinct	numeric	Number of different values in a column.
low_value	raw	Minimum value in a column.
high_value	raw	Maximum value in a column.
density	numeric	Column density.
num_nulls	numeric	Number of empty values in a column.
num_buckets	numeric	Number of buckets in the histogram of a column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
last_analyzed	date	Last analysis date.
sample_size	numeric	Sample size used to analyze a column.
character_set_name	character varying(44)	Not supported yet. Its value is **NULL**.
char_col_decl_length	numeric	Declaration length of a column of the character type.
global_stats	character varying(3)	Not supported yet. Its value is **no**.
user_stats	character varying(3)	Not supported yet. Its value is **no**.
avg_col_len	numeric	Average length of a column, in bytes.
char_length	numeric	Column length (in characters). It takes effect only on the varchar, nvarchar2, bpchar, and char types.
char_used	character varying(1)	Not supported. Set it to **B** for the varchar, bpchar, or char data type. Set it to **C** for the nvarchar2 data type. For other data types, set it to **NULL**.
v80_fmt_image	character varying(3)	Not supported yet. Its value is **NULL**.
data_upgraded	character varying(3)	Not supported yet. Its value is **yes**.
histogram	character varying(15)	Specifies whether the histogram exists and the type of the histogram if it exists. • **none**: no histogram. • **frequency**: frequency histogram. • **equi-width**: equal-width histogram.
default_on_null	character varying(3)	Not supported yet. Its value is **NULL**.
identity_column	character varying(3)	Specifies whether a column is an identity column. • **yes**: It is an identity column. • **no**: It is not an identity column.
sensitive_column	character varying(3)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
evaluation_edition	character varying(128)	Not supported yet. Its value is **NULL**.
unusable_before	character varying(128)	Not supported yet. Its value is **NULL**.
unusable_beginning	character varying(128)	Not supported yet. Its value is **NULL**.
collation	character varying(100)	Collation rule of a column. This column conflicts with reserved keywords. Therefore, add the view name when calling this column.
comments	text	Comment of a column.
schema	character varying(64)	Name of the namespace to which the column belongs.

NOTE

- During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of the **last_analyzed** column in the view is changed to **datea**.
- Currently, the CREATE TABLE syntax does not support the creation of tables with identity columns. When the MY_TAB_COLUMNS view is queried, the value of **identity_column** in the returned result is always **no**.

7.3.13.123 MY_TAB_COMMENTS

MY_TAB_COMMENTS displays comments on all tables and views owned by the current user. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-405**.

Table 7-405 MY_TAB_COMMENTS columns

Name	Type	Description
owner	character varying(64)	Owner of a table or view.
table_name	character varying(64)	Name of a table or view.
comments	text	Comments.
schema	character varying(64)	Name of the namespace to which the table belongs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.124 MY_TAB_HISTOGRAMS

MY_TAB_HISTOGRAMS displays histogram information about the tables and views owned by the current user. All users can access this view. This view exists in both PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-406**.

Table 7-406 MY_TAB_HISTOGRAMS columns

Name	Type	Description
table_name	character varying(128)	Table name.
column_name	character varying(4000)	Column name.
endpoint_number	numeric	Bucket ID of the histogram.
endpoint_value	numeric	Not supported yet. Its value is **NULL**.
endpoint_actual_value	character varying(4000)	Actual value of the bucket endpoint.
endpoint_actual_value_raw	raw	Not supported yet. Its value is **NULL**.
endpoint_repeat_count	numeric	Not supported yet. Its value is **NULL**.
scope	character varying(7)	Not supported. The value is **shared**.

7.3.13.125 MY_TAB_MODIFICATIONS

MY_TAB_MODIFICATIONS displays statistics about modifications to tables owned by the current user since the last statistics collection on the tables. Currently, this view displays only tables on which INSERT, DELETE, and UPDATE operations have been performed. All users can access this view. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-407**.

Table 7-407 MY_TAB_MODIFICATIONS columns

Name	Type	Description
table_name	character varying(128)	Table name.
partition_name	character varying(128)	Partition name.
subpartition_name	character varying(128)	Subpartition name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
inserts	numeric	Approximate number of insertions since the last statistics collection.
updates	numeric	Approximate number of updates since the last statistics collection.
deletes	numeric	Approximate number of deletions since the last statistics collection.
timestamp	date	Last modification time. Currently, the modification time of a partitioned table is not supported. The value is **NULL**. **TIMESTAMP (0) WITHOUT TIME ZONE** is displayed in an O-compatible database.
truncated	character varying(3)	Not supported yet. Its value is **NULL**.
drop_segments	numeric	Not supported yet. Its value is **NULL**.
schema_name	character varying(128)	Name of the schema to which the table belongs.

NOTE

During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of the **timestamp** column in the view is changed to **datea**.

7.3.13.126 MY_TAB_STATS_HISTORY

MY_TAB_STATS_HISTORY provides the table statistics history of the tables owned by the current user. All users can access this view. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-408**.

Table 7-408 MY_TAB_STATS_HISTORY columns

Name	Type	Description
table_name	character varying(128)	Table name.
partition_name	character varying(128)	Not supported yet. Its value is **NULL**.
subpartition_name	character varying(128)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
stats_update_time	timestamp(6) with time zone	Time when statistics are updated. Database restart is not supported. Otherwise, data loss will occur.

7.3.13.127 MY_TAB_STATISTICS

MY_TAB_STATISTICS displays statistics about tables owned by the current user in the database. This view exists in the PG_CATALOG and SYS schemas and all users can access this view. For details about the columns, see **Table 7-409**.

Table 7-409 MY_TAB_STATISTICS columns

Name	Type	Description
table_name	character varying(128)	Table name.
partition_name	character varying(128)	Not supported yet. Its value is **NULL**.
partition_position	numeric	Not supported yet. Its value is **NULL**.
subpartition_name	character varying(128)	Not supported yet. Its value is **NULL**.
subpartition_position	numeric	Not supported yet. Its value is **NULL**.
object_type	character varying(12)	Object type • **TABLE** • **PARTITION** • **SUBPARTITION**
num_rows	numeric	Number of rows in an object.
blocks	numeric	Not supported yet. Its value is **NULL**.
empty_blocks	numeric	Not supported yet. Its value is **NULL**.
avg_space	numeric	Not supported yet. Its value is **NULL**.
chain_cnt	numeric	Not supported yet. Its value is **NULL**.
avg_row_len	integer	Average row length, including the row overhead.
avg_space_freelist_blocks	numeric	Not supported yet. Its value is **NULL**.
num_freelist_blocks	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
avg_cached_blocks	numeric	Not supported yet. Its value is **NULL**.
avg_cache_hit_ratio	numeric	Not supported yet. Its value is **NULL**.
im_imcu_count	numeric	Not supported yet. Its value is **NULL**.
im_block_count	numeric	Not supported yet. Its value is **NULL**.
im_stat_update_time	timestamp(6) without time zone	Not supported yet. Its value is **NULL**.
scan_rate	numeric	Not supported yet. Its value is **NULL**.
sample_size	numeric	Number of samples used for analyzing a table.
last_analyzed	timestamp with time zone	Date when a table was last analyzed. Database restart is not supported. Otherwise, data loss will occur.
global_stats	character varying(3)	Not supported yet. Its value is **NULL**.
user_stats	character varying(3)	Not supported yet. Its value is **NULL**.
stattype_locked	character varying(5)	Not supported yet. Its value is **NULL**.
stale_stats	character varying(7)	Not supported yet. Its value is **NULL**.
notes	character varying(25)	Not supported yet. Its value is **NULL**.
scope	character varying(7)	Not supported. The default value is **SHARED**.

7.3.13.128 MY_TABLES

MY_TABLES displays information about the tables owned by the current user. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-410**.

Table 7-410 MY_TABLES columns

Name	Type	Description
owner	character varying(64)	Table owner.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
table_name	character varying(64)	Table name.
tablespace_name	character varying(64)	Tablespace name of the table.
dropped	character varying	Specifies whether the current table is deleted. • **yes**: deleted. • **no**: not deleted.
num_rows	numeric	Estimated number of rows in the table.
status	character varying(8)	Specifies whether the current table is valid. • **valid**: The current table is valid. • **unusable**: The current table is unavailable.
sample_size	numeric	Number of samples used for analyzing the table.
temporary	character(1)	Specifies whether the table is a temporary table. • **y**: It is a temporary table. • **n**: It is not a temporary table.
pct_free	numeric	Minimum percentage of free space in a block.
ini_trans	numeric	Initial number of transactions.
max_trans	numeric	Maximum number of transactions.
avg_row_len	integer	Average number of bytes in each row.
partitioned	character varying(3)	Specifies whether the table is a partitioned table. • **yes**: It is a partitioned table. • **no**: It is not a partitioned table.
last_analyzed	timestamp with time zone	Last time when the table was analyzed. Database restart is not supported. Otherwise, data loss will occur.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
row_movement	character varying(8)	Specifies whether to allow partition row movement. • **enabled**: Partition row movement is allowed. • **disabled**: Partition row movement is not allowed.
compression	character varying(8)	Specifies whether to enable table compression. • **enabled**: Table compression is enabled. • **disabled**: Table compression is disabled.
duration	character varying(15)	Time elapsed when a temporary table is processed. • **NULL**: The table is not a temporary table. • **sys$session**: The table is a temporary session table. • **sys$transaction**: The table is a temporary transaction table.
logical_replication	character varying(8)	Specifies whether to enable logical replication. • **enabled**: Logical replication is enabled. • **disabled**: Logical replication is disabled.
external	character varying(3)	Specifies whether the table is a foreign table. • **yes**: It is a foreign table. • **no**: It is not a foreign table.
logging	character varying(3)	Specifies whether to log changes made to the table. • **yes**: The database logs changes made to the table. • **no**: The database does not log changes made to the table.
default_collation	character varying(100)	Default collation of a table.
degree	character varying(10)	Number of instances in a scanned table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
table_lock	character varying(8)	Specifies whether to enable table-level lock. • **enabled**: Table-level lock is enabled. • **disabled**: Table-level lock is disabled.
nested	character varying(3)	Specifies whether the table is a nested table. • **yes**: It is a nested table. • **no**: It is not a nested table.
buffer_pool	character varying(7)	Default buffer pool of a table.
flash_cache	character varying(7)	Smart flash cache hint in database for a table block.
cell_flash_cache	character varying(7)	Cell flash cache hint for a table block.
skip_corrupt	character varying(8)	Specifies whether to skip corrupted blocks during table scanning. • **enabled**: Corrupted blocks will be skipped. • **disabled**: Corrupted blocks will not be skipped.
has_identity	character varying(3)	Specifies whether the table has an identity column. • **yes**: It has an identity column. • **no**: It has no identity column.
inmemory	character varying(8)	Not supported. The value is **disabled**.
segment_created	character varying(3)	Specifies whether a table segment has been created. • **yes**: A table segment has been created. • **no**: No table segment has been created.
monitoring	character varying(3)	Specifies whether to track changes made to the table. • **yes**: The database tracks changes made to the table. • **no**: The database does not track changes made to the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
cluster_name	character varying(128)	Not supported yet. Its value is **NULL**.
iot_name	character varying(128)	Not supported yet. Its value is **NULL**.
pct_used	numeric	Not supported yet. Its value is **NULL**.
initial_extent	numeric	Not supported yet. Its value is **NULL**.
next_extent	numeric	Not supported yet. Its value is **NULL**.
min_extents	numeric	Not supported yet. Its value is **NULL**.
max_extents	numeric	Not supported yet. Its value is **NULL**.
pct_increase	numeric	Not supported yet. Its value is **NULL**.
freelists	numeric	Not supported yet. Its value is **NULL**.
freelist_groups	numeric	Not supported yet. Its value is **NULL**.
backed_up	character varying(1)	Not supported yet. Its value is **NULL**.
blocks	numeric	Not supported yet. Its value is **NULL**.
empty_blocks	numeric	Not supported yet. Its value is **NULL**.
avg_space	numeric	Not supported yet. Its value is **NULL**.
chain_cnt	numeric	Not supported yet. Its value is **NULL**.
avg_space_freelist_blocks	numeric	Not supported yet. Its value is **NULL**.
num_freelist_blocks	numeric	Not supported yet. Its value is **NULL**.
instances	character varying(10)	Not supported yet. Its value is **NULL**.
cache	character varying(5)	Not supported yet. Its value is **NULL**.
iot_type	character varying(12)	Not supported yet. Its value is **NULL**.
secondary	character varying(1)	Not supported yet. Its value is **NULL**.
global_stats	character varying(3)	Not supported yet. Its value is **NULL**.
user_stats	character varying(3)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
cluster_owner	character varying(30)	Not supported yet. Its value is **NULL**.
dependencies	character varying(8)	Not supported yet. Its value is **NULL**.
compression_for	character varying(30)	Not supported yet. Its value is **NULL**.
read_only	character varying(3)	Not supported yet. Its value is **NULL**.
result_cache	character varying(7)	Not supported yet. Its value is **NULL**.
clustering	character varying(3)	Not supported yet. Its value is **NULL**.
activity_tracking	character varying(23)	Not supported yet. Its value is **NULL**.
dml_timestamp	character varying(25)	Not supported yet. Its value is **NULL**.
container_data	character varying(3)	Not supported yet. Its value is **NULL**.
inmemory_priority	character varying(8)	Not supported yet. Its value is **NULL**.
inmemory_distribute	character varying(15)	Not supported yet. Its value is **NULL**.
inmemory_compression	character varying(17)	Not supported yet. Its value is **NULL**.
inmemory_duplicate	character varying(13)	Not supported yet. Its value is **NULL**.
duplicated	character varying(1)	Not supported yet. Its value is **NULL**.
sharded	character varying(1)	Not supported yet. Its value is **NULL**.
hybrid	character varying(3)	Not supported yet. Its value is **NULL**.
cellmemory	character varying(24)	Not supported yet. Its value is **NULL**.
containers_default	character varying(3)	Not supported yet. Its value is **NULL**.
container_map	character varying(3)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
extended_data_link	character varying(3)	Not supported yet. Its value is **NULL**.
extended_data_link_map	character varying(3)	Not supported yet. Its value is **NULL**.
inmemory_service	character varying(12)	Not supported yet. Its value is **NULL**.
inmemory_service_name	character varying(1000)	Not supported yet. Its value is **NULL**.
container_map_object	character varying(3)	Not supported yet. Its value is **NULL**.
memoptimize_read	character varying(8)	Not supported yet. Its value is **NULL**.
memoptimize_write	character varying(8)	Not supported yet. Its value is **NULL**.
has_sensitive_column	character varying(3)	Not supported yet. Its value is **NULL**.
admit_null	character varying(3)	Not supported yet. Its value is **NULL**.
data_link_dml_enabled	character varying(3)	Not supported yet. Its value is **NULL**.
object_id_type	character varying(16)	Not supported yet. Its value is **NULL**.
table_type_owner	character varying(128)	Not supported yet. Its value is **NULL**.
table_type	character varying(128)	Not supported yet. Its value is **NULL**.
compress_for	character varying(30)	Not supported yet. Its value is **NULL**.

7.3.13.129 MY_TABLESPACES

MY_TABLESPACES displays the description of tablespaces that store objects owned by users. By default, it is accessible to all users. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-411**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-411 MY_TABLESPACES columns

Name	Type	Description
tablespace_name	character varying(64)	Tablespace name.
block_size	numeric	Not supported yet. Its value is **NULL**.
initial_extent	numeric	Not supported yet. Its value is **NULL**.
next_extent	numeric	Not supported yet. Its value is **NULL**.
min_extents	numeric	Not supported yet. Its value is **NULL**.
max_extents	numeric	Not supported yet. Its value is **NULL**.
max_size	numeric	Not supported yet. Its value is **NULL**.
pct_increase	numeric	Not supported yet. Its value is **NULL**.
min_extlen	numeric	Not supported yet. Its value is **NULL**.
contents	character varying(9)	Not supported yet. Its value is **NULL**.
status	character varying(9)	Tablespace status. The default value is **online**.
logging	character varying(9)	Not supported yet. Its value is **NULL**.
force_logging	character varying(3)	Not supported yet. Its value is **NULL**.
extent_management	character varying(10)	Not supported yet. Its value is **NULL**.
allocation_type	character varying(9)	Not supported yet. Its value is **NULL**.
segment_space_management	character varying(6)	Not supported yet. Its value is **NULL**.
def_tab_compression	character varying(8)	Not supported yet. Its value is **NULL**.
retention	character varying(11)	Not supported yet. Its value is **NULL**.
bigfile	character varying(3)	Not supported yet. Its value is **NULL**.
predicate_evaluation	character varying(7)	Not supported yet. Its value is **NULL**.
encrypted	character varying(3)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
compress_for	character varying(30)	Not supported yet. Its value is **NULL**.
def_inmemory	character varying(8)	Not supported yet. Its value is **NULL**.
def_inmemory_priority	character varying(8)	Not supported yet. Its value is **NULL**.
def_inmemory_distribute	character varying(15)	Not supported yet. Its value is **NULL**.
def_inmemory_compression	character varying(17)	Not supported yet. Its value is **NULL**.
def_inmemory_duplicate	character varying(13)	Not supported yet. Its value is **NULL**.
shared	character varying(12)	Not supported yet. Its value is **NULL**.
def_index_compression	character varying(8)	Not supported yet. Its value is **NULL**.
index_compress_for	character varying(13)	Not supported yet. Its value is **NULL**.
def_cellmemory	character varying(14)	Not supported yet. Its value is **NULL**.
def_inmemory_service	character varying(12)	Not supported yet. Its value is **NULL**.
def_inmemory_service_name	character varying(1000)	Not supported yet. Its value is **NULL**.
lost_write_protect	character varying(7)	Not supported yet. Its value is **NULL**.
chunk_tablespace	character varying(1)	Not supported yet. Its value is **NULL**.

7.3.13.130 MY_TRIGGERS

MY_TRIGGERS displays information about the triggers owned by the current user. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-412**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-412 MY_TRIGGERS columns

Name	Type	Description
owner	character varying(128)	Trigger owner.
trigger_name	character varying(64)	Trigger name.
trigger_type	character varying	Trigger type, which can be **before statement**, **before each row**, **after statement**, **after each row**, or **instead of**.
triggering_event	character varying	Event that triggers a trigger. The value can be **update**, **insert**, **delete**, or **truncate**.
table_owner	character varying(64)	Owner of the table that defines a trigger.
base_object_type	character varying(18)	Basic object type of a trigger, which can be **table** or **view**.
table_name	character varying(64)	Name of the table or view that defines a trigger.
column_name	character varying(4000)	Not supported yet. Its value is **NULL**.
referencing_name	character varying(422)	Not supported. The value is **referencing new as new old as old**.
when_clause	character varying(4000)	Content of the WHEN clause. **trigger_body** can be executed only when the value is **true**.
status	character varying(64)	• **O**: The trigger is enabled in origin or local mode. • **D**: The trigger is disabled. • **R**: The trigger is enabled in replica mode. • **A**: The trigger is always enabled.
description	character varying(4000)	Trigger description, which is used to rebuild a trigger creation statement.
action_type	character varying(11)	Action type of a trigger, which only supports **call**.
trigger_body	text	Statement executed when a trigger is triggered.
crossedition	character varying(7)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
before_statement	character varying(3)	Not supported yet. Its value is **NULL**.
before_row	character varying(3)	Not supported yet. Its value is **NULL**.
after_row	character varying(3)	Not supported yet. Its value is **NULL**.
after_statement	character varying(3)	Not supported yet. Its value is **NULL**.
instead_of_row	character varying(3)	Not supported yet. Its value is **NULL**.
fire_once	character varying(3)	Not supported yet. Its value is **NULL**.
apply_server_only	character varying(3)	Not supported yet. Its value is **NULL**.

7.3.13.131 MY_TYPE_ATTRS

MY_TYPE_ATTRS displays all types of attributes owned by the current user in the database. This view exists in the PG_CATALOG and SYS schemas and all users can access this view. For details about the columns, see **Table 7-413**.

Table 7-413 MY_TYPE_ATTRS columns

Name	Type	Description
type_name	character varying(128)	Data type name.
attr_name	character varying(128)	Attribute name.
attr_type_mod	character varying(7)	Type modifier of an attribute: • **REF** • **POINT**
attr_type_owner	character varying(128)	Owner of an attribute type.
attr_type_name	character varying(128)	Name of an attribute type.
length	numeric	Length of the CHAR attribute, or the maximum length of the VARCHAR and character varying attribute.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
precision	numeric	Decimal precision of a number or DECIMAL attribute, or binary precision of a FLOAT attribute.
scale	numeric	Decimal places for a NUMERIC or DECIMAL attribute.
character_set_name	character varying(44)	Character set name (**CHAR_CS** or **NCHAR_CS**) of an attribute.
attr_no	numeric	Syntax order number or location (not used as an ID number) of an attribute specified in the type specification or CREATE TYPE statement
inherited	character varying(3)	Specifies whether an attribute is inherited from a supertype. • **YES**: It is inherited from a supertype. • **NO**: It is not inherited from a supertype.

7.3.13.132 MY_TYPES

MY_TYPES describes all object types owned by the current user. All users can access this view. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-414**.

Table 7-414 MY_TYPES columns

Name	Type	Description
type_name	character varying(128)	Type name.
type_oid	raw	Type identifier (OID).
typecode	character varying(128)	Type code.
attributes	numeric	Number of attributes in a type.
methods	numeric	Not supported. The value is **0**.
predefined	character varying(3)	Specifies whether the type is a predefined type.
incomplete	character varying(3)	Specifies whether the type is incomplete.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
final	character varying(3)	Not supported yet. Its value is **NULL**.
instantiable	character varying(3)	Not supported yet. Its value is **NULL**.
persistable	character varying(3)	Not supported yet. Its value is **NULL**.
supertype_owner	character varying(128)	Not supported yet. Its value is **NULL**.
supertype_name	character varying(128)	Not supported yet. Its value is **NULL**.
local_attributes	numeric	Not supported yet. Its value is **NULL**.
local_methods	numeric	Not supported yet. Its value is **NULL**.
typeid	raw	Not supported yet. Its value is **NULL**.

7.3.13.133 MY_VIEWS

MY_VIEWS displays information about all views of the current user. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-415**.

Table 7-415 MY_VIEWS columns

Name	Type	Description
owner	character varying(64)	Owner of the view.
view_name	character varying(64)	Name of the view.
text	text	Text of the view.
text_length	integer	Text length of the view.
text_vc	character varying(4000)	View creation statement. This column may truncate the view text. The BEQUEATH clause will not appear as part of the **TEXT_VC** column in this view.
type_text_length	numeric	Not supported yet. Its value is **NULL**.
type_text	character varying(4000)	Not supported yet. Its value is **NULL**.
oid_text_length	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
oid_text	character varying(4000)	Not supported yet. Its value is **NULL**.
view_type_owner	character varying(128)	Not supported yet. Its value is **NULL**.
view_type	character varying(128)	Not supported yet. Its value is **NULL**.
superview_name	character varying(128)	Not supported yet. Its value is **NULL**.
editioning_view	character varying(1)	Not supported yet. Its value is **NULL**.
read_only	character varying(1)	Not supported yet. Its value is **NULL**.
container_data	character varying(1)	Not supported yet. Its value is **NULL**.
bequeath	character varying(12)	Not supported yet. Its value is **NULL**.
origin_con_id	character varying(256)	Not supported yet. Its value is **NULL**.
default_collation	character varying(100)	Not supported yet. Its value is **NULL**.
containers_default	character varying(3)	Not supported yet. Its value is **NULL**.
container_map	character varying(3)	Not supported yet. Its value is **NULL**.
extended_data_link	character varying(3)	Not supported yet. Its value is **NULL**.
extended_data_link_map	character varying(3)	Not supported yet. Its value is **NULL**.
has_sensitive_column	character varying(3)	Not supported yet. Its value is **NULL**.
admit_null	character varying(3)	Not supported yet. Its value is **NULL**.
pdb_local_only	character varying(3)	Not supported yet. Its value is **NULL**.

7.3.13.134 NLS_DATABASE_PARAMETERS

NLS_DATABASE_PARAMETERS lists the permanent NLS parameters of the database server. This view exists in the PG_CATALOG and SYS schemas. It is

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

accessible to all users. Due to different database kernels and parameter setting formats, the query results of the same parameters in this view may be obviously different from those in Oracle Database. For details about the columns, see **Table 7-416**.

Table 7-416 NLS_DATABASE_PARAMETERS columns

Name	Type	Description
parameter	character varying(128)	Parameter name
value	character varying(64)	Parameter value

7.3.13.135 NLS_INSTANCE_PARAMETERS

NLS_INSTANCE_PARAMETERS lists the permanent NLS parameters of the database client. This view exists in the PG_CATALOG and SYS schemas. It is accessible to all users. Due to different database kernels and parameter setting formats, the query results of the same parameters in this view may be obviously different from those in Oracle Database. For details about the columns, see **Table 7-417**.

Table 7-417 NLS_INSTANCE_PARAMETERS columns

Name	Type	Description
parameter	character varying(128)	Parameter name
value	character varying(64)	Parameter value

7.3.13.136 PG_AVAILABLE_EXTENSION_VERSIONS

PG_AVAILABLE_EXTENSION_VERSIONS displays extension versions of certain database features. For details about the columns, see **Table 7-418**.

Table 7-418 PG_AVAILABLE_EXTENSION_VERSIONS columns

Name	Type	Description
name	name	Extension name
version	text	Version name
installed	boolean	Specifies whether this extension version is installed. Otherwise, the value is **false**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
superuser	boolean	Specifies whether only SYSADMINs are allowed to install the extension. Otherwise, the value is **false**.
relocatable	boolean	Specifies whether the extension can be relocated to another schema. Otherwise, the value is **false**.
schema	name	Name of the schema that the extension must be installed into (**NULL** if the extension is partially or fully relocatable)
requires	name[]	Names of prerequisite extensions (**NULL** if none)
comment	text	Comment from the extension's control file.

7.3.13.137 PG_AVAILABLE_EXTENSIONS

PG_AVAILABLE_EXTENSIONS displays the extension information about certain database features. For details about the columns, see **Table 7-419**.

Table 7-419 PG_AVAILABLE_EXTENSIONS columns

Name	Type	Description
name	name	Extension name
default_version	text	Name of the default version (**NULL** if none is specified)
installed_version	text	Currently installed version of the extension (**NULL** if no version is installed)
comment	text	Comment from the extension's control file.

7.3.13.138 PG_CONTROL_GROUP_CONFIG

PG_CONTROL_GROUP_CONFIG displays Cgroup configuration information in a system. Only users with the SYSADMIN permission can query this view. For details about the columns, see **Table 7-420**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-420 PG_CONTROL_GROUP_CONFIG columns

Name	Type	Description
pg_control_group_config	text	Configuration information of the Cgroup

7.3.13.139 PG_CURSORS

PG_CURSORS displays cursors that are currently available. For details about the columns, see **Table 7-421**.

Table 7-421 PG_CURSORS columns

Name	Type	Description
name	text	Cursor name
statement	text	Query statement when the cursor is declared to change.
is_holdable	boolean	**True** if the cursor is holdable (it can be accessed after the transaction that declared the cursor has committed); **false** otherwise
is_binary	boolean	Specifies whether the cursor is declared BINARY. If it is, the value is **TRUE**. Otherwise, the value is **FALSE**.
is_scrollable	boolean	Specifies whether the cursor is scrollable (it allows rows to be retrieved in a nonsequential manner). If it is, the value is **TRUE**. Otherwise, the value is **FALSE**.
creation_time	timestamp with time zone	Timestamp at which the cursor is declared

7.3.13.140 PG_EXT_STATS

PG_EXT_STATS displays extended statistics stored in **PG_STATISTIC_EXT**. The extension statistics means multiple columns of statistics. For details about the columns, see **Table 7-422**.

Table 7-422 PG_EXT_STATS columns

Name	Type	Reference	Description
schemaname	name	**nspname** in **PG_NAMESPACE**	Name of the schema that contains a table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
tablename	name	**relname** in PG_CLASS	Table name.
attname	int2vector	PG_STATISTIC_EXT.stakey	Columns to be combined for collecting statistics.
inherited	boolean	-	Inherited tables are not supported currently. The value of this column is **false**.
null_frac	real	-	Percentage of column combinations that are null to all records.
avg_width	integer	-	Average width of column combinations, in bytes.
n_distinct	real	-	• Estimated number of distinct values in a column combination if the value is greater than 0. • Negative number obtained by multiplying the result calculated by dividing the number of distinct values by the number of rows by –1 if the value is less than 0. For example, **-1** indicates that the number of distinct values is the same as the number of rows for a column combination. 1. The negative form is used because ANALYZE assumes that the number of distinct values increases as the table grows. 2. The positive form is used when the column seems to have a fixed number of possible values. • The number of distinct values is unknown if the value is **0**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
n_dndistinct	real	-	Number of not-null distinct values in the **dn1** column combination. • Exact number of distinct values if the value is greater than 0. • Negative number obtained by multiplying the result calculated by dividing the number of distinct values by the number of rows by –1 if the value is less than 0. For example, if a value in a column combination appears twice in average, **n_dndistinct** equals **-0.5**. • The number of distinct values is unknown if the value is **0**.
most_common_vals	anyarray	-	List of the most common values in a column combination. If this combination does not have the most common values, this column will be **NULL**. None of the most common values in the column is **NULL**.
most_common_freqs	real[]	-	List of the frequencies of the most common values in a column combination. The frequencies are obtained by dividing the number of occurrences of each value by the number of rows. If the value of **most_common_vals** is **NULL**, the value of this column is **NULL**.
most_common_vals_null	anyarray	-	List of the most common values in a column combination. If this combination does not have the most common values, this column will be **NULL**. At least one of the common values in the column is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
most_common_freqs_null	real[]	-	List of the frequencies of the most common values in a column combination. The frequencies are obtained by dividing the number of occurrences of each value by the number of rows. If the value of **most_common_vals_null** is **NULL**, the value of this column is **NULL**.
histogram_bounds	anyarray	-	Boundary value list of the histogram
partitionname	name	**relname** in PG_PARTITION	Name of the partition in the partitioned table. For a non-partitioned table, this column is left blank.
subpartitionname	name	**relname** in PG_PARTITION	Name of the subpartition in the partitioned table. For a distributed system, this column is left blank.

7.3.13.141 PG_GET_SENDERS_CATCHUP_TIME

PG_GET_SENDERS_CATCHUP_TIME displays catchup information of the currently active primary/standby instance sender thread on the DN. For details about the columns, see **Table 7-423**.

Table 7-423 PG_GET_SENDERS_CATCHUP_TIME columns

Name	Type	Description
pid	bigint	Current sender thread ID
lwpid	integer	Current sender lwpid
local_role	text	Local role
peer_role	text	Peer role

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
state	text	Current sender's replication status • **Startup**: startup state. • **Backup**: backup state. • **Catchup**: catch-up state, which indicates that the standby node is catching up with the primary node. • **Streaming**: streaming replication state. When the standby node catches up with the primary node, the replication remains in this state.
type	text	Current sender type • **Wal**: type that logs are written in advance. • **Data**: data type.
catchup_start	timestamp with time zone	Startup time of a catchup task
catchup_end	timestamp with time zone	End time of the catchup task

7.3.13.142 PG_GROUP

PG_GROUP displays the database role authentication and the relationship between group members. For details about the columns, see **Table 7-424**.

Table 7-424 PG_GROUP columns

Name	Type	Description
groname	name	Group name
grosysid	oid	Group ID.
grolist	oid[]	Array containing the IDs of all roles in the group.

7.3.13.143 PG_GTT_STATS

PG_GTT_STATS displays statistics about a single column in all global temporary tables of the current session by calling pg_get_gtt_statistics(). Before using this function, perform ANALYZE on the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-425 PG_GTT_STATS columns

Name	Type	Description
schemaname	name	Schema name.
tablename	name	Name of a global temporary table.
attname	name	Attribute name.
inherited	boolean	Specifies whether to collect statistics for objects that have inheritance relationship.
null_frac	real	Percentage of column entries that are null.
avg_width	integer	Average stored width, in bytes, of non-null entries.
n_distinct	real	Number of distinct, non-null data values in the column.
most_common_vals	text[]	List of the most common values, which is sorted by occurrence frequency.
most_common_freqs	real[]	Frequencies of the most common values.
histogram_bounds	text[]	Data distribution (excluding the most common values) in a frequency histogram description column.
correlation	real	Correlation coefficient.
most_common_elems	text[]	List of the most common element values, which is used for the array type or some other type.
most_common_elem_freqs	real[]	Frequencies of the most common element values.
elem_count_histogram	real[]	Array type histogram.

7.3.13.144 PG_GTT_RELSTATS

PG_GTT_RELSTATS displays basic information about all global temporary tables of the current session by calling pg_get_gtt_relstats(). The query result of the distributed global temporary table on a CN is not meaningful, and the query result on a DN is empty. Therefore, it is not recommended in distributed mode.

Table 7-426 PG_GTT_RELSTATS columns

Name	Type	Description
schemaname	name	Schema name

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
tablename	name	Name of a global temporary table.
relfilenode	oid	ID of a file object.
relpages	integer	Number of disk pages of a global temporary table.
reltuples	real	Number of records in a global temporary table
relallvisible	integer	Number of pages that are marked as all visible
relfrozenxid	xid	All transaction IDs before this one have been replaced with a permanent (frozen) transaction ID in the table.
relminmxid	xid	Reserved

7.3.13.145 PG_GTT_ATTACHED_PIDS

PG_GTT_ATTACHED_PIDS displays which sessions are using global temporary tables by calling the pg_get_attached_pid() function. In distributed mode, GS_GTT_ATTACHED_PIDS or GS_GLOBAL_GTT_ATTACHED_PIDS is recommended.

Table 7-427 PG_GTT_ATTACHED_PIDS columns

Name	Type	Description
schemaname	name	Schema name.
tablename	name	Name of a global temporary table.
relid	oid	OID of a global temporary table.
pids	bigint[]	Thread PID list.
sessionids	bigint[]	Session ID list.

7.3.13.146 PG_INDEXES

PG_INDEXES displays information about each index in the database. For details about the columns, see **Table 7-428**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-428 PG_INDEXES columns

Name	Type	Reference	Description
schemaname	name	**nspname** in PG_NAMESPACE	Name of the schema that contains tables and indexes.
tablename	name	**relname** in PG_CLASS	Name of the table where the index is located.
indexname	name	**relname** in PG_CLASS	Index name
tablespace	name	**nspname** in PG_TABLESPACE	Name of the tablespace that contains the index.
indexdef	text	-	Index definition (a reconstructed **CREATE INDEX** command)

7.3.13.147 PG_LOCKS

PG_LOCKS displays information about locks held by open transactions. For details about the columns, see **Table 7-429**.

Table 7-429 PG_LOCKS columns

Name	Type	Reference	Description
locktype	text	-	Type of the locked object. The value can be **relation**, **extend**, **page**, **tuple**, **transactionid**, **virtualxid**, **object**, **userlock**, **advisory**, or **tablespace**.
database	oid	**oid** in PG_DATABASE	OID of the database in which the locked target exists. • The OID is **0** if the target is a shared object. • The OID is **NULL** if the locked object is a transaction.
relation	oid	**oid** in PG_CLASS	OID of the relationship targeted by the lock (**NULL** if the object is not a relation or part of a relation).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
page	integer	-	Page number targeted by the lock within the relation (**NULL** if the object is not a relation page or row page).
tuple	smallint	-	Row number targeted by the lock within the page (**NULL** if the object is not a row).
bucket	integer	-	Hash bucket ID.
virtualxid	text	-	ID of the virtual transaction The value is **NULL** if the object is not a virtual transaction.
transactionid	xid	-	ID of the transaction. The value is **NULL** if the object is not a transaction.
classid	oid	**oid** in **PG_CLASS**	OID of the system catalog that contains the object (**NULL** if the object is not a general database object).
objid	oid	-	OID of the lock target within its system catalog (**NULL** if the target is not a general database object).
objsubid	smallint	-	Column number for a column in the table. The value is **0** if the object is some other object type. The value is **NULL** if the object is not a general database object.
virtualtransaction	text	-	Virtual ID of the virtual transaction holding or awaiting this lock.
pid	bigint	-	Logical ID of the server thread holding or awaiting this lock. The value is **NULL** if the lock is held by a prepared transaction.
sessionid	bigint	-	ID of the session that holds or waits for the lock.
mode	text	-	Lock mode held or desired by this thread. The value can be **AccessShareLock**, **RowShareLock**, **RowExclusiveLock**, **ShareLock**, **ShareRowExclusiveLock**, **ExclusiveLock**, or **AccessExclusive-Lock**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
granted	boolean	-	• The value is **TRUE** if the lock is a held lock. • The value is **FALSE** if the lock is an awaited lock.
fastpath	boolean	-	The value is **TRUE** if the lock is obtained through **fast-path**, and is **FALSE** if the lock is obtained through the main lock table.
locktag	text	-	Lock information that the session waits for. It can be parsed using the locktag_decode() function.
global_sessionid	text	-	Global session ID.

7.3.13.148 PG_NODE_ENV

PG_NODE_ENV displays the environment variable information of the current node. Only users with the SYSADMIN or MONADMIN permission can query this view. For details about the columns, see **Table 7-430**.

Table 7-430 PG_NODE_ENV columns

Name	Type	Description
node_name	text	Current node name
host	text	Host name of the node
process	integer	Number of the node process
port	integer	Port ID of the node
installpath	text	Installation directory of the node
datapath	text	Data directory of the node
log_directory	text	Log directory of the node

7.3.13.149 PG_OS_THREADS

PG_OS_THREADS provides status information about all the threads under the current node. For details about the columns, see **Table 7-431**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-431 PG_OS_THREADS columns

Name	Type	Description
node_name	text	Current node name
pid	bigint	PID of the thread running under the current node process
lwpid	integer	Lightweight thread ID corresponding to the PID.
thread_name	text	Thread name corresponding to the PID.
creation_time	timestamp with time zone	Thread creation time corresponding to the PID.
dbid	oid	ID of the database to which a thread belongs.

7.3.13.150 PG_PREPARED_STATEMENTS

PG_PREPARED_STATEMENTS displays information about all prepared statements that are available in the current session. For details about the columns, see **Table 7-432**.

Table 7-432 PG_PREPARED_STATEMENTS columns

Name	Type	Description
name	text	Identifier of the prepared statement
statement	text	Query string for creating this prepared statement. For prepared statements created through SQL, this is the PREPARE statement submitted by the client. For prepared statements created through the frontend/backend protocol, this is the text of the prepared statement itself.
prepare_time	timestamp with time zone	Timestamp when the prepared statement is created.
parameter_types	regtype[]	Expected parameter types for the prepared statement in the form of an array of **regtype**. The OID corresponding to an element of this array can be obtained by converting the **regtype** value to an OID value.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
from_sql	boolean	• **True** if the prepared statement was created through the PREPARE statement. • **False** if the statement was prepared through the frontend/backend protocol.

7.3.13.151 PG_PREPARED_XACTS

PG_PREPARED_XACTS displays information about transactions that are currently prepared for two-phase commit. For details about the columns, see **Table 7-433**.

Table 7-433 PG_PREPARED_XACTS columns

Name	Type	Reference	Description
transaction	xid	-	Numeric identifier of the prepared transaction.
gid	text	-	Global identifier of the prepared transaction.
prepared	timestamp with time zone	-	Time at which the transaction is prepared for commit
owner	name	**rolname** in **PG_AUTHID**	Name of the user who executes the transaction.
database	name	**PG_DATABASE**.datname	Name of the database that executes the transaction.

7.3.13.152 PG_REPLICATION_SLOTS

PG_REPLICATION_SLOTS displays replication slot information. For details about the columns, see **Table 7-434**.

Table 7-434 PG_REPLICATION_SLOTS columns

Name	Type	Description
slot_name	text	Replication slot name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
plugin	text	Name of the output plug-in corresponding to the logical replication slot.
slot_type	text	Replication slot type. • **physical**: physical replication slot. • **logical**: logical replication slot.
datoid	oid	OID of the database where the replication slot resides.
database	name	Name of the database where the replication slot resides.
active	boolean	Specifies whether the replication slot is activated. • **t** (true): yes. • **f** (false): no.
xmin	xid	XID of the earliest transaction that the database must reserve for the replication slot.
catalog_xmin	xid	XID of the earliest system catalog-involved transaction that the database must reserve for the logical replication slot.
restart_lsn	text	Physical location of the earliest Xlog required by the replication slot.
dummy_standby	boolean	Reserved parameter.
confirmed_flush	text	Dedicated logical replication slot. The client confirms the location of the received log.
confirmed_csn	xid	Dedicated logical replication slot. The client confirms the CSN corresponding to the last transaction in the received log.

Example:

```
-- Perform query on the DN.
gaussdb=# SELECT * FROM pg_replication_slots;
 slot_name |      plugin      | slot_type | datoid | database | active | xmin | catalog_xmin | restart_lsn |
dummy_standby | confirmed_flush | confirmed_csn
-----------+------------------+-----------+--------+----------+--------+------+--------------+-------------+---------------
+-----------------+---------------
 dn_6002   |                  | physical  |      0 |          | t      |      |              | 0/3622B528  | f             |
 dn_6003   |                  | physical  |      0 |          | t      |      |              | 0/3622B528  | f             |
 slot_lsn  | mppdb_decoding | logical   | 131072 | db_test  | f      |      |        66658 | 0/36252350  | f             |
0/362523D0      |
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
 slot_test | mppdb_decoding | logical   | 131072 | db_test  | f      |      |        66658 | 0/36251718  | f
|               |      10025527
(4 rows)

-- Perform query on the CN.
gaussdb=# SELECT * FROM pg_replication_slots;
 slot_name |      plugin      | slot_type | datoid | database | active | xmin | catalog_xmin | restart_lsn |
dummy_standby | confirmed_flush | confirmed_csn
-----------+----------------+-----------+--------+----------+--------+------+--------------+-------------+---------------
+-----------------+---------------
 slot_test | mppdb_decoding | logical   | 139264 | db_test  | f      |      |              |             | f
|               |
(1 row)
```

NOTICE

When a query is performed on a DN, the **confirmed_csn** query result of the LSN-based logical replication slot is empty, and the **confirmed_flush** query result of the CSN-based logical replication slot is empty. When a query is performed on the CN, **catalog_xmin**, **restart_lsn**, **confirmed_flush**, and **confirmed_csn** of the CSN-based logical replication slot are not displayed and the query result is empty.

7.3.13.153 PG_RULES

PG_RULES displays useful information about rewriting rules. For details about the columns, see **Table 7-435**.

Table 7-435 PG_RULES columns

Name	Type	Description
schemaname	name	Name of the schema that contains a table.
tablename	name	Name of the table to which the rule applies
rulename	name	Rule name
definition	text	Rule definition (reconstructed by the CREATE statement).

7.3.13.154 PG_RUNNING_XACTS

PG_RUNNING_XACTS displays information about running transactions on the current node. For details about the columns, see **Table 7-436**.

Table 7-436 PG_RUNNING_XACTS columns

Name	Type	Description
handle	integer	Handle corresponding to the transaction in GTM.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
gxid	xid	Transaction ID.
state	tinyint	Transaction status. • **3**: prepared. • **0**: starting.
node	text	Node name.
xmin	xid	Minimum transaction ID on the node.
vacuum	boolean	Specifies whether the current transaction is lazy vacuum. • **t** (true): yes • **f** (false): no.
timeline	bigint	Number of database restarts.
prepare_xid	xid	ID of the transaction in the prepared state; otherwise, the value is **0**.
pid	bigint	Thread ID corresponding to the transaction.
next_xid	xid	Transaction ID sent from a CN to a DN.
dbid	oid	ID of the database corresponding to the transaction.

7.3.13.155 PG_SESSION_IOSTAT

PG_SESSION_IOSTAT displays I/O workload management information about the task currently executed by the user. Only the user with the SYSADMIN or MONADMIN permission can query this view. For details about the columns, see **Table 7-437**.

IOPS is counted by 10 thousands per second.

NOTE

If no data is displayed in the queried view, contact Huawei technical support.

Table 7-437 PG_SESSION_IOSTAT columns

Name	Type	Description
query_id	bigint	Job ID.
mincurriops	integer	Minimum I/O of the current job across each DN.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
maxcurriops	integer	Maximum I/O of the current job across each DN.
minpeakiops	integer	Minimum I/O peak at each DN during job running.
maxpeakiops	integer	Maximum I/O peak at each DN during job running.
io_limits	integer	**io_limits** set for the job.
io_priority	text	**io_priority** set for the job.
query	text	Job.
node_group	text	Node group of the user to which a job belongs.
curr_io_limits	integer	Real-time value of **io_limits** when **io_priority** is used to control I/Os.

7.3.13.156 PG_SETTINGS

PG_SETTINGS displays information about parameters of the running database. For details about the columns, see **Table 7-438**.

Table 7-438 PG_SETTINGS columns

Name	Type	Description
name	text	Parameter name.
setting	text	Current parameter value.
unit	text	Unit of the parameter.
category	text	Logical group of the parameter.
short_desc	text	Brief description of the parameter.
extra_desc	text	Detailed description of the parameter.
context	text	Context required to set the parameter value, including **internal**, **postmaster**, **sighup**, **backend**, **superuser**, and **user**.
vartype	text	Parameter type, including **bool**, **enum**, **integer**, **real**, or **string**.
source	text	Method of assigning the parameter value.
min_val	text	Minimum value of the parameter. If the parameter type is not numeric, the value of this column is **null**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
max_val	text	Maximum value of the parameter. If the parameter type is not numeric, the value of this column is **null**.
enumvals	text[]	Valid values of an enum-type parameter. If the parameter type is not enum, the value of this column is **null**.
boot_val	text	Default parameter value used upon the database startup.
reset_val	text	Default parameter value used upon the database reset.
sourcefile	text	Configuration file used to set parameter values. If parameter values are not configured using the configuration file, the value of this column is **null**.
sourceline	integer	Row number of the configuration file for setting parameter values. If parameter values are not configured using the configuration file, the value of this column is **null**.

NOTE

The query results of the PG_SETTINGS columns do not contain empty character strings in ORA-compatible mode, but may contain empty character strings instead of **NULL** in other compatible modes.

7.3.13.157 PG_SHARED_MEMORY_DETAIL

PG_SHARED_MEMORY_DETAIL displays usage information about all the shared memory contexts. For details about the columns, see **Table 7-439**.

Table 7-439 PG_SHARED_MEMORY_DETAIL columns

Name	Type	Description
contextname	text	Name of the memory context.
level	smallint	Hierarchy of the memory context.
parent	text	Name of the parent memory context.
totalsize	bigint	Total size of the shared memory, in bytes.
freesize	bigint	Remaining size of the shared memory, in bytes.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
usedsize	bigint	Used size of the shared memory, in bytes.

7.3.13.158 PG_STATS

PG_STATS displays single-column statistics stored in the pg_statistic table. The GUC parameter **autovacuum_naptime** specifies the interval for updating statistics recorded in the view. For details about the columns, see **Table 7-440**.

Table 7-440 PG_STATS columns

Name	Type	Reference	Description
schemaname	name	**nspname** in **PG_NAMESPACE**	Name of the schema that contains a table.
tablename	name	**relname** in **PG_CLASS**	Table name.
attname	name	**PG_ATTRIBUTE**.attname	Column name.
inherited	boolean	-	Inherited tables are not supported currently. The value **f** (false) indicates no.
null_frac	real	-	Percentage of column entries that are null.
avg_width	integer	-	Average width in bytes of column's entries.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
n_distinct	real	-	• Estimated number of distinct values in the column if the value is greater than 0. • Negative number obtained by multiplying the result calculated by dividing the number of distinct values by the number of rows by –1 if the value is less than 0. For example, value **–1** indicates that the number of distinct values is the same as the number of rows for a unique column. 1. The negated form is used when ANALYZE believes that the number of distinct values is likely to increase as the table grows. 2. The positive form is used when the column seems to have a fixed number of possible values. • The number of distinct values is unknown if the value is **0**.
n_dndistinct	real	-	Number of not-null distinct values in the **dn1** column. • Exact number of distinct values if the value is greater than **0**. • Negative number obtained by multiplying the result calculated by dividing the number of distinct values by the number of rows by –1 if the value is less than 0. For example, if the value of a column appears twice in average, set **n_dndistinct** to **-0.5**. • The number of distinct values is unknown if the value is **0**.
most_common_vals	anyarray	-	List of the most common values in a column. If this column does not have the most common value, the value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
most_common_freqs	real[]	-	List of the frequencies of the most common values in a column. The frequencies are obtained by dividing the number of occurrences of each value by the number of rows. The value is **NULL** if the value of **most_common_vals** is **NULL**.
histogram_bounds	anyarray	-	Frequency histogram consisting of values excluding null values and MCVs. If a value appears in the value of **most_common_vals**, it does not appear in the histogram. If the column data type does not have the < operator or the list specified by **most_common_vals** contains all values of the column, the histogram information of the column is **NULL**.
correlation	real	-	Correlation between the physical row sequence and logical row sequence of a column value. The value ranges from –1 to +1. When the value is close to –1 or +1, the index scan overhead is less than that when the value is close to 0 because random access to the disk is reduced. This column is **NULL** if the column data type does not have a < operator.
most_common_elems	anyarray	-	A list of non-null element values most often appearing
most_common_elem_freqs	real[]	-	List that records the frequency of the most commonly used non-null elements.
elem_count_histogram	real[]	-	A histogram of the counts of distinct non-null element values.
partitionname	name	**relname** in **PG_PARTITION**	Name of the partition in the partitioned table. For a non-partitioned table, this column is left blank.
subpartitionname	name	**relname** in **PG_PARTITION**	Name of the subpartition in the partitioned table. For a distributed system, this column is left blank.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.159 PG_STAT_ACTIVITY

PG_STAT_ACTIVITY displays information about the current user's queries. The columns save information about the last query. For details about the columns, see **Table 7-441**.

NOTE

When PG_STAT_ACTIVITY is queried, GaussDB determines whether to allow the access based on the permissions of **current_user**. If **current_user** is an initial user or a SYSADMIN, they can query information about all sessions. If **current_user** is a common user, they can only view their own session information (**session_user** refers to the information about the current user).

Table 7-441 PG_STAT_ACTIVITY columns

Name	Type	Description
datid	oid	OID of the database that the user session connects to in the background.
datname	name	Name of the database that the user session connects to in the background.
pid	bigint	Background thread ID.
sessionid	bigint	Session ID.
usesysid	oid	OID of the user logged in to the background.
usename	name	Name of the user logged in to the background.
application_name	text	Name of the application connected to the background.
client_addr	inet	IP address of the client connected to the background. If this column is **NULL**, it indicates either the client is connected via a Unix socket on the server or this is an internal thread, such as **AUTOVACUUM**.
client_hostname	text	Host name of the connected client, as reported by a reverse DNS lookup of **client_addr**. This column will be non-null only for IP connections and only when **log_hostname** is enabled.
client_port	integer	TCP port number that the client uses for communication with the background (**-1** if a Unix socket is used).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
backend_start	timestamp with time zone	Time when this session was started, that is, when the client connected to the server.
xact_start	timestamp with time zone	Time when the current active transaction was started (**NULL** if no transaction is active). If the current query is the first of its transaction, the value of this column is the same as that of the **query_start** column.
query_start	timestamp with time zone	Time when the current active query was started, or time when the last query was started if the value of **state** is not **active**. For a stored procedure, function, or package, the first query time is displayed and does not change with the running of statements in the stored procedure.
state_change	timestamp with time zone	Time when **state** was last modified.
waiting	boolean	Specifies whether the background is waiting for a lock. • **t** (true): yes. • **f** (false): no.
enqueue	text	Queuing status of a statement. The value must be one of the following: • **waiting in queue**: The statement is in the queue. • Empty: The statement is running.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
state	text	Overall status of this background. The value must be one of the following: • **active**: The background is executing a query. • **idle**: The background is waiting for a new client command. • **idle in transaction**: The background is in a transaction, but there is no statement being executed in the transaction. • **idle in transaction (aborted)**: The background is in a transaction, but there are statements failed in the transaction. • **fastpath function call**: The background is executing a fast-path function. • **disabled**: This state is reported if **track_activities** is disabled in this background. **NOTE** Common users can view their own session status only. The state information of other accounts is empty. For example, after user **judy** is connected to the database, the state information of user **joe** and the initial user **omm** in **pg_stat_activity** is empty. SELECT datname, usename, usesysid, state,pid FROM pg_stat_activity; datname | usename | usesysid | state | pid ----------+---------+----------+--------+----------------- testdb | omm | 10 | | 139968752121616 testdb | omm | 10 | | 139968903116560 db_tpcds | judy | 16398 | active | 139968391403280 testdb | omm | 10 | | 139968643069712 testdb | omm | 10 | | 139968680818448 testdb | joe | 16390 | | 139968563377936 (6 rows)
resource_pool	name	Resource pool used by the user.
query_id	bigint	ID of a query statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
query	text	Latest query at the background. If the value of **state** is **active**, this column shows the ongoing query. In all other states, it shows the last query that was executed. When a stored procedure is executed, the content displayed in this column is affected by the GUC parameter **track_procedure_sql**. ● When the GUC parameter **track_procedure_sql** is set to **on**, this column also prints the statements that are being executed in the stored procedure. ● When the GUC parameter **track_procedure_sql** is set to **off**, this column prints only the statements that are called by the stored procedure.
connection_info	text	A string in JSON format recording the driver type, driver version, driver deployment path, and process owner of the connected database. For details, see the GUC parameter **connection_info**.
global_sessionid	text	Global session ID.
unique_sql_id	bigint	Unique SQL statement ID. When a stored procedure is executed, the value of this column is affected by the GUC parameter **instr_unique_sql_track_type**. ● When the GUC parameter **instr_unique_sql_track_type** is set to **ALL**, this column displays **unique_sql_id** of statements executed in the stored procedure. ● When the GUC parameter **instr_unique_sql_track_type** is set to **TOP**, this column displays **unique_sql_id** of executed top-level statements.
trace_id	text	Driver-specific trace ID, which is associated with an application request.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
top_xid	xid	Top-level transaction ID of a transaction.
current_xid	xid	Current transaction ID of a transaction.
xlog_quantity	bigint	Size of Xlogs currently used by a transaction, in bytes.

7.3.13.160 PG_STAT_ACTIVITY_NG

PG_STAT_ACTIVITY_NG displays information about all queries in the node group of the current user. For details about the columns, see **Table 7-442**.

Table 7-442 PG_STAT_ACTIVITY_NG columns

Name	Type	Description
datid	oid	OID of the database that the user session connects to in the background.
datname	name	Name of the database that the user session connects to in the background.
pid	bigint	Background thread ID.
sessionid	bigint	Session ID.
global_sessionid	text	Global session ID.
usesysid	oid	OID of the user logged in to the background.
username	name	Name of the user logged in to the background.
application_name	text	Name of the application connected to the background.
client_addr	inet	IP address of the client connected to the background. If this column is **NULL**, it indicates either the client is connected via a Unix socket on the server or this is an internal thread, such as **AUTOVACUUM**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
client_hostname	text	Host name of the connected client, as reported by a reverse DNS lookup of **client_addr**. This column will be non-null only for IP connections and only when **log_hostname** is enabled.
client_port	integer	TCP port number that the client uses for communication with the background (**-1** if a Unix socket is used).
backend_start	timestamp with time zone	Time when this session was started, that is, when the client connected to the server.
xact_start	timestamp with time zone	Time when the current active transaction was started (**NULL** if no transaction is active). If the current query is the first of its transaction, the value of this column is the same as that of the **query_start** column.
query_start	timestamp with time zone	Time when the current active query was started, or time when the last query was started if the value of **state** is not **active**. For a stored procedure, function, or package, the first query time is displayed and does not change with the running of statements in the stored procedure.
state_change	timestamp with time zone	Time when **state** was last modified.
waiting	Boolean	Specifies whether the background is waiting for a lock. • **t** (true): yes. • **f** (false): no.
enqueue	text	Queuing status of a statement. The value can be: • **waiting in queue**: The statement is in the queue. • **Empty**: The statement is running.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
state	text	Overall status of this background. The value can be: • **active**: The background is executing a query. • **idle**: The background is waiting for a new client command. • **idle in transaction**: The background is in a transaction, but there is no statement being executed in the transaction. • **idle in transaction (aborted)**: The background is in a transaction, but there are statements failed in the transaction. • **fastpath function call**: The background is executing a fast-path function. • **disabled**: This state is reported if **track_activities** is disabled in this background. **NOTE** Common users can view their own session status only. The state information of other accounts is empty. For example, after user **judy** is connected to the database, the state information of user **joe** and the initial user **omm** in **pg_stat_activity** is empty. SELECT datname, usename, usesysid, state,pid FROM pg_stat_activity_ng; datname | usename | usesysid | state | pid ----------+----------+----------+--------+----------------- testdb | omm | 10 | | 139968752121616 testdb | omm | 10 | | 139968903116560 db_tpcds | judy | 16398 | active | 139968391403280 testdb | omm | 10 | | 139968643069712 testdb | omm | 10 | | 139968680818448 testdb | joe | 16390 | | 139968563377936 (6 rows)
resource_pool	name	Resource pool used by the user.
query_id	bigint	ID of a query statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
query	text	Latest query at the background. If the value of **state** is **active**, this column shows the ongoing query. In all other states, it shows the last query that was executed.
node_group	text	Node group of the user to which the statement belongs.

7.3.13.161 PG_STAT_ALL_INDEXES

PG_STAT_ALL_INDEXES displays statistics about accesses to a specific index by querying each index row in the current database. For details about the columns, see **Table 7-443**.

Indexes can be used via either simple index scans or bitmap index scans. In a bitmap scan the output of indexes can be combined via AND or OR rules, so it is difficult to associate individual heap row fetches with specific indexes when a bitmap scan is used. Therefore, each bitmap scan increments the **pg_stat_all_indexes.idx_tup_read** count(s) for the index(es) it uses, and it increments the **pg_stat_all_tables.idx_tup_fetch** count for the table, but it does not affect **pg_stat_all_indexes.idx_tup_fetch**.

Table 7-443 PG_STAT_ALL_INDEXES columns

Name	Type	Description
relid	oid	OID of the table that the index is created for.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema that the index is in.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_scan	bigint	Number of index scans initiated on the index.
idx_tup_read	bigint	Number of index entries returned by scans on the index.
idx_tup_fetch	bigint	Number of live rows fetched in the original table by a simple index scan that uses the index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.162 PG_STAT_ALL_TABLES

PG_STAT_ALL_TABLES queries information about each table in the current database (including TOAST tables) and displays statistics about a specific table. For details about the columns, see **Table 7-444**.

Table 7-444 PG_STAT_ALL_TABLES columns

Name	Type	Description
relid	oid	OID of the table.
schemaname	name	Name of the schema that the table is in.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of rows HOT updated (that is, with no separate index update required).
n_live_tup	bigint	Estimated number of live rows.
n_dead_tup	bigint	Estimated number of inactive rows. It indicates the number of inactive row pointers in Ustore tables.
last_vacuum	timestamp with time zone	Last time when the table was cleared.
last_autovacuum	timestamp with time zone	Last time when the table was vacuumed by the autovacuum daemon thread.
last_analyze	timestamp with time zone	Last time when the table was analyzed.
last_autoanalyze	timestamp with time zone	Last time when the table was analyzed by the autovacuum daemon thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
vacuum_count	bigint	Number of times the table is cleared.
autovacuum_count	bigint	Number of times the table has been vacuumed by the autovacuum daemon.
analyze_count	bigint	Number of times the table has been manually analyzed.
autoanalyze_count	bigint	Number of times the table has been analyzed by the autovacuum daemon thread.
last_data_changed	timestamp with time zone	Time when the data in the table changes last time. Operations that cause data changes include insert, update, delete, and truncate on the table and exchange, truncate, and drop on the table partitions (partitions/subpartitions). This column is not recorded in the system catalog. Data in this column is recorded only on the local CN.

7.3.13.163 PG_STAT_BAD_BLOCK

PG_STAT_BAD_BLOCK displays statistics about page verification failures after a node is started. For details about the columns, see **Table 7-445**.

Table 7-445 PG_STAT_BAD_BLOCK columns

Name	Type	Description
nodename	text	Node name.
databaseid	integer	OID of a database.
tablespaceid	integer	Tablespace OID.
relfilenode	integer	File object ID.
bucketid	smallint	ID of the bucket for consistent hashing.
forknum	integer	File type. The options are as follows: • **0**: main data file. • **1**: FSM file. • **2**: VM file. • **3**: BCM file.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
error_count	integer	Number of verification failures.
first_time	timestamp with time zone	Time of the first verification failure.
last_time	timestamp with time zone	Time of the latest verification failure.

7.3.13.164 PG_STAT_BGWRITER

PG_STAT_BGWRITER displays statistics about the activity of the background writer process. For details about the columns, see **Table 7-446**.

Table 7-446 PG_STAT_BGWRITER columns

Name	Type	Description
checkpoints_timed	bigint	Number of scheduled checkpoints that have been performed.
checkpoints_req	bigint	Number of checkpoints that have been performed actively.
checkpoint_write_time	double precision	Total time spent in the checkpoint processing portion when writing files to disk, in milliseconds.
checkpoint_sync_time	double precision	Total time spent in the checkpoint processing portion when synchronizing files to disk, in milliseconds.
buffers_checkpoint	bigint	Number of buffers written by checkpoints.
buffers_clean	bigint	Number of buffers written by the background writer process.
maxwritten_clean	bigint	Number of times that cleanup scanning stops because the background writer process writes too many buffers.
buffers_backend	bigint	Number of buffers written directly by the backend.
buffers_backend_fsync	bigint	Number of times that the backend calls fsync (usually, even if the backend executes these write actions, the background writer process processes them again).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
buffers_alloc	bigint	Number of buffers allocated.
stats_reset	timestamp with time zone	Time at which these statistics were last reset.

7.3.13.165 PG_STAT_DATABASE

PG_STAT_DATABASE displays statistics about each database in cluster. For details about the columns, see **Table 7-447**.

Table 7-447 PG_STAT_DATABASE columns

Name	Type	Description
datid	oid	OID of a database.
datname	name	Database name.
numbackends	integer	Number of backends currently connected to this database. This is the only column in the view that returns the status value. Other columns return the accumulated value since the last reset.
xact_commit	bigint	Number of transactions in this database that have been committed.
xact_rollback	bigint	Number of transactions in this database that have been rolled back.
blks_read	bigint	Number of disk blocks read in this database.
blks_hit	bigint	Number of times that disk blocks have been found in the buffer cache and therefore the disk blocks do not need to be read (only those found in the buffer cache are counted, excluding those found in the file system cache of the OS).
tup_returned	bigint	Number of rows returned by queries in this database.
tup_fetched	bigint	Number of rows fetched by queries in this database.
tup_inserted	bigint	Number of rows inserted by queries in this database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
tup_updated	bigint	Number of rows updated by queries in this database.
tup_deleted	bigint	Number of rows deleted by queries in this database.
conflicts	bigint	Number of queries canceled due to database recovery conflicts (conflicts occurring only on the standby node). For details, see **PG_STAT_DATABASE_CONFLICTS**.
temp_files	bigint	Number of temporary files created by queries in this database. Calculates all temporary files, regardless of why the temporary files are created (such as sorting or hashing) or the setting of the **log_temp_files** parameter.
temp_bytes	bigint	Total amount of data written to temporary files by queries in this database. Calculates all temporary files, regardless of why the temporary files are created or the setting of the **log_temp_files** parameter.
deadlocks	bigint	Number of all deadlocks detected in the database since the last reset of statistics.
blk_read_time	double precision	Time spent reading data file blocks by backends in this database, in milliseconds.
blk_write_time	double precision	Time spent reading data file blocks by backends in this database, in milliseconds.
stats_reset	timestamp with time zone	Time when the status statistics are reset.

7.3.13.166 PG_STAT_DATABASE_CONFLICTS

PG_STAT_DATABASE_CONFLICTS displays statistics about database conflicts. For details about the columns, see **Table 7-448**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-448 PG_STAT_DATABASE_CONFLICTS columns

Name	Type	Description
datid	oid	ID of a database
datname	name	Database name
confl_tablespace	bigint	Number of conflicting tablespaces
confl_lock	bigint	Number of conflicting locks
confl_snapshot	bigint	Number conflicting snapshots
confl_bufferpin	bigint	Number of conflicting buffers
confl_deadlock	bigint	Number of conflicting deadlocks

7.3.13.167 PG_STAT_REPLICATION

PG_STAT_REPLICATION displays information about the log synchronization thread, such as the locations where the sender sends logs and where the receiver receives logs. For details about the columns, see **Table 7-449**.

Table 7-449 PG_STAT_REPLICATION columns

Name	Type	Description
pid	bigint	PID of the thread
usesysid	oid	User system ID.
usename	name	Username
application_name	text	Program name
client_addr	inet	Client address
client_hostname	text	Client name
client_port	integer	Port of the client
backend_start	timestamp with time zone	Start time of the program

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
state	text	Status of the log synchronization thread: • **startup**: The thread is being started. • **catchup**: The thread is establishing a connection between the standby node and the primary node. • **streaming**: The thread has established a connection between the standby node and the primary node and is replicating data streams. • **backup**: The thread is sending a backup. • **stopping**: The thread is being stopped.
sender_sent_location	text	Location of the last Xlog sent by the sender to the receiver.
receiver_write_location	text	Location of the last Xlog written by the receiver to the file system.
receiver_flush_location	text	Location of the last Xlog flushed by the receiver to disk.
receiver_replay_location	text	Location of the last Xlog replayed by the receiver.
sync_priority	integer	Priority of synchronous duplication (**0** indicates asynchronization.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
sync_state	text	Synchronization state: • **async**: asynchronous replication. • **sync**: synchronous replication. • **potential**: The standby node is asynchronous currently, but if a current synchronization node fails, the standby node becomes synchronous. • **quorum**: switches between the synchronous and asynchronous states to ensure that there are more than a certain number of synchronous standby nodes. Generally, the number of synchronous standby nodes is (n +1)/2-1, where *n* indicates the total number of copies. Whether the standby node is synchronous depends on whether logs are received first. For details, see the description of the **synchronous_standby_names** parameter.

7.3.13.168 PG_STAT_SYS_INDEXES

PG_STAT_SYS_INDEXES displays index status information about all the system catalogs in the pg_catalog and information_schema schemas. For details about the columns, see **Table 7-450**.

Table 7-450 PG_STAT_SYS_INDEXES columns

Name	Type	Description
relid	oid	OID of the table that the index is created for.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema that the index is in.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_scan	bigint	Number of index scans initiated on the index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
idx_tup_read	bigint	Number of index entries returned by scans on the index.
idx_tup_fetch	bigint	Number of live rows fetched in the original table by a simple index scan that uses the index.

7.3.13.169 PG_STAT_SYS_TABLES

PG_STAT_SYS_TABLES displays statistics about the system catalogs of all the namespaces in pg_catalog and information_schema schemas. For details about the columns, see **Table 7-451**.

Table 7-451 PG_STAT_SYS_TABLES columns

Name	Type	Description
relid	oid	OID of the table.
schemaname	name	Name of the schema that the table is in.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of rows HOT updated (with no separate index update required).
n_live_tup	bigint	Estimated number of live rows.
n_dead_tup	bigint	Estimated number of inactive rows. It indicates the number of inactive row pointers in Ustore tables.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
last_vacuum	timestamp with time zone	Last time when this table was manually vacuumed (excluding **VACUUM FULL**)
last_autovacuum	timestamp with time zone	Last time when the table was vacuumed by the autovacuum daemon thread.
last_analyze	timestamp with time zone	Last time when the table was manually analyzed.
last_autoanalyze	timestamp with time zone	Last time when the table was analyzed by the autovacuum daemon thread.
vacuum_count	bigint	Number of times the table has been manually vacuumed (not counting **VACUUM FULL**).
autovacuum_count	bigint	Number of times the table has been vacuumed by the autovacuum daemon thread.
analyze_count	bigint	Number of times the table has been manually analyzed.
autoanalyze_count	bigint	Number of times the table has been analyzed by the autovacuum daemon thread.
last_data_changed	timestamp with time zone	Last modification time of the table data.

7.3.13.170 PG_STAT_USER_FUNCTIONS

PG_STAT_USER_FUNCTIONS shows user-defined function status information in a namespace. (The language of the function is non-internal language.) For details about the columns, see **Table 7-452**.

Table 7-452 PG_STAT_USER_FUNCTIONS columns

Name	Type	Description
funcid	oid	Function ID.
schemaname	name	Schema name
funcname	name	Function name
calls	bigint	Number of times the function has been called

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
total_time	double precision	Total time spent in the function and all other functions called by it
self_time	double precision	Total time spent in the function itself, excluding other functions called by it

7.3.13.171 PG_STAT_USER_INDEXES

PG_STAT_USER_INDEXES displays information about the index status of user-defined ordinary tables and TOAST tables. For details about the columns, see **Table 7-453**.

Table 7-453 PG_STAT_USER_INDEXES columns

Name	Type	Description
relid	oid	OID of the table that the index is created for.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema that the index is in.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_scan	bigint	Number of index scans initiated on the index.
idx_tup_read	bigint	Number of index entries returned by scans on the index.
idx_tup_fetch	bigint	Number of live rows fetched in the original table by a simple index scan that uses the index.

7.3.13.172 PG_STAT_USER_TABLES

PG_STAT_USER_TABLES displays information about user-defined ordinary tables and TOAST tables in the namespaces. For details about the columns, see **Table 7-454**.

Table 7-454 PG_STAT_USER_TABLES columns

Name	Type	Description
relid	oid	OID of the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
schemaname	name	Name of the schema that the table is in.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of rows HOT updated (with no separate index update required).
n_live_tup	bigint	Estimated number of live rows.
n_dead_tup	bigint	Estimated number of inactive rows. It indicates the number of inactive row pointers in Ustore tables.
last_vacuum	timestamp with time zone	Last time when this table was manually vacuumed (excluding **VACUUM FULL**)
last_autovacuum	timestamp with time zone	Last time when the table was vacuumed by the autovacuum daemon thread.
last_analyze	timestamp with time zone	Last time when the table was manually analyzed.
last_autoanalyze	timestamp with time zone	Last time when the table was analyzed by the autovacuum daemon thread.
vacuum_count	bigint	Number of times the table has been manually vacuumed (not counting **VACUUM FULL**).
autovacuum_count	bigint	Number of times the table has been vacuumed by the autovacuum daemon thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
analyze_count	bigint	Number of times the table has been manually analyzed.
autoanalyze_count	bigint	Number of times the table has been analyzed by the autovacuum daemon thread.
last_data_changed	timestamp with time zone	Last modification time of the table data.

7.3.13.173 PG_STAT_XACT_ALL_TABLES

PG_STAT_XACT_ALL_TABLES displays transaction status information about all ordinary tables and TOAST tables in the namespaces. For details about the columns, see **Table 7-455**.

Table 7-455 PG_STAT_XACT_ALL_TABLES columns

Name	Type	Description
relid	oid	OID of the table.
schemaname	name	Name of the schema that the table is in.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of rows HOT updated (with no separate index update required).

7.3.13.174 PG_STAT_XACT_SYS_TABLES

PG_STAT_XACT_SYS_TABLES displays transaction status information of the system catalog in the namespace. For details about the columns, see **Table 7-456**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-456 PG_STAT_XACT_SYS_TABLES columns

Name	Type	Description
relid	oid	OID of the table.
schemaname	name	Name of the schema that the table is in.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of rows HOT updated (with no separate index update required).

7.3.13.175 PG_STAT_XACT_USER_FUNCTIONS

PG_STAT_XACT_USER_FUNCTIONS contains statistics on the execution of each function. For details about the columns, see **Table 7-457**.

Table 7-457 PG_STAT_XACT_USER_FUNCTIONS columns

Name	Type	Description
funcid	oid	Function ID.
schemaname	name	Schema name.
funcname	name	Function name.
calls	bigint	Number of times that the function has been called.
total_time	double precision	Total time spent in the function and all other functions called by it.
self_time	double precision	Total time spent in the function itself, excluding other functions called by it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.176 PG_STAT_XACT_USER_TABLES

PG_STAT_XACT_USER_TABLES displays transaction status information of the user table in the namespace. For details about the columns, see **Table 7-458**.

Table 7-458 PG_STAT_XACT_USER_TABLES columns

Name	Type	Description
relid	oid	OID of the table.
schemaname	name	Name of the schema that the table is in.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of rows HOT updated (with no separate index update required).

7.3.13.177 PG_STATIO_ALL_INDEXES

PG_STATIO_ALL_INDEXES displays information about each index row in the current database, showing statistics about I/O on that specific index. For details about the columns, see **Table 7-459**.

Table 7-459 PG_STATIO_ALL_INDEXES columns

Name	Type	Description
relid	oid	OID of the table where the index is located.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema that contains the index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
relname	name	Name of the table where the index is located.
indexrelname	name	Index name.
idx_blks_read	bigint	Number of disk blocks read from the index.
idx_blks_hit	bigint	Number of cache hits in the index.

7.3.13.178 PG_STATIO_ALL_SEQUENCES

PG_STATIO_ALL_SEQUENCES displays the I/O statistics of each sequence in the current database. For details about the columns, see **Table 7-460**.

Table 7-460 PG_STATIO_ALL_SEQUENCES columns

Name	Type	Description
relid	oid	OID of this sequence.
schemaname	name	Name of the schema where the sequence is located.
relname	name	Name of the sequence.
blks_read	bigint	Number of disk blocks read from the sequence.
blks_hit	bigint	Number of cache hits in this sequence.

7.3.13.179 PG_STATIO_ALL_TABLES

PG_STATIO_ALL_TABLES displays the I/O statistics of each table (including TOAST tables) in the current database. For details about the columns, see **Table 7-461**.

Table 7-461 PG_STATIO_ALL_TABLES columns

Name	Type	Description
relid	oid	Table OID.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
heap_blks_read	bigint	Number of disk blocks read from the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
heap_blks_hit	bigint	Number of cache hits in this table.
idx_blks_read	bigint	Number of disk blocks read from all indexes in the table.
idx_blks_hit	bigint	Number of cache hits of all indexes in the table.
toast_blks_read	bigint	Number of disk blocks (if any) read from the TOAST table of this table.
toast_blks_hit	bigint	Number of cache hits (if any) in the TOAST table of this table.
tidx_blks_read	bigint	Number of disk blocks (if any) read from the TOAST table indexes of this table.
tidx_blks_hit	bigint	Number of cache hits (if any) in the TOAST table indexes of this table.

7.3.13.180 PG_STATIO_SYS_INDEXES

PG_STATIO_SYS_INDEXES displays I/O status information for all system catalog indexes in a namespace. For details about the columns, see **Table 7-462**.

Table 7-462 PG_STATIO_SYS_INDEXES columns

Name	Type	Description
relid	oid	OID of the table where the index is located.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema that the index is in.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_blks_read	bigint	Number of disk blocks read from the index.
idx_blks_hit	bigint	Number of cache hits in the index.

7.3.13.181 PG_STATIO_SYS_SEQUENCES

PG_STATIO_SYS_SEQUENCES displays I/O status information about all the sequences in a namespace. For details about the columns, see **Table 7-463**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-463 PG_STATIO_SYS_SEQUENCES columns

Name	Type	Description
relid	oid	OID of this sequence.
schemaname	name	Name of the schema where the sequence is located.
relname	name	Name of the sequence.
blks_read	bigint	Number of disk blocks read from the sequence.
blks_hit	bigint	Number of cache hits in this sequence.

7.3.13.182 PG_STATIO_SYS_TABLES

PG_STATIO_SYS_TABLES shows I/O status information about all the system catalogs in a namespace. For details about the columns, see **Table 7-464**.

Table 7-464 PG_STATIO_SYS_TABLES columns

Name	Type	Description
relid	oid	Table OID.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
heap_blks_read	bigint	Number of disk blocks read from the table.
heap_blks_hit	bigint	Number of cache hits in this table.
idx_blks_read	bigint	Number of disk blocks read from the index in the table.
idx_blks_hit	bigint	Number of cache hits in the table.
toast_blks_read	bigint	Number of disk blocks (if any) read from the TOAST table of this table.
toast_blks_hit	bigint	Number of cache hits (if any) in the TOAST table of this table.
tidx_blks_read	bigint	Number of disk blocks (if any) read from the TOAST table indexes of this table.
tidx_blks_hit	bigint	Number of cache hits (if any) in the TOAST table indexes of this table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.183 PG_STATIO_USER_INDEXES

PG_STATIO_USER_INDEXES displays I/O status information about all the user relationship table indexes in a namespace. For details about the columns, see **Table 7-465**.

Table 7-465 PG_STATIO_USER_INDEXES columns

Name	Type	Description
relid	oid	OID of the table where the index is located.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema that the index is in.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_blks_read	bigint	Number of disk blocks read from the index.
idx_blks_hit	bigint	Number of cache hits in the index.

7.3.13.184 PG_STATIO_USER_SEQUENCES

PG_STATIO_USER_SEQUENCES shows I/O status information about all the user relationship table sequences in the namespaces. For details about the columns, see **Table 7-466**.

Table 7-466 PG_STATIO_USER_SEQUENCES columns

Name	Type	Description
relid	oid	OID of this sequence.
schemaname	name	Name of the schema where the sequence is in.
relname	name	Name of the sequence.
blks_read	bigint	Number of disk blocks read from the sequence.
blks_hit	bigint	Number of cache hits in this sequence.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.185 PG_STATIO_USER_TABLES

PG_STATIO_USER_TABLES displays I/O status information about all the user relationship tables in the namespace. For details about the columns, see **Table 7-467**.

Table 7-467 PG_STATIO_USER_TABLES columns

Name	Type	Description
relid	oid	Table OID.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
heap_blks_read	bigint	Number of disk blocks read from the table.
heap_blks_hit	bigint	Number of cache hits in this table.
idx_blks_read	bigint	Number of disk blocks read from the index in the table.
idx_blks_hit	bigint	Number of cache hits in the table.
toast_blks_read	bigint	Number of disk blocks (if any) read from the TOAST table of this table.
toast_blks_hit	bigint	Number of cache hits (if any) in the TOAST table of this table.
tidx_blks_read	bigint	Number of disk blocks (if any) read from the TOAST table indexes of this table.
tidx_blks_hit	bigint	Number of cache hits (if any) in the TOAST table indexes of this table.

7.3.13.186 PG_TABLES

PG_TABLES is used to query useful information about each table in a database. For details about the columns, see **Table 7-468**.

Table 7-468 PG_TABLES columns

Name	Type	Reference	Description
schemaname	name	**nspname** in **PG_NAMESPACE**	Name of the schema that contains a table.
tablename	name	**relname** in **PG_CLASS**	Table name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Reference	Description
tableowner	name	pg_get_userbyid(**PG_CLASS**.relowner)	Table owner.
tablespace	name	**spcname** in **PG_TABLESPACE**	Tablespace that contains the table. The default value is null.
hasindexes	boolean	**PG_CLASS**.relhasindex	The value is **t** if the table has (or recently had) an index; otherwise, the value is **f**.
hasrules	boolean	**PG_CLASS**.relhasrules	The value is **t** if the table has rules; otherwise, the value is **f**.
hastriggers	boolean	**PG_CLASS**.RELHASTRIGGERS	The value is **t** if the table has triggers; otherwise, the value is **f**.
tablecreator	name	pg_get_userbyid(**PG_OBJECT**.creator)	Table owner.
created	timestamp with time zone	**PG_OBJECT**.ctime	Time when a table is created.
last_ddl_time	timestamp with time zone	**PG_OBJECT**.mtime	Time when the last DDL operation is performed on a table.

7.3.13.187 PG_THREAD_WAIT_STATUS

PG_THREAD_WAIT_STATUS allows you to test the blocking status about the worker thread and auxiliary thread of the current instance. For details about the columns, see **Table 7-469**.

Table 7-469 PG_THREAD_WAIT_STATUS columns

Name	Type	Description
node_name	text	Current node name.
db_name	text	Database name.
thread_name	text	Thread name.
query_id	bigint	Query ID. It is equivalent to **debug_query_id**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
tid	bigint	Thread ID of the current thread.
sessionid	bigint	Current session ID.
lwtid	integer	Lightweight thread ID of the current thread.
psessionid	bigint	Parent session ID.
tlevel	integer	Level of the streaming thread.
smpid	integer	Concurrent thread ID.
wait_status	text	Waiting status of the current thread. For details, see **Table 7-470**.
wait_event	text	If **wait_status** is **acquire lock**, **acquire lwlock**, or **wait io**, this column describes the lock, lightweight lock, or I/O information. Otherwise, this column is empty.
locktag	text	Information about the lock that the current thread is waiting to obtain.
lockmode	text	Lock mode that the current thread is waiting to obtain. The values include modes in the table-level lock, row-level lock, and page-level lock.
block_sessionid	bigint	ID of the session that blocks the current thread from obtaining the lock.
global_sessionid	text	Global session ID.

Table 7-470 shows the waiting status in the **wait_status** column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-470 Waiting states

Value	Description	Cause	Suggestions
wait cmd	Waits for reading network communication packets to complete.	The local end waits for the next command from the peer end.	Network communication is often more time-consuming than other operations. If a connection stays in this state for a long time, you can check why the peer end has not sent commands by examining the pg_stat_activity, gs_stack, dbe_perf.pooler_status, and pg_thread_wait_status views.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
wait pooler get conn	Waits for pooler to obtain connections.	The local node attempts to acquire connections to other nodes from the connection pool. If no suitable connection is found, the local node will try to establish a new connection. If the connection can be reused directly from the connection pool, the time taken to obtain the connection is minimal. When service traffic is high and connections are frequently rebuilt (such as when the connection pool is cleared or connections are not reclaimed due to errors), acquiring connections can be time-consuming.	Modify the GUC parameters **minimum_pool_size** and **pooler_maximum_idle_time** according to the service scenario and review the current connection status using logs. If many connection destruction logs are observed, investigate the root cause.
wait pooler abort conn	Waits for pooler to terminate connections.	This state is activated by the clean connection operation and waits for the transaction that meets the conditions to exit and release the connection.	Stop services before clearing connections.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
wait pooler clean conn	Waits for pooler to clear connections.	This state is activated by the clean connection operation and waits for the thread that meets the conditions to exit and release the connection.	Stop services before clearing connections.
pooler create conn	Waits for the pooler to establish a connection. The connection is being established with the node specified by **nodename**, and there are *N* connections waiting to be established.	If the network connection is poor or the peer service is busy, the connection setup time is prolonged.	Check the network status or reduce the service traffic.
set cmd	Waits for running the **SET**, **RESET**, **TRANSACTION BLOCK LEVEL PARA SET**, or **SESSION LEVEL PARA SET** statement on the connection. The statement is being executed on the node specified by **nodename**.	The **SET**, **RESET**, **TRANSACTION BLOCK LEVEL PARA SET**, or **SESSION LEVEL PARA SET** statement is being executed on the specified node.	Check the network status or reduce the service traffic.
cancel query	Cancels the SQL statement that is being executed through a connection.	After sending a statement cancellation command to the peer end, the local end waits for the execution result from the peer end.	Check the network status or reduce the service traffic.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
stop query	Stops the query that is being executed through a connection.	After sending a statement cancellation command to the peer end, the local end waits for the execution result from the peer end.	Check the network status or reduce the service traffic.
wait node	Waits for receiving data from a connected node. The system is waiting for data from the plevel thread of the node specified by **nodename**. The data of *N* connections is waiting to be returned. If **phase** is included, the possible phases are as follows: • **begin**: The transaction is being started. • **commit**: The transaction is being committed. • **rollback**: The transaction is being rolled back.	After sending the command, the local end waits for the execution result from the peer end.	Network communication is often more time-consuming than other operations. If a connection stays in this state for a long time, you can check why the peer end has not sent commands by examining the pg_stat_activity, gs_stack, dbe_perf.pooler_status, and pg_thread_wait_status views.
wait transaction sync	Waits for synchronizing the transaction specified by **xid**.	The local transaction is waiting for completion.	Modify the **transaction_sync_naptime** parameter.
wait wal sync	Waits for WALs of a specified LSN to be synchronized to the standby node.	Forcible synchronization of primary and standby logs is enabled.	Change the commit mode to local commit.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
wait data sync	Waits for data pages to be synchronized to the standby node.	Forcible synchronization of data pages to the standby instance is enabled.	Disable the mixed streaming replication function (**enable_mix_replication**).
wait wal flush	In the primary node, Xlogs are waiting to be flushed to disks and the primary node is waiting to reach consensus with the standby node in DCF mode. In the standby node, the minimum recovery point is waiting to be updated and the standby node is waiting to reach consensus with the primary node in DCF mode.	Logs are waiting to be flushed to disk or the minimum recovery point is waiting to be updated.	Check the I/O performance.
wait data sync queue	Waits for putting the row-store data pages into the synchronization queue.	Forcible synchronization of data pages to the standby instance is enabled.	Disable the mixed streaming replication function (**enable_mix_replication**).
flush data	Waits for sending data to the plevel thread of the node specified by **nodename**. If **phase** is included, the possible phase is **wait quota**, indicating that the current communication flow is waiting for the quota value.	Data has been sent to other nodes.	Check the network status or reduce the service traffic.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
wait reserve td	The tdslot of the U-heap page is allocated when DML is performed on the U-heap page.	DML operations have been performed on the Ustore table.	Reduce concurrent DML operations on the Ustore table.
wait td rollback	Rolls back the transactions to be rolled back on the tdslot of the U-heap page.	When DML is executed on the Ustore table, no tdslot can be directly used. The tdslot that is aborted and not rolled back needs to be released.	Reduce concurrent DML operations on the Ustore table.
wait available td	Waits for the idle TD of the U-heap and UB-tree pages.	When DML is performed on the Ustore table, no TD can be allocated on the U-heap and UB-tree pages.	Reduce concurrent DML operations on the Ustore table.
wait transaction rollback	Waits for the undo rollback corresponding to a transaction ID to complete.	When DML is performed on the Ustore table, the transaction is aborted.	Reduce the number of abort transactions that perform DML operations on the Ustore table.
prune table	Waits for a heap table to clear historical deleted data.	Page cleanup has been triggered.	Reduce the number of concurrent DML statements in the Ustore table.
prune index	Waits for an index to clear historical deleted data.	The **VACUUM** command is issued.	Reduce parallel VACUUM operations.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
stream get conn	Waits for connecting to the consumer object of the node specified by **nodename** when the stream flow is initialized. There are *N* consumers waiting to be connected.	A connection setup request has been sent to the peer end and no result has returned yet.	View PG_COMM_RECV_STREAM on the peer end to check the logical connection status, or network status.
wait producer ready	Waits for each producer to be ready when the stream flow is initialized. The system is waiting for the procedure of the plevel thread on the node specified by **nodename** to be ready. There are *N* producers waiting to be ready.	The stream consumer waits for the producer to complete the execution and receive the data sent by the producer.	Check the network status or reduce the service traffic.
synchronize quit	Waits for the threads in the stream thread group to exit when the stream plan ends. If information similar to "synchronize quit, total *m*, remain *n*" is displayed in the **wait_status** column, there are **m** concurrent tasks, and **n** of them are not in the unified exit state.	Threads in the stream thread group have not exited yet.	This wait event occurs in the SQL statements of the stream plan and SMP plan. You can add hints for non-stream plans or disable **enable_stream_operator** to avoid this problem.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
wait stream group destroy	Waits for the stream node group to be destroyed.	Waits for destroying the stream node group when the stream plan ends.	This wait event occurs in the SQL statements of the stream plan and SMP plan. You can add hints for non-stream plans or disable **enable_stream_operator** to avoid this problem.
wait stream thread ready	Waits for the stream producer to be ready. It is blocked until its thread ID is successfully set.	The stream producer fails to update the thread ID.	This wait event occurs in the SQL statements of the stream plan and SMP plan. You can add hints for non-stream plans or disable **enable_stream_operator** to avoid this problem.
wait create stream thread	Waits for the stream thread to start.	The stream thread has not started because of reasons such as the upper limit of resources.	This wait event occurs in the SQL statements of the stream plan and SMP plan. You can add hints for non-stream plans or disable **enable_stream_operator** to avoid this problem.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
wait active statement	Waits for job execution under resource and load control.	The number of concurrent requests exceeds the upper limit.	Reduce concurrent operations.
wait memory	Waits for obtaining the memory.	A thread is waiting for the memory.	-
Sort	Currently, the Sort operator is used for sorting.	1. A large amount of data is queried. 2. The plan is not appropriate. For example, the filter criteria are not pushed down.	Modify the SQL statement to adjust the execution plan.
Sort - fetch tuple	Currently, the Sort operator is used for sorting. **fetch tuple** indicates that the Sort operator is obtaining tuples.	1. A large amount of data is queried. 2. The plan is not appropriate. For example, the tables' join reordering is not appropriate.	Modify the SQL statement to adjust the execution plan.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
Sort - write file	Currently, the Sort operator is used for sorting. **write file** indicates that the Sort operator is writing data to disks.	1. A large amount of data is queried. 2. The plan is not appropriate. For example, the filter criteria are not pushed down. 3. The current worker thread is experiencing memory insufficiency due to either inadequate configured memory or memory occupation by other threads.	1. Modify the SQL statement to adjust the execution plan. 2. Increase the memory.
Material	Currently, the Material operator is used.	1. A large amount of data is queried. 2. The plan is not appropriate. For example, the filter criteria are not pushed down.	1. Modify the SQL statement to adjust the execution plan. 2. Increase the memory.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
Material - write file	The **Material** operator is being executed. **write file** indicates that the **Material** operator is writing data to disks.	1. A large amount of data is queried. 2. The plan is not appropriate. For example, the filter criteria are not pushed down. 3. The current worker thread is experiencing memory insufficiency due to either inadequate configured memory or memory occupation by other threads.	1. Modify the SQL statement to adjust the execution plan. 2. Increase the memory.
HashJoin - build hash	The **HashJoin** operator is being executed. In this phase, you need to pay attention to the execution time-consuming. • **build hash**: The **HashJoin** operator is creating a hash table.	1. A large amount of data is queried. 2. The plan is not appropriate. For example, the tables' join reordering is not appropriate.	Modify the SQL statement to adjust the execution plan and add the corresponding index to accelerate the inner hash table building by hash join.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
HashJoin - write file	The **HashJoin** operator is being executed. In this phase, you need to pay attention to the execution time-consuming. • **write file**: The **HashJoin** operator is writing data to disks.	1. A large amount of data is queried. 2. The plan is not appropriate. For example, the tables' join reordering is not appropriate. 3. The current worker thread is experiencing memory insufficiency due to either inadequate configured memory or memory occupation by other threads.	1. Modify the SQL statement to adjust the execution plan. 2. Increase the memory.
HashAgg - build hash	The **HashAgg** operator is being executed. In this phase, you need to pay attention to the execution time-consuming. • **build hash**: The **HashAgg** operator is creating a hash table.	1. A large amount of data is queried. 2. The plan is not appropriate. For example, the tables' join reordering is not appropriate.	1. Modify the SQL statement to adjust the execution plan. 2. Increase the memory.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
HashAgg - write file	The **HashAgg** operator is being executed. In this phase, you need to pay attention to the execution time-consuming. • **write file**: The **HashAgg** operator is writing data to disks.	1. A large amount of data is queried. 2. The plan is not appropriate. For example, the tables' join reordering is not appropriate. 3. The current worker thread is experiencing memory insufficiency due to either inadequate configured memory or memory occupation by other threads.	1. Modify the SQL statement to adjust the execution plan. 2. Increase the memory.
HashSetop - build hash	The **HashSetop** operator is being executed. In this phase, you need to pay attention to the execution time-consuming. • **build hash**: The **HashSetop** operator is creating a hash table.	1. A large amount of data is queried. 2. The plan is not appropriate. For example, the tables' join reordering is not appropriate.	Modify the SQL statement to adjust the execution plan.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
HashSetop - write file	The **HashSetop** operator is being executed. In this phase, you need to pay attention to the execution time-consuming. • **write file**: The **HashSetop** operator is writing data to disks.	1. A large amount of data is queried. 2. The plan is not appropriate. For example, the tables' join reordering is not appropriate. 3. The current worker thread is experiencing memory insufficiency due to either inadequate configured memory or memory occupation by other threads.	1. Modify the SQL statement to adjust the execution plan. 2. Increase the memory.
create index	Waits for index creation.	An index has been created.	Do not create indexes on non-key columns or columns with low differentiation.
analyze	Executes ANALYZE for the table specified by **relname**. If **phase** is included, the possible phase is **autovacuum**, indicating that the database automatically enables the autovacuum thread to execute **ANALYZE**.	Automatic or manual analysis has been triggered.	Modify **default_statistics_target** to reduce the sampling rate, or increase **autovacuum_analyze_scale_factor** and **autovacuum_analyze_threshold** to reduce the analysis interval.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
vacuum	Executes VACUUM for the table specified by **relname**. If **phase** is included, the possible phase is **autovacuum**, indicating that the database automatically enables the autovacuum thread to execute **VACUUM**.	The **VACUUM** or **ANALYZE** command has been issued.	Cancel the **VACUUM** or **ANALYZE** operation. For the autovacuum thread, you can reduce the number of concurrent DML statements on the page.
vacuum full	Executes **VACUUM FULL** for the table specified by **relname**.	The **VACUUM FULL** command has been issued.	Avoid repeated update operations on the table to prevent table bloats.
vacuum gpi	Clears the global partitioned index of a partitioned table during the VACUUM or autovacuum process.	A partitioned table has a GPI. After the DDL statement is executed on the partitioned table, dead tuples generated by the GPI depend on autovacuum/ VACUUM for asynchronous clearance.	Reduce the DDL frequency of the partitioned table or use the local index.
gtm connect	Waits for connecting to GTM.	The connection to the GTM is being established.	Check the GTM and network status.
gtm get csn	Waits for obtaining the CSN from GTM.	A transaction is committed but the CSN has not been obtained from the GTM.	Check the GTM and network status.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
gtm get snapshot	Waits for obtaining transaction snapshots from GTM.	A transaction is committed but the snapshots have not been obtained from the GTM.	Check the GTM and network status.
gtm create sequence	Waits for GTM to create a sequence.	A message has been sent to the GTM to create a sequence.	Check the GTM and network status.
gtm alter sequence	Waits for GTM to modify a sequence.	A message has been sent to the GTM to modify the sequence.	Check the GTM and network status.
gtm get sequence val	Waits for obtaining the next value of a sequence from GTM.	A message has been sent to the GTM to obtain the sequence.	Check the GTM and network status.
gtm set sequence val	Waits for GTM to set a sequence value.	A message has been sent to the GTM to set the sequence.	Check the GTM and network status.
gtm drop sequence	Waits for GTM to delete a sequence.	A message has been sent to the GTM to delete a sequence.	Check the GTM and network status.
gtm rename sequence	Waits for GTM to rename a sequence.	A message has been sent to the GTM to rename a sequence.	Check the GTM and network status.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
gtm set disaster cluster	Waits for the static configuration information of the standby cluster for DR to be set in the GTM.	A standby cluster for DR is set up, and a switchover is performed between the primary and standby clusters, or the GTM communication is abnormal.	In normal cases, the waiting time is less than 1s. If this wait event persists for a long time, check whether the communication between GaussDB and GTM is normal.
gtm get disaster cluster	Waits for obtaining the static configuration information of the standby cluster for DR from the GTM.	A standby cluster for DR is set up, and a switchover occurs between the primary and standby clusters. The topology of the primary cluster node changes, or the GTM communication is abnormal.	In normal cases, the waiting time is less than 1s. If this wait event persists for a long time, check whether the communication between GaussDB and GTM is normal.
gtm del disaster cluster	Waits for the static configuration information of the standby cluster for DR to be deleted in the GTM.	A standby cluster for DR is set up, and a switchover is performed between the primary and standby clusters, or the GTM communication is abnormal.	In normal cases, the waiting time is less than 1s. If this wait event persists for a long time, check whether the communication between GaussDB and GTM is normal.
gtm set standby read info	Waits for GTM to write the read on standby information.	The standby node reads the CSN information and commit time from the GTM.	Check the GTM and network status.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
gtm get standby read info	Waits for obtaining the read on standby information from the GTM.	The standby node obtains the last information from the GTM.	Check the GTM and network status.
gtm set target standby info	Reserved for new features.	Reserved for new features.	Reserved for new features.
gtm get target standby info	Reserved for new features.	Reserved for new features.	Reserved for new features.
gtm reset standby read csn	Reserved for new features.	Reserved for new features.	Reserved for new features.
wait sync consumer next step	The consumer waits for the next iteration.	Waits for the data of the next iteration during the execution of the RecursiveUnion operator. This is available only in distributed mode.	Set **enable_stream_recursive** to **off**.
wait sync producer next step	The producer waits for the next iteration.	Waits for the next iteration during the execution of the producer corresponding to the RecursiveUnion operator. This function is available only in distributed mode.	Set **enable_stream_recursive** to **off**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
gtm set consistency point	Waits for the GTM to set the global consistency snapshot of the standby cluster for DR.	When the standby cluster for DR is running properly, the CN collects and calculates consistency snapshots and reports them to the GTM.	In normal cases, the waiting time is less than 1s. If this wait event persists for a long time, check whether the communication between GaussDB and GTM is normal.
wait sync bgworkers	Waits for an index worker thread created in parallel to complete local scanning and sorting.	Parallel index creation has been triggered.	Disable parallel index creation.
flush local buffer for GTT	Waits for the local buffer to flush dirty pages.	The local buffer is waiting for dirty page flushing.	Reduce the service volume and check the I/O performance.
standby get snapshot	Waits for obtaining snapshots in the read on standby scenario during parallel replay.	In parallel replay mode, if the standby node is replaying page cleanup logs when the query on the standby node obtains snapshots, it is important to wait until the replay is finished to avoid potential data inconsistencies.	Reduce the execution frequency of **VACUUM** on the host or enable the ultimate RTO replay mode.
recovery conflict with lock	The level-8 lock log replay on the standby node conflicts with the query. The system waits until the query is complete.	The primary node executes DDL statements.	Reduce the DDL execution frequency on the primary node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
recovery conflict with snapshot	The page cleanup log replay on the standby node conflicts with the query. The system waits until the query is complete.	The primary node executes VACUUM.	Reduce the VACUUM execution frequency on the primary node.
recovery conflict with tablespace	The tablespace deletion log replay on the standby node conflicts with the query. The system waits until the query is complete.	The primary node deletes the tablespace.	Reduce the DDL execution frequency on the primary node.
recovery conflict with database	The database deletion log replay on the standby node conflicts with the query. The system waits until the query is complete.	The primary node deletes the database.	Reduce the DDL execution frequency on the primary node.
recovery conflict with truncate	The file truncation log replay on the standby node conflicts with the query. The system waits until the query is complete.	The primary node generates file truncation logs after executing VACUUM.	Reduce the VACUUM execution frequency on the primary node.
recovery conflict with standby query timeout	When the ultimate RTO read on standby is performed, the query times out and is canceled.	The query times out.	Reduce the query time.
recovery conflict with force recycle	When the ultimate RTO read on standby is performed, forcible recycling occurs, and the query is canceled.	The space usage of historical files for ultimate RTO read on standby exceeds the threshold.	Increase the value of the GUC parameter **max_standby_base_page_size**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
recovery conflict with buffer pin	The replayer thread is resolving the conflict between the query and replay of the buffer pin type.	Database restoration conflicts with buffer locking.	Optimize the buffer management policy to reduce the conflict between the restoration operation and buffer locking.
wait io control	Waits for job execution under I/O control.	I/O operations are restricted by the system control mechanism, such as I/O resource quota or priority control.	Check and adjust the I/O control policy to ensure that key I/O operations can be performed in a timely manner.
wait gs_sleep	Waits for the server thread delay.	Internal scheduling mechanism of the database, which is used to control the execution pace of threads.	Check the configuration parameters of the database. If necessary, adjust the thread scheduling policy.
wait vacuum delay	Waits for VACUUM delay.	The automatic vacuum process delays the execution, or the manual vacuum operation waits for a proper execution window.	Adjust the vacuum execution policy. For example, increase the frequency of automatic vacuum or execute the vacuum when the system load is low.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
wait seq scan	Execution time of seq scan.	The database performs sequential scans due to lack of indexes or insufficient query optimization.	Optimize the query statement and add proper indexes to reduce full table scanning and improve the query efficiency.
wait index scan	Execution time of index scan.	The database performs index scan because the index design is improper or the query condition does not match the index.	Check the index validity and query conditions, optimize the index structure, and improve the index scan efficiency.
wait checkpoint start	Start time of the checkpoint phase.	The checkpoint operation of the database starts due to an I/O performance bottleneck.	Optimize the I/O performance, adjust the checkpoint frequency and time, and reduce the I/O wait time.
wait checkpoint done	End time of the checkpoint phase.	The checkpoint operation of the database ends due to an I/O performance bottleneck.	Optimize the I/O performance and adjust the checkpoint configuration to reduce the waiting time.
wait xact start command	Start time of xact.	At the beginning of a transaction, a transaction log bottleneck or lock wait occurs.	Optimize the transaction log processing to reduce the transaction start delay.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
wait xact commit command	Commit time of xact.	In the transaction commit phase, a lock wait or transaction log bottleneck occurs.	Optimize the transaction processing process to reduce lock contention and improve the transaction log processing efficiency.
Accept client conn - Total Time	Time for waiting for a client connection.	Client connection time is collected.	Reduce service concurrency.
Accept client conn - ThrdPool - add epoll	Waits for the listening events to join the epoll.	Statistics on joined listening events are collected.	Reduce service concurrency.
Accept client conn - ThrdPool - wait worker	Wait for the worker to process the session.	Statistics on the time for waiting for a worker are collected.	Reduce service concurrency.
Accept client conn - ThrdPool - init session	Waits for session initialization.	Statistics on the session initialization time are collected.	Reduce service concurrency.
Accept client conn - Worker - init proc	Wait for proc initialization.	Statistics on the proc initialization time are collected.	Reduce service concurrency.
Accept client conn - Worker - init session	Waits for session initialization.	Statistics on the session initialization time are collected.	Reduce service concurrency.
security audit write pipe	Waits for writing audit logs to the pipe.	The security audit log system encounters a bottleneck when writing logs.	Optimize the audit log system, add resources, or adjust the audit policy to reduce the write wait time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Value	Description	Cause	Suggestions
wait ugin cleanup proc	Waits for clearing the UGIN index buffer.	The UGIN index buffer is being cleared.	This is called when the UGIN index buffer is cleared. You are advised to reduce the value of **gin_pending_list_limit**.
wait gin cleanup proc	Waits for clearing the GIN index buffer.	The GIN index buffer is being cleared.	This is called when the GIN index buffer is cleared. You are advised to reduce the value of **gin_pending_list_limit**.
wait fetch undo record	Waits for reading the target undo record.	A transaction is rolled back synchronously or asynchronously.	Reduce transaction aborts in DML operations.
deadlock check	Waits for deadlock detection to complete.	If the lock waiting time exceeds the configured deadlock detection time, deadlock detection is triggered.	Optimize the service process to reduce lock conflicts, or increase the value of **deadlock_timeout** to avoid frequent deadlock detection.
invalid wait event	Unexpected wait event.	An unexpected wait event occurs.	Contact Huawei technical support.

If **wait_status** is **acquire lwlock**, **acquire lock**, or **wait io**, there is an event performing I/O operations or waiting for obtaining the lightweight lock corresponding to **wait_event** or transaction lock.

Table 7-471 describes the corresponding wait events when **wait_status** is **acquire lwlock**. If **wait_event** is **extension**, the lightweight lock is dynamically allocated and is not monitored.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-471 List of wait events corresponding to lightweight locks

wait_event	Description	Cause	Suggestions
OidGenLock	Used to prevent different threads from generating the same OID.	The OID allocation concurrency is high.	Reduce concurrent operations.
XidGenLock	Used to prevent two transactions from obtaining the same XID.	The XID allocation concurrency is high.	Reduce concurrent DDL and DML operations.
ProcArrayLock	Used to prevent concurrent access to or concurrent modification on the ProcArray shared array.	There are a large number of threads in the system.	Reduce the number of concurrent threads.
SInvalReadLock	Used to prevent concurrent execution with invalid message deletion.	A read lock is added when invalid messages in the invalid queue are read.	Reduce the DDL execution frequency and the number of invalid message sending times.
SInvalWriteLock	Used to prevent concurrent execution with invalid message write and deletion.	A write lock is added when an invalid message is inserted into an invalid queue or a non-read-only operation is performed on an invalid queue.	Reduce the DDL execution frequency and the number of invalid message sending times.
WALBufMappingLock	Used to protect the mapping between WAL cache pages and LSN offsets in the shared memory to prevent records from being inserted into uninitialized WAL files.	Inserting records and initializing WAL files are concurrent.	Increase the wal_buffers size.
WALWriteLock	Used to prevent concurrent write from a WAL buffer to a disk.	WAL files are written to disks concurrently.	Reduce the service volume and check the I/O performance.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
ControlFileLock	Used to prevent concurrent read/write or concurrent write/write on the **pg_control** file.	The **pg_control** file is concurrently accessed in scenarios such as database initialization, checkpoint, backup and restoration, log archiving, and primary and standby synchronization.	Reduce concurrent access to the **pg_control** file.
CheckpointLock	Used to prevent multi-checkpoint concurrent execution.	The checkpoint logic has been triggered.	Reduce concurrency.
MultiXactGenLock	Used to allocate a unique MultiXact ID in serial mode.	A large number of concurrent MultiXact IDs are allocated.	Reduce concurrent DML operations.
MultiXactOffsetControlLock	Used to prevent concurrent read/write or concurrent write/write on **pg_multixact/offset**.	The number of concurrent **pg_multixact/offset** operations is too large.	Reduce concurrency.
MultiXactMemberControlLock	Used to prevent concurrent read/write or concurrent write/write on **pg_multixact/members**.	The number of concurrent **pg_multixact/members** operations is too large.	Reduce concurrency.
RelCacheInitLock	Used to add a lock before an operation is performed on the **init** file when messages are invalid.	When the relcache is initialized or the relcache init file is invalidated, the init file is locked.	Do not frequently destroy the creation thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
CheckpointerCommLock	Used to send file flush requests to a checkpointer. The request structure needs to be inserted to a request queue in serial mode.	After dirty pages are generated during dirty page flushing, shrink, and truncate operations, sync requests are collected.	Reduce the number of concurrent operations.
TablespaceCreateLock	Used to check whether a tablespace exists.	A tablespace is created or dropped.	Reduce the number of concurrent DROP or CREATE operations on a tablespace.
BtreeVacuumLock	Used to prevent VACUUM from clearing pages that are being used by B-tree indexes.	The **VACUUM** command is issued.	Reduce parallel VACUUM operations.
AutovacuumLock	Used to access the autovacuum worker array in serial mode.	Autovacuum Launcher is scheduling worker threads.	Adjust the Autovacuum worker thread information. No optimization is required.
AutovacuumScheduleLock	Used to distribute tables requiring VACUUM in serial mode.	The Autovacuum thread determines whether the table needs to be cleared.	Reduce concurrent DML operations on the Astore table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
AutoanalyzeLock	Used to obtain and release resources related to a task that allows for autoanalyze execution.	When a plan is generated, if no statistics have been collected in the base table, the statistics are automatically collected. A maximum of 10 threads can be used to automatically collect statistics on a single node at the same time. This lightweight lock is used to control the resource allocation for automatic statistics collection.	If a large number of conflicts exist in the lightweight lock, a large number of tables whose reltuples/relpages is 0 exist in the service database for a long time. In this case, you do not need to automatically collect statistics. You can disable the autoanalyze parameter for optimization.
SyncScanLock	Used to determine the start position of a relfilenode during heap scanning.	The **synchronized sequential scan** command is issued.	Reduce concurrent scanning operations.
BarrierLock	Used to ensure that only one thread is creating a barrier at a time.	Two threads are creating barriers at the same time.	Reduce the number of create barrier operations.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
NodeTableLock	Protects the shared structure that stores CN and DN information.	During distributed scale-out or scale-in, the pgxc_pool_reload function is called to update the cluster topology cached in the node pool. An exclusive lock is required. In this case, the read lock in the distributed statement parsing and execution phases is blocked.	Perform cluster scaling during off-peak hours to reduce service conflicts.
PoolerLock	Used to prevent two threads from simultaneously obtaining the same connection from a connection pool.	The lock obtains a connection from the connection pool.	Generally, this wait event is not triggered during stable service running. If the alarm is triggered frequently, reduce the service volume.
AlterPortLock	Used to protect the coordinator node from changing the registration port number.	The port number is registered during thread initialization.	Generally, this wait event is not triggered during stable service running. If the alarm is triggered frequently, check whether the background thread is frequently started.
RelationMappingLock	Used to wait for the mapping file between system catalogs and storage locations to be updated.	**pg_filenode.map** is concurrently created.	Reduce concurrent access to pg_filenode.map.
SerializableXactHashLock	Used to prevent concurrent read/write or concurrent write/write on a sharing structure for serializable transactions.	There is concurrent access to serializable transactions.	Reduce concurrent access to serializable transactions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
SerializableFinishedListLock	Used to prevent concurrent read/write or concurrent write/write on a shared linked list for completed serial transactions.	There is concurrent access to serializable transactions.	Reduce concurrent access to serializable transactions.
SerializablePredicateLockListLock	Used to protect a linked list of serializable transactions that have locks.	There is concurrent access to serializable transactions.	Reduce concurrent access to serializable transactions.
OldSerXidLock	Used to protect a structure that records serializable transactions that have conflicts.	There is concurrent access to serializable transactions.	Reduce concurrent access to serializable transactions.
FileStatLock	Used to protect a data structure that stores statistics file information.	When reading files, the storage system calls pgstat to report statistics.	The size of the built-in statistics array is fixed and will reach the limit quickly. No workaround is required.
SyncRepLock	Used to protect Xlog synchronization information during primary/standby replication.	Waits for the standby node to return the Xlog flushing information.	The delay is related to the network delay between the primary and standby nodes and the I/O performance of the standby node.
DataSyncRepLock	Used to protect data page synchronization information during primary-standby replication.	Waits for the standby node to return the data page flushing information.	Disable the mixed streaming replication function (**enable_mix_replication**).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
ExtensionConnectorLibLock	1. Used to add a lock when a specific dynamic library is loaded or uninstalled in ODBC connection initialization scenarios. (deprecated for EC) 2. Heterogeneous database links are used to initialize OCI connections. They are locked when OCI-related dynamic libraries are loaded or uninstalled.	Heterogeneous database links are used to connect to the Oracle Database.	Do not use heterogeneous database links.
GTMHostInfoLock	Used to prevent concurrent access to or concurrent modification on GTM host information.	There is concurrent access to the GTM host.	Check the GTM and network status.
ReplicationSlotAllocationLock	Used to add a lock when a primary node allocates streaming replication slots during primary-standby replication.	Create or clear a replication slot, or change the replication slot status.	Used only when a replication slot is created or cleared and the replication slot status is changed. No optimization is required.
ReplicationSlotControlLock	Used to prevent concurrent update of streaming replication slot status during primary-standby replication.	The replication slot status has been changed.	Used only when the replication slot status is changed. No optimization is required.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
FullBuildXlogCopyStartPtrLock	Controls the concurrent access and storage of the start position of full build to protect Xlog replication operations in full build.	Full build	Used for the build operation. No optimization needed.
LogicalReplicationSlotPersistentDataLock	Used to control the concurrent sequence of logical replication slot persistence, which protects data modified in the logical replication slot during logical replication.	The logical replication slot data needs to be modified.	Used only when the logical replication slot data is modified. No optimization is required.
ResourcePoolHashLock	Used to prevent concurrent access to or concurrent modification on a resource pool table, a hash table.	A lock is applied for when the resource pool hash table is read or updated.	The lock is applied for when a resource pool is added, deleted, queried, or modified. The lock is frequently applied for during peak hours. You can reduce the two operations to reduce the possibility of lock waiting.
WorkloadCGroupHashLock	Used to prevent concurrent access to or concurrent modification on a hash table that contains Cgroup information.	A lock is applied for when the Cgroup information hash table is read or updated.	Reduce service pressure.
OBSGetPathLock	Used to prevent concurrent read/write or concurrent write/write on an OBS path.	-	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
WorkloadUserInfoLock	Used to prevent concurrent access to or concurrent modification on a hash table that contains user information about workload management.	A lock is applied for when the workload management user information hash table is read or updated.	Reduce service pressure.
WorkloadIOUtilLock	Used to protect a structure that records **iostat** and CPU load information.	A lock is applied for when load information such as IOSTAT and CPU is read or updated.	Reduce service pressure.
WorkloadNodeGroupLock	Used to prevent concurrent access to or concurrent modification on a hash table that contains node group information in memory.	A lock is applied for when node group information is read or updated.	Reduce service pressure.
JobShmemLock	Used to protect global variables in the shared memory that is periodically read during a scheduled job where MPP is compatible with Oracle Database.	Lock for the scheduler thread and worker thread to access the scheduled task list.	Disable scheduled tasks.
OBSRuntimeLock	Obtains environment variables, for example, *GAUSSHOME*.	-	-
LLVMDumpIRLock	Used to export the assembly language for dynamically generating functions.	After the **enable_codegen_print** parameter is enabled, this wait event occurs when the codegen is triggered.	Set **enable_codegen_print** to **off** (default value). This parameter is enabled only for fault locating.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
LLVMParseIRLock	Used to compile and parse a finished IR function from the IR file at the start position of a query.	When the query triggers the codegen execution, the wait event occurs. Generally, the wait time is very short.	Set **enable_codegen** to **off**.
RPNumberLock	Deprecated.	Deprecated.	Deprecated.
ClusterRPLock	Deprecated.	Deprecated.	Deprecated.
WaitCountHashLock	Used to protect a shared structure in user statement counting scenarios.	When updating or querying the wait count hash table, a wait may occur due to lock contention.	Optimize the access logic of the wait count hash table to reduce lock contention or optimize the query performance.
InstrWorkloadLock	Protects the access to the memory hash table that stores workload management statistics.	A lock is requested when the hash table that stores workload transaction statistics is read or updated.	Use the GUC parameter **enable_resource_track** to disable the workload transaction statistics function.
CBMParseXlogLock	Protection lock used when CBM parses Xlogs to control the concurrent access and storage of CBM files.	Two threads are using the lock at the same time.	Try again.
DelayDDLLock	Used to control the concurrent sequence of DDL and delay backup function of the deletion.	DDL delay is triggered.	After the DDL delay operation is complete, perform the disable delay DDL operation.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
BadBlockStatHashLock	Used to protect the concurrent access and storage of the hash table that stores statistics about damaged shared pages.	When the bad block statistics are updated or queried, the wait may occur due to lock contention.	Optimize the access frequency of bad block statistics, reduce lock contention, or increase the degree of parallelism for processing bad blocks.
RowPageReplicationLock	Used to control the concurrency sequence of row-store page replication and is used to manage data page replication of row store.	-	-
RcvWriteLock	Used to control the concurrent sequence of the WAL receiver writer to prevent concurrent calls to WalDataRcvWrite.	WAL data is written.	Reduce the service volume and check the I/O performance.
InstanceTimeLock	Used to protect the global variable **g_instance.stat_cxt.gInstanceTimeInfo**, which is used to collect statistics on the node-level time model.	Node-level time model information is refreshed or read.	No workaround is required for node-level time model indicator statistics.
PercentileLock	Used to protect the global percentile buffers.	A lock is applied for when the global cache of percentile statistics or results is read or updated.	Disable the percentile function or increase the calculation period.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
XlogRemoveSegLock	Used to control the concurrent access and storage of the latest deleted WAL file information to protect the recycling of Xlog segment files.	The Xlog of the checkpoint is replayed.	Reduce checkpoints.
DnUsedSpaceHashLock	Used to update space usage information corresponding to a session.	A lock is applied for when the space usage information hash table corresponding to a session is read or updated.	Reduce sessions.
CsnMinLock	Used to control the concurrent access and storage of CSN MIN.	There are a large number of threads in the system.	Reduce the number of concurrent threads.
GPCClearLock	Used to protect the clearing of the global plan cache hash table.	GPC cleans up plans to be eliminated.	This is a normal GPC operation. You can manually clear the data in the dbe_perf.global_plancache_clean view or disable **enable_global_plancache**.
CaptureViewFileHashLock	Used to protect the concurrent access and storage of the hash table when capturing the performance view.	Collect data from multiple tables at the same time.	It is mainly used to locate slow SQL statements. No workaround is required.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
UndoZoneLock	Used to lock the bitmap of the undo zone, which protects the concurrent access and storage of the undo zone.	Each worker thread needs to be allocated with an undo zone. The undo zone allocation is managed by the bitmap. During allocation, the bitmap needs to be locked to ensure that each thread is allocated to a different undo zone.	Reduce the number of concurrent DML operations on the Ustore table.
RollbackReqHashLock	Used to lock the asynchronous rollback hash table, which manages concurrent access and storage of the hash table that stores the rollback request information in the shared memory.	The asynchronous rollback mechanism performs rollback in the background, which does not affect the execution of other statements. During asynchronous rollback, you need to add an asynchronous rollback task to a task hash table. Operations on hash tables must be locked to prevent concurrency problems.	Minimize rollback transactions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
HypoIndexLock	Used to create, delete and reset the virtual index by the lightweight lock.	This prevents address access exceptions caused by concurrent addition and deletion of global-level virtual indexes in multiple sessions.	This function is disabled by default. It is used during SQL optimization and does not affect database performance.
WALWritePaxosLock	Used to protect the concurrent sequence of WALs written to the paxos replication component.	This avoids Xlog disorder caused by concurrent write operations of multiple threads.	Generally, this lock is required only when the Xlog flushing thread is called non-concurrently or the WAL buffer is full. This does not affect the performance.
BlockchainVersionLock	Used to control the concurrent read and write of global block numbers in the ledger database.	In the tamper-proof table feature of the ledger database, to ensure uniqueness of a block number in the global blockchain table, an incremental block number needs to be generated by means of locking when each record is generated.	To avoid frequent addition, deletion, and modification operations on the ledger table, you can adjust the scope of the tamper-proof table based on service requirements and use the sharding policy to reduce concurrent modification on the global blockchain table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
GlobalPrevHashLock	Used to control the concurrent read and write of global verification hash tables in the ledger database.	In a tamper-proof table feature of the ledger database, to ensure that chain tampering verification information can be generated between a record in the global blockchain table and a previous row, each record needs to be locked when being generated to generate a continuous hash value. A locking mechanism can ensure security of concurrent write.	To avoid frequent addition, deletion, and modification operations on the ledger table, you can adjust the scope of the tamper-proof table based on service requirements and use the sharding policy to reduce concurrent modification on the global blockchain table.
PldebugLock	Used for the lock processing operation in the stored procedure debugging function.	The pldebugger is turned on or off.	-
LocalBarrierBufferLock	Controls the modification of the local barrier file to avoid concurrent update.	The local barrier files are updated at the same time.	Wait until the update operation is complete.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
RoleIdLock	Used to control concurrent read and write operations on the pg_user_status system catalog.	Providing password protection policies. When a user logs in to the system using an incorrect password, the system catalog pg_user_status is updated to record the number of login failures and update the lock status. This ensures the security of database accounts.	Login using an incorrect password is insecure. The possible cause is that the password is changed but the user is not aware of the change, or malicious users continuously attempt to change the database password. The database administrator can periodically check the locking status of all users and analyze the locked accounts to reduce the number of login attempts with incorrect passwords. The following situations are involved: 1. If the database is attacked, the administrator must analyze the cause in a timely manner, find out the attack source, and take corresponding measures. 2. If the password is changed but the user is unaware of the change, the system prompts the user to change the password in a timely manner to ensure account security.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
InstrStmtTrackCtlLock	Used to protect the concurrent access and storage of the hash table when full SQL statements are dynamically enabled.	In the dynamic full SQL capture scenario, different unique query IDs are frequently tracked, or the same unique query ID is frequently tracked and untracked.	It is mainly used to locate slow SQL statements. No workaround is required.
HashUidLock	Protects the addition, deletion, modification, and query of UIDs.	A table with users is accessed.	-
ParallelDecodeLock	A lock is used to allocate global slots for parallel decoding to protect the concurrent sequence of parallel decoding.	Parallel decoding is not used.	Reduce the startup of parallel decoding services.
XLogMaxCSNLock	Used to protect the maximum volume of CSNs that can be restored locally in DR mode.	There are a large number of threads in the system.	Reduce the number of concurrent threads.
DisasterCacheLock	Lock of the node mapping table in the standby cluster for DR, which protects the concurrent access and storage of the DR information cache in the shared memory.	A standby cluster for DR is set up, a switchover is performed between the primary and standby clusters, the topology of the nodes in the primary cluster changes, or the node mapping is rebuilt.	This is a normal scenario and does not require optimization.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
MaxCSNArrayLock	Used to protect the restoration progress of standby node CSNs in each shard in the shared memory.	DR read is triggered.	-
RepairBadBlockStatHashLock	Used to protect the concurrent access and storage of the hash table that stores damaged pages in the shared memory.	CRC corruption or LSN corruption is detected, and the page corruption information is recorded.	Prevent bad blocks.
ReplicationOriginLock	Locks are used for creating, deleting, and changing the status of publication and subscription.	Publication and subscription are used.	Reduce the startup of publication and subscription.
LogicalRepWorkerLock	Locks are used for starting, stopping, and associating the background replayer thread in the publication and subscription.	Publication and subscription are used.	Reduce the startup of publication and subscription.
XGBoostLibLock	Used by DB4AI to call the XGBoost library.	The xgboost open-source library is incompatible with the openEuler system and needs to be dynamically called. The dlopen needs to be locked during loading.	The loading of the xgboost library is a passive call to special events. If no DB4AI-related operations (such as model creation and prediction task execution) are performed, the lock does not affect the database performance.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
NgroupDestoryLock	Adds a lock to the concurrent modification of the Node Group hash table.	When operations such as pgxc_pool_reload are performed to clear the entire hash table, an exclusive (X) lock is added. The X lock is mutually exclusive with other S locks, and a wait event occurs.	This wait event occurs only when pgxc_pool_reload or other operations involving node groups are performed.
SQLAdvisorLock	Used to manage the concurrent access and storage of recommended hash tables in distribution key management.	-	-
AuditIndexFileLock	Used to protect concurrent read and write of index files in the audit log.	Audit log management depends on the audit index information in the memory. The aging and rotation of audit logs in the audit thread update the index information. The worker thread reads the index information during audit query or deletion. Locks are used to ensure that the index information is correctly updated or read.	In normal cases, the critical area for reading and writing audit index information is displayed only when audit query and deletion are triggered. If a lock is waiting for a long time, you can stop the audit query or deletion, or use the GUC parameter to control the speed of generating audit logs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
GsStackLock	Used to ensure that the gs_stack function is not concurrently called.	Preventing gs_stack from concurrent calls.	The lock is applied for only when gs_stack is used. If gs_stack is not used, the performance is not affected. No workaround is required.
DropArchiveSlotLock	Used to protect the concurrent sequence of deleting the archive slot.	Concurrent operations are performed on the archive slot at the same time.	Try again.
AboCacheLock	Controls the concurrent read and write of the ABO multi-column intelligent statistics model cache.	When a multi-column intelligent statistics model is trained and called, the system attempts to cache and read the cached model to optimize performance. A lock is required to ensure that the model is correctly read.	Set **enable_ai_stats** to **off**.
DistributeDecodeSlotLock	The creation of replication slots for distributed logical decoding is mutually exclusive.	Distributed logical decoding is used concurrently.	-
ExrtoSnapshotLock	Locks the read on standby snapshot in ultimate RTO mode.	The lock is held when a snapshot is generated during normal replay. When the query is canceled during DDL log replay, the lock may be held for a long time.	Reduce the DDL execution frequency on the primary node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
LocalBarrierRecordFlushLock	Flushes the barrier information in the memory to disk.	The barrier flushing operation is called at the same time.	Wait until the disk flushing operation is complete.
ClogTruncateLock	Locks the concurrent Clog file delete and read/write operations.	Avoid the concurrent Clog file delete and read/write errors.	-
RedoTruncateLock	Lock used when a truncated file is replayed.	Avoid concurrent checkpoint and truncate operations.	Infrequent DDL operations have little impact on performance.
SysToastOidGeneratorLock	Generates the OID information of the TOAST table if the TOAST table is involved during fine-grained restoration.	The IMPDP table operation is called at the same time.	Wait until the fine-grained restoration is complete.
RedoStatisticLock	Locks the DFX statistics replay.	Avoid concurrent query and modification.	-
DisasterCSNInfoListLock	Locks the CSN and LSN mapping table read by the ultimate RTO DR.	When barrier logs are replayed, the mapping between CSNs and LSNs is generated.	The lock is not held for a long time. The operation is lightweight.
WorkloadRuleLock	Locks workloads to prevent concurrent operations.	A lock is added when workload resources are read and written.	Reduce service concurrency.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
AcmFeedbackLock	Controls the concurrent read and write of global variables related to the operator collection of the feedback cardinality estimation model.	During operator collection and model training, the collected global information about the operator linked list and operator quantity needs to be maintained. Locks are required to ensure that the information is correct.	Set **enable_adaptive_cost** to **off**.
AcmFeedbackModelCacheLock	Controls the concurrent read and write of the ABO-based feedback cardinality estimation model cache.	When the feedback cardinality estimation model is trained and called, performance is optimized by caching and attempting to read cached models. When this operation is performed, a lock needs to be added to ensure that the model is correctly read.	Set **enable_adaptive_cost** to **off**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
AcmParameterLock	Controls the concurrent read and write of global variables during the training and update of the intelligent cost estimation model.	When the intelligent cost model is trained and updated, the model parameters in the global variable *acm_trainer* need to be updated. A lock needs to be added to ensure that the update and reading are correct.	Set **enable_adaptive_cost** to **off**.
ArchiveTaskLock	Adds or deletes tasks to or from the archive task queue during archiving.	Multiple archive tasks add data to or delete data from the archive queue at the same time.	Wait for the next task.
TruncateFileLock	Lock held when the standby node replays truncated file logs.	The primary node generates file truncation logs after executing VACUUM.	Do not delete a large amount of primary node data in a short period of time.
TmpBarrierBufferLock	Clears and merges local barriers.	There are multiple operations of clearing and merging local barriers.	Wait until the local barrier is cleared and merged.
RedoBarrierBufferLock	Locks and updates the barrier information in the memory.	Information in the redo barrier is read or updated.	Wait until the update is complete.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
AdioToggleLock	AdioToggleLock adds a write lock when the ADIO feature is enabled, and it adds a shared lock when pread_wrapper and pwrite_wrapper are enabled.	When the ADIO feature is enabled, the read and write calls are mutually exclusive.	Non-performance-sensitive lock, which does not need to be optimized.
LedgerRepairLock	Controls the concurrent execution of archive and repair operations in the ledger database.	The locking mechanism secures the concurrent writing of the archive and repair operations in the ledger database.	Do not perform concurrent archive and repair operations on the ledger table.
ConsistencyVerifyStatHashLock	Protects the concurrent access and storage of the hash table used for data-index consistency check entry registration in the shared memory.	The data-index consistency check is performed.	Avoid data index inconsistency (for example, fault injection) and concurrent data-index consistency check.
PreLogicalStandbyControlLock	Controls the concurrency of logical standby system function execution and replay flushing.	Physical-to-logical conversion or logical replay.	Non-performance-sensitive lock, which does not need to be optimized.
LogicalStandbyMetaFlushLock	Controls the concurrent write operations when the physical-to-logical switchover is prepared for decoding metadata.	Physical-to-logical switchover to collect decoding metadata.	Non-performance-sensitive lock, which does not need to be optimized.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
LogicalStandbyMetaControlLock	Controls concurrent modification of the logical standby metadata directory.	Concurrent metadata maintenance is controlled during logical replay.	Non-performance-sensitive lock, which does not need to be optimized.
WALReceiverDieLock	walrcv exit lock.	The concurrent walrcv exits are prevented when data is written to the buffer.	Non-performance-sensitive lock, which does not need to be optimized.
TSDictionaryDefinitionLock	Global lock of the dictionary system catalog gs_ts_dict_definition.	The mutual exclusion between read and write operations on the gs_ts_dict_definition system catalog is controlled to ensure the integrity and consistency of the dictionary content.	Reduce concurrent ALTER TEXT SEARCH, gs_ts_dict_add_definition, and cache loading operations on TokenWeight dictionaries.
RespoolBufferContextBlockLock	Buffer pool lock.	The buffer pool is read and written.	Reduce service concurrency.
ParallelBuildIdentifierLock	Lock for allocating build IDs during parallel build.	Parallel build is enabled.	The lock is not held for a long time. The operation is lightweight.
ParallelBuildQueueLock	Lock for processing the file name queue during parallel build.	Parallel build is enabled.	The lock is not held for a long time. The operation is lightweight.
InstrStmtGstraceCtlLock	Prevents GSTrace from being concurrently called when it registers a trace function.	GSTrace is prevented from being concurrently called when it registers a trace function.	This lock is requested only when the GSTrace trace API is called. No workaround is required.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
InstrStmtGstraceSQLCtlLock	Prevents GSTrace from being concurrently called when it registers a trace SQL statement.	GSTrace is prevented from being concurrently called when it registers a trace SQL statement.	This lock is requested only when the GSTrace SQL registration API is called or when SQL statements are registered and traced. No workaround is required.
InconsistencyAlarmHashLock	Protects the concurrent access and storage of the hash table used for data-index consistency check entry registration in the shared memory.	The data-index consistency check is performed.	Avoid data index inconsistency (for example, fault injection) and concurrent data-index consistency check.
StackAddrToNameLock	Prevents gs_stack from being concurrently called when it converts symbolic addresses to symbolic names.	gs_stack is prevented from being concurrently called when it converts symbolic addresses to symbolic names.	This lock is requested only when gs_stack or the perf flame graph function is called. No optimization is required.
BufMappingLock	Used to protect operations on a table mapped to shared buffer.	Partition lock of the hash table mapped to buffer.	Adjust the number of partitions to avoid excessive conflicts.
BucketBufMappingLock	Partition lock of the private buffer during bucket scale-out.	Private buffer contention during bucket scale-out.	This is involved only in the scale-out process and does not affect services. No adjustment method is available.
LockMgrLock	Used to protect a common lock structure.	Competing to write to the primary lock table when obtaining a regular lock.	Modify the GUC parameter **num_internal_lock_partitions**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
PredicateLockMgrLock	Used to protect a lock structure that has serializable transactions.	Used to protect a lock structure that has serializable transactions.	-
OperatorRealTLock	Used to prevent concurrent access to or concurrent modification on a global structure that contains real-time data at the operator level.	-	-
OperatorHistLock	Used to prevent concurrent access to or concurrent modification on a global structure that contains historical data at the operator level.	When updating or querying the operator history, a wait may occur due to lock contention.	Optimize the logic for accessing historical operator information to reduce lock contention or optimize the query performance.
SessionRealTLock	Used to prevent concurrent access to or concurrent modification on a global structure that contains real-time data at the query level.	A lock is applied for when the hash table of real-time session statistics is read or updated.	Reduce the number of sessions.
SessionHistLock	Used to prevent concurrent access to or concurrent modification on a global structure that contains historical data at the query level.	A lock is applied for when the hash table of historical session statistics is read or updated.	Reduce the number of sessions.
InstanceRealTLock	Used to protect the update of the hash table that stores shared instance statistics.	A lock is requested when the instance statistics hash table is read or updated.	Reduce service pressure.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
CSNBufMappingLock	Used to control the concurrent access and storage of mapping in the shared CSNlog page.	There is concurrent access to the CSN page.	Reduce the number of concurrent transactions.
CLogBufMappingLock	Used to manage the cache of commit logs.	Used to manage the cache of commit logs.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
UniqueSQLMappingLock	Unique SQL statistics are stored in the unique SQL hash table in the memory. UniqueSQLMappingLock is set to LWLOCK to protect data in the hash table.	Unique SQL statistics are stored in the unique SQL hash table in the memory. The current version has 128 hash partitions. When data is updated in a partition, an S lock needs to be added to the corresponding partition to prevent a memory problem from being triggered by a deletion action during the update. In this case, if a new statement template falls into the partition, the statement template can be added only after the update operation is complete and the lock is released. When a statement template needs to be added to or deleted from a hash table, an exclusive (X) lock needs to be added to the partition to avoid adding, deleting, and writing operations on the partition. In this case, any	Generally, statement templates are not added or deleted during stable service running. Therefore, the wait event is not triggered. If the wait event occurs frequently, perform the following operations to avoid this problem: 1. If there are a large number of select in statements in the service model, you can set **instr_unique_sql_combination_options** to **'in_clause'**. Principle: Normalize multiple parameters in the SELECT IN () statement to reduce the number of newly generated statement templates and the number of exclusive (X) locks added to the hash table. Impact: The **unique_sql_id** in the original SELECT IN () statement changes. If the SQL patch has been installed for this type of statement, you need to collect statistics on the new ID of the corresponding statement and install the patch again. 2. If the wait event is caused by cause 2 ("[UniqueSQL] clean unique sql finished." is frequently printed in logs), run the **reset_unique_sql**

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
		operation on the statement information in the partition needs to be performed after the insertion and deletion are complete. The possible causes of UniqueSQLMappingLock wait event are as follows: 1. A large number of new statement templates are generated due to high concurrency. 2. The unique_sql reset function reset_unique_sql() is called when the service concurrency is high.	command during off-peak hours. Principle: The unique SQL hash table is frequently updated during peak hours. The reset operation adds an exclusive (X) lock to the hash table, causing large-scale blocking. The reset operation during off-peak hours can reduce the impact of the lock on services. Impact: none 3. If the wait event is caused by cause 2 ("[UniqueSQL] Auto-cleanup over, %d uniquesqls are recycled." is frequently printed in logs), set **enable_auto_clean_unique_sql** to **off** or increase **instr_unique_sql_count**. Principle: Disabling the automatic eviction function can prevent the hash table from being locked and deleted during the running. Increasing the value of **instr_unique_sql_count** can increase the capacity of the unique SQL hash table and reduce the frequency of automatic eviction. Impact: If the automatic elimination function is disabled, new statement templates will not be recorded after the hash table is full. As a result,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
			some statistics are lost (the existing templates can still be recorded). If **instr_unique_sql_count** is increased, the memory usage of the unique_sql hash table increases. You need to modify **instr_unique_sql_count** based on the current memory usage. 4. If a large number of services are frequently blocked by UniqueSQLMapping-Lock and the preceding conditions are not met, disable the unique SQL function and set **instr_unique_sql_count** to **0**. Principle: Manually disable the unique SQL function. After the function is disabled, the unique SQL hash table is not operated and the wait event is not triggered. Impact: The loss of statement-level statistics affects fault locating. (The SQL statistics panel in the full SQL, ASP, and WDR reports depends on the unique SQL function. If the unique SQL function is disabled, the preceding functions are affected.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
InstrUserLockId	Used to protect the concurrent modification of the hash table that stores user login and logout records.	The hash table is concurrently used when user login and logout information is read or updated.	This lock is applied for when a user logs in to or out of the system or queries user login and logout information. This reduces the number of users or prevents users from logging in to or out of the system at the same time, reducing the possibility of waiting for the lock.
GPCMappingLock	Used to manage the global plan cache.	The plan has been obtained, saved, or cleared upon expiration.	This is a normal lock obtaining operation during GPC execution. The write lock may take a long time during plan expiration clearance. You can reduce the DDL frequency or disable **enable_global_plancache**.
ProcXactMappingLock	Used to protect the concurrent access and storage of the hash table that stores the mapping information between transaction IDs and threads.	There are a large number of threads in the system.	Reduce the number of concurrent threads.
ASPMappingLock	Used to manage the concurrent access and storage of the ASP hash table.	The ASP obtains data from the uniqueSQL.	The ASP is used to obtain the uniqueSQL data. No workaround is required.
RecoveryXidMappingLock	Protects the shared memory required for recovering two-phase transactions.	A prepared transaction exists before the fault occurs.	Reduce the number of prepare commit transactions before the process is restarted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
BufferIOLock	Used to protect the concurrent I/O for loading or evicting a single page in the shared buffer.	Access lock of the hash table mapped to buffer.	Reduce concurrent I/O operations on the same page.
BufferContentLock	Used to protect the concurrent read and modification operations on a single page in the shared buffer.	Buffer page content lock.	Reduce concurrent read and write operations on the same page.
BucketBufferIOLock	I/O lock of the private buffer during bucket scale-out.	The private buffer is contended for during bucket scale-out.	-
BucketBufferContentLock	Content lock of the private buffer during bucket scale-out.	The private buffer is contended for during bucket scale-out.	-
UndoPerZoneLock	Used to protect the concurrent access and storage of each information in undo zone.	Each worker thread is bound to an undo zone. When the metadata of the undo zone is modified, the undo zone needs to be locked to prevent concurrent modification of the worker thread and recycling thread.	Reduce DML operations on the Ustore table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
UndoSpaceLock	Used to protect the concurrent access and storage of undo spaces.	Each undo zone is bound to an undo space to manage physical files. When the metadata of the undo space is modified, the undo space needs to be locked to prevent concurrent modification of the worker thread and recycling thread.	Reduce DML operations on the Ustore table.
PGPROCLock	Used to protect the concurrent access and storage of global shared thread array.	A lock conflict occurs between the process of allocating proc resources when a connection is created or the connection exits and the process of obtaining snapshots.	Avoid frequent connection establishment.
ReplicationSlotLock	Used to protect a replication slot.	A replication slot is flushed to disk.	Background tasks do not need to be optimized.
Async Ctl	Used to protect the shared memory structure of the asynchronous control module.	Data consistency is ensured during concurrent access of multiple processes.	Reduce concurrent operations.
CLOG Ctl	Protects concurrent access and storage of Clogs.	Each node needs to access the Clog to update the transaction state information.	Reduce the number of concurrent transactions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
CSNLOG Ctl	Protects concurrent access and storage of CSN logs.	Each node needs to update the CSN of a transaction to the CSN log when committing the transaction.	Reduce the number of concurrent transactions.
MultiXactOffset Ctl	Protects concurrent access and storage of transaction ID offsets in multixact logs.	When multiple transactions operate a tuple at the same time, multixact IDs need to be allocated and stored in logs.	Reduce the number of transactions that modify or lock the same tuple.
MultiXactMember Ctl	Protects concurrent access and storage of transaction IDs in multixact logs.	When multiple transactions operate a tuple at the same time, multixact IDs need to be allocated and stored in logs.	Reduce the number of transactions that modify or lock the same tuple.
OldSerXid SLRU Ctl	Protects the shared memory of serialized transactions.	When the transaction isolation level is set to serialization, multiple transactions modify tuples at the same time.	Reduce the scenarios where serialized transactions modify the same tuple.
WALInsertLock	Used to prevent concurrent execution with WAL insertion.	WAL records are inserted.	Reduce the service volume and check the I/O performance.
SnapshotBlockLock	Used to control the concurrent sequence of snapshot-based backup and disk flushing. During snapshot backup, the doublewrite page cannot be flushed to disks.	Snapshot backup is being performed.	Wait until the snapshot backup is complete.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
DoubleWriteLock	Used to manage doublewrite operations.	Doublewrite files are written.	The number of pagewriters is no longer maintained and is equal to the number of doublewrite files.
DWSingleFlushFirstLock	Used to control the concurrent sequence of non-segment-page single-page doublewrite files.	Single-page doublewrite files are written.	Avoid concurrent page eviction.
DWSingleFlushSecondLock	Used to control the concurrent sequence of segment-page single-page doublewrite files.	Single-page doublewrite files are written.	Avoid concurrent page eviction.
DWSingleFlushSecondBufTagLock	Used to control the concurrent access and storage of metadata of segment-page single-page doublewrite files.	Single-page doublewrite files are written.	Avoid concurrent page eviction.
RestartPointQueueLock	Incremental checkpoint redo update lock concurrently controls the redo update position during the replay process.	The redo point update position is concurrently controlled during replay.	Reduce the checkpoint frequency of the primary node and reduce concurrent access during restoration and on-demand replay.
PruneDirtyQueueLock	Lock for clearing the dirty page queue at an incremental checkpoint.	Avoid the conflict between checkpoint update of the dirty page tail position and dirty page queue clearing by the pagewriter main thread during restoration.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
UnlinkRelHashTblLock	Used to protect the concurrent access and storage of the hash table that restores files to be deleted.	All physical files of a table are deleted.	Reduce DDL operations.
UnlinkRelForkHashTblLock	Used to protect the concurrent access and storage of the hash table that restores the fork files to be deleted.	A specified forked physical file is deleted.	Reduce DDL operations.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
LWTRANCHE_ACCOUNT_TABLE	Controls the concurrent read and write of the hash table that stores the account locking records on the standby node.	A password protection policy is provided. When a user uses an incorrect password to log in to the standby node, the hash table is updated to record the number of login failures and the updated lock status, and the information is stored in the system memory. When the number of login failures reaches the maximum, the user is not allowed to log in to the standby node, ensuring the security of the standby node.	Optimization suggestion: Login using an incorrect password is insecure. The possible cause is that the password is changed but the user is not aware of the change, or malicious users continuously attempt to change the database password. The database administrator can periodically check the locking status of all users and analyze the locked accounts to reduce the number of login attempts with incorrect passwords. The following situations are involved: (1) Malicious attacks on the database: Administrators can analyze the login account based on logs, find out the attack source, and take corresponding measures. (2) Password changed without user's awareness: Remind users to promptly update their passwords to ensure normal service operations.
MPFLLOCK	Maximum LSN that has been replayed and flushed to disk.	The maximum LSN that has been replayed and flushed to disk is updated.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
GlobalTempTableControl	Protects the local shared hash table of the global temporary table.	A global temporary table is inserted, deleted, modified, truncated, or accessed.	Reduce concurrent DDL and DML operations on global temporary tables.
NGroupMappingLock	Adds a lock to the concurrent modification of a single bucket that protects the node group hash table.	An exclusive (X) lock is added when a node group in a single bucket is inserted or deleted. This lock is mutually exclusive with other X or S locks, and a wait event is generated.	This wait event occurs only when operations involving node groups are performed.
MatviewSeqnoLock	Used to maintain the **seqno** column in the Mlog table in the fast-refresh materialized view function.	When a fast-refresh materialized view is defined, the incremental data of the base table is maintained through the Mlog table. This lock is applied for when incremental data in the base table is written to the Mlog table. The **seqno** column ensures that the written incremental data has a unique and incremental sequence number (marking the sequence of the incremental data).	When a fast-refresh materialized view is defined, this lock is called when incremental data in the base table is written to the Mlog table. If the IUD performance of a table is important, do not create a fast-refresh materialized view for the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
IOStatLock	Used to concurrently maintain the I/O statistics hash table of resource management.	A lock is requested when the I/O statistics hash table is read or updated.	Reduce I/O operations.
StandbyStmtHistLock	It is mainly used for the FullSQL locating capability of the standby node.	When slow SQL files are flushed to disks, the standby_statement_history function is protected from out-of-bounds read on the same resource.	It is mainly used to locate slow SQL statements. No workaround is required.
WALBufferInitWait	Used to protect the initialization and disk flushing sequence of WAL pages in the shared memory.	The WAL buffer is initialized.	Triggered only during initialization. No optimization needed.
WALInitSegment	Used to protect the initialization sequence of WAL segment files.	Xlogs are pre-initialized.	Triggered only during initialization. No optimization needed.
SegmentHeadPartitionLock	Used to protect the partition lock of metadata in the segment-page header.	When the extent of a segment-page object changes, the segment header needs to be locked to prevent concurrency exceptions.	-
TwoPhaseStatePartLock	Used to protect the status information of two-phase transactions (in each partition).	Used to protect the status information of two-phase transactions (in each partition).	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
RoleIdPartLock	Protects the addition, deletion, and query of process role IDs.	Process role IDs in thread pool mode are added, deleted, and queried.	-
GPRCMappingLock	Used to manage the access and modification operations of the global cache hash table of autonomous transactions.	In the high-concurrency autonomous transaction session scenario, there is a possibility that the lock wait event occurs.	Reduce the number of concurrent sessions in autonomous transactions.
PgwrSyncQueueLock	Used to protect the concurrent access and storage of file queues to be flushed to disks.	Incremental checkpoint.	Reduce checkpoints.
BarrierHashTblLock	Used to protect the concurrent access and storage of barrier tables in the shared memory.	Concurrent access to the barrier_hash_table. During normal distributed DR running, the pre-parsing thread parses barriers and inserts them into the hash table. After replaying barrier logs, the replayer thread removes barriers from the hash table.	-
PageRepairHashTblLock	Used to protect the concurrent access and storage of the hash table that stores repaired pages.	CRC corruption or LSN corruption is detected, and bad blocks are recorded.	Prevent bad blocks.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
FileRepairHashTblLock	Used to protect the concurrent access and storage of the hash table that stores repaired files.	Information is recorded when a file is damaged.	Prevent file damage.
ReplicationOriginLock	Locks are used for creating, deleting, and changing the status of publication and subscription.	Publication and subscription are used.	Reduce the startup of publication and subscription.
PCABufferContentLock	The cross-page transparent compression feature requires a cache page to cache the cmap page to protect the consistency of the cache page. The function is similar to the content lock of shared_buffer.	The consistency of cache pages is protected.	-
WALConsensusWait	Used to protect the transaction commit or log replay which is performed only when the logs are consistent.	Ensure that WALs are consistent on multiple nodes, which may be caused by network latency or node performance differences.	Optimize the network configuration and node performance to reduce the WAL consensus time, or adjust the transaction commit policy to reduce the dependency on consensus mechanism.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
SegMappingLock	Protects concurrent access to segment-page cache hash tables.	The segment-page mode maintains a hash table to store the mapping between logical and physical locations. The hash table is accessed concurrently and needs to be locked.	-
PendingDelLock	Protects concurrent modification of the pending list on threads.	If there are too many operations such as creation, deletion, and modification in a transaction, the pending list may be too long, and it may take a long time to operate the pending list.	Reduce the number of DDL operations in a transaction.
AutonomousSessionLock	Locks are added when autonomous transactions are connected.	Autonomous transactions are established in sequence in high concurrency scenarios.	Reduce the number of concurrent autonomous transactions.
CgroupHashLock	Prevents concurrent initialization locks.	Concurrent initialization.	Reduce service concurrency.
IMCVInfoHashTblLock	Protects the hash table that stores all IMCVDescs in HTAP.	The hash table that stores all IMCVDescs in HTAP is protected.	Use partitioned tables to construct multiple gs_imcv to reduce conflicts.
IMCVDescExpandLock	Protects the concurrent security of a single IMCVDesc.	The concurrent security of a single IMCVDesc is protected.	Reduce the number of concurrent COLVIEW loading operations.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
LogicalStandbyWalLock	Mutual exclusion occurs when the logical standby node pulls and processes remote WALs.	Logical replay is used.	-
DelayDdlInfoLock	Lock held by the standby node for delayed DDL statements.	The primary node executes DDL statements.	Reduce the DDL execution frequency on the primary node.
NvmeLSPosLock	Protects the read and write operations on each of the 4096 NVMe hash partitions.	Statistics on the time of the partition lock range are collected to locate hang and performance problems on the live network.	Adjust the index structure to disperse the distribution of hot pages.
BucketWriterLock	Sync lock of the private buffer during bucket scale-out.	Private buffer contention during bucket scale-out.	-
ClogBucketLock	Protects the bucket Clog control file.	A large number of bucket table services concurrently extend the Clog page.	-
BufFreelistLock	Deprecated.	Deprecated.	Deprecated.
CUSlotListLock	Deprecated.	Deprecated.	Deprecated.
ShmemIndexLock	N/A.	N/A.	N/A.
CLogControlLock	Deprecated.	Deprecated.	Deprecated.
SubtransControlLock	Deprecated.	Deprecated.	Deprecated.
TwoPhaseStateLock	Deprecated.	Deprecated.	Deprecated.
AddinShmemInitLock	Deprecated.	Deprecated.	Deprecated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
AsyncCtlLock	Deprecated.	Deprecated.	Deprecated.
AsyncQueueLock	Deprecated.	Deprecated.	Deprecated.
DataFileIdCacheLock	Deprecated.	Deprecated.	Deprecated.
CStoreColspaceCacheLock	Deprecated.	Deprecated.	Deprecated.
CStoreCUCacheSweepLock	Deprecated.	Deprecated.	Deprecated.
MetaCacheSweepLock	Deprecated.	Deprecated.	Deprecated.
FdwPartitionCacheLock	Deprecated.	Deprecated.	Deprecated.
DfsConnectorCacheLock	Deprecated.	Deprecated.	Deprecated.
dummyServerInfoCacheLock	Deprecated.	Deprecated.	Deprecated.
SearchServerLibLock	Deprecated.	Deprecated.	Deprecated.
DfsUserLoginLock	Deprecated.	Deprecated.	Deprecated.
DfsSpaceCacheLock	Deprecated.	Deprecated.	Deprecated.
LsnXlogChkFileLock	Deprecated.	Deprecated.	Deprecated.
WorkloadStatHashLock	Deprecated.	Deprecated.	Deprecated.
WorkloadIoStatHashLock	Deprecated.	Deprecated.	Deprecated.
WorkloadSessionInfoLock	Deprecated.	Deprecated.	Deprecated.
WorkloadRecordLock	Deprecated.	Deprecated.	Deprecated.
PgfdwLock	Deprecated.	Deprecated.	Deprecated.
RelfilenodeReuseLock	Deprecated.	Deprecated.	Deprecated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
GPCCommitLock	Deprecated.	Deprecated.	Deprecated.
GPCTimelineLock	Deprecated.	Deprecated.	Deprecated.
TsTagsCacheLock	Deprecated.	Deprecated.	Deprecated.
MaxPageFlushLsnFileLock	Deprecated.	Deprecated.	Deprecated.
StreamingEngineConnLock	Deprecated.	Deprecated.	Deprecated.
StreamingEngineExecLock	Deprecated.	Deprecated.	Deprecated.
PartIdCacheLock	Deprecated.	Deprecated.	Deprecated.
PartOidCacheLock	Deprecated.	Deprecated.	Deprecated.
DeleteCompactionLock	Deprecated.	Deprecated.	Deprecated.
DeleteConsumerLock	Deprecated.	Deprecated.	Deprecated.
UHeapStatLock	N/A.	N/A.	N/A.
ConsumerStateLock	Deprecated.	Deprecated.	Deprecated.
SyncPaxosLock	Deprecated.	Deprecated.	Deprecated.
BackgroundWorkerLock	Deprecated.	Deprecated.	Deprecated.
HadrSwitchoverLock	Deprecated.	Deprecated.	Deprecated.
UniqueSqlEvictLock	N/A.	N/A.	N/A.
ExrtoRecycleResidualUndoLock	Deprecated.	Deprecated.	Deprecated.
DblinkFdwRelationLock	Deprecated.	Deprecated.	Deprecated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
SPMGlobalLock	N/A.	N/A.	N/A.
SlotsArrayLock	Deprecated.	Deprecated.	Deprecated.
MtdStatusLock	Deprecated.	Deprecated.	Deprecated.
AltLogicalMapClearLock	N/A.	N/A.	N/A.
AltSnapTrackerXminLock	N/A.	N/A.	N/A.
AltSnapTrackerCsnLock	Deprecated.	Deprecated.	Deprecated.
OndemandQueueStatLock	N/A.	N/A.	N/A.
AscendHashLock	N/A.	N/A.	N/A.
CacheSlotMappingLock	Deprecated.	Deprecated.	Deprecated.
LtxidMappingLock	N/A.	N/A.	N/A.
UspagrpMappingLock	Deprecated.	Deprecated.	Deprecated.
GlobalSeqLock	Deprecated.	Deprecated.	Deprecated.
GlobalWorkloadLock	Deprecated.	Deprecated.	Deprecated.
NormalizedSqlLock	Deprecated.	Deprecated.	Deprecated.
StartBlockMappingLock	Deprecated.	Deprecated.	Deprecated.
DataCacheLock	Deprecated.	Deprecated.	Deprecated.
MetaCacheLock	Deprecated.	Deprecated.	Deprecated.
GeneralExtendedLock	Deprecated.	Deprecated.	Deprecated.
PLdebugger	Deprecated.	Deprecated.	Deprecated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
GlobalCompilationProductHashTblLock	N/A.	N/A.	N/A.
WALFlushWait	Deprecated.	Deprecated.	Deprecated.
AuditIndextblLock	Deprecated.	Deprecated.	Deprecated.
GlobalPlanCacheLock	Deprecated.	Deprecated.	Deprecated.
QueryPlanCacheLock	Deprecated.	Deprecated.	Deprecated.
SPMMappingLock	N/A.	N/A.	N/A.
SPMGlobalInfoLock	Deprecated.	Deprecated.	Deprecated.
AntiCacheLock	N/A.	N/A.	N/A.
SQLFeedbackLock	N/A.	N/A.	N/A.
IMCUDataCacheLock	Deprecated.	Deprecated.	Deprecated.
IMCUMetaCacheLock	Deprecated.	Deprecated.	Deprecated.
IMCVDescLatch	Deprecated.	Deprecated.	Deprecated.
WalRemoteWaitLock	N/A.	N/A.	N/A.
AbnormalSQLMappingLock	N/A.	N/A.	N/A.

Table 7-472 describes the corresponding wait events when **wait_status** is **wait io**.

Table 7-472 I/O wait events

wait_event	Description	Cause	Suggestions
buffile_read	Waits for reading data from the temporary file.	The query statement generates temporary files.	Increase the value of **work_mem**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
buffile_write	Waits for writing temporary files.	The query statement generates temporary files.	Increase the value of **work_mem**.
BufHashTableSearch	Searches the hash table in the shared buffer (page eviction may be triggered).	A specified buffer is searched from the buffer.	Increase the size of the shared buffer to increase the buffer hit ratio.
buffer_strategy_get	Time required for finding the buffer in the buffer replacement policy.	The available buffer is obtained.	Reduce the checkpoint time, pagewriter_sleep time, and bgwriter_delay time to accelerate the cache scanning frequency and reduce concurrency.
ControlFileRead	Reads the **pg_control** file, mainly during database startup, checkpoint execution, and primary/standby verification.	During database startup, checkpoint execution, and primary and standby verification, data is loaded from the control file.	Reduce the operations of starting and stopping the database, checkpoint, and switchover.
ControlFileSync	Flushes the **pg_control** file to a disk, mainly during database initialization.	When the database is started, data in the **pg_control** file is flushed to the disk.	Reduce the number of database startup and stopping operations.
ControlFileSyncUpdate	Flushes the **pg_control** file to a disk, mainly during database startup, checkpoint execution, and primary/standby verification.	During database startup, checkpoint execution, and primary and standby verification, data in the **pg_control** file is flushed to disks.	Reduce the operations of starting and stopping the database, checkpoint, and switchover.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
ControlFileWrite	Writes the **pg_control** file, mainly during database initialization.	When the database is started, the control file is initialized and written to the disk.	Reduce the number of database startup and stopping operations.
ControlFileWriteUpdate	Updates the **pg_control** file, mainly during database startup, checkpoint execution, and primary/standby verification.	During database startup, checkpointer execution, and primary and standby verification, the control file change information needs to be written into the **pg_control** file.	Reduce the operations of starting and stopping the database, checkpoint, and switchover.
CopyFileRead	Reads a file during file copying.	The file I/O performance bottleneck occurs or the source file is locked by other operations.	Optimize the I/O performance or ensure that the source file is not locked by other operations for a long time to reduce lock contention.
CopyFileWrite	Writes a file during file copying.	The file I/O performance bottleneck occurs or the target file is locked by other operations.	Optimize the I/O performance or ensure that the target file is not locked by other operations for a long time to reduce lock contention.
DataFileExtend	Writes content to a file during file extending.	The file is extended.	Reduce the service volume and check the I/O performance.
DataFileImmediateSync	Flushes a table data file to a disk immediately.	The checkpoint logic has been triggered.	Adjust the checkpoint interval and check the I/O performance.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
DataFilePrefetch	Reads a table data file asynchronously.	The table data file information is missing in the memory and needs to be loaded from the disk.	Adjust the value of **shared_buffers** and check the I/O performance.
DataFileRead	Reads a table data file synchronously.	The table data file information is missing in the memory and needs to be loaded from the disk.	Adjust the value of **shared_buffers** and check the I/O performance.
DataFileSync	Flushes table data file modifications to a disk.	The checkpoint logic has been triggered.	Adjust the checkpoint interval and check the I/O performance.
DataFileTruncate	Truncates a table data file.	Truncate operation has been triggered.	Reduce concurrent operations on the truncate table.
DataFileWrite	Writes a table data file.	The checkpoint logic has been triggered.	Adjust the checkpoint interval and check the I/O performance.
SegFileExtend	Extends segment-page files.	When the database requires more storage space, the segment file is extended. Due to the performance limitation of the file system, the segment files will wait.	Optimize the I/O performance of the file system or allocate sufficient storage space in advance to reduce expansion operations.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
SegReadDisk	Reads segment-page files.	When accessing data, the database needs to read segment files from disks. Due to I/O performance restrictions, the database waits for the segment files to be read.	Optimize the I/O performance of the storage device or increase the cache to reduce read operations.
SegWriteDisk	Writes segment-page files.	When updating data, the database needs to write data to the segment file. Due to the I/O performance limitation, the database waits for data to be written to the segment file.	Optimize the I/O performance of the storage device or increase the parallelism of write operations.
SegSync	Flushes segment-page files.	After data is modified, you need to ensure data persistence. Therefore, you need to wait for file synchronization.	Optimize the I/O performance of the storage device, or adjust the synchronization frequency to reduce the waiting time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
SegFileShrink	Collapses segment-page files.	When the database releases unnecessary storage space, segment files are shrunk. Due to the performance limitation of the file system, the segment files will wait.	Optimize the I/O performance of the file system or reduce the frequency of segment file shrinking.
LsnInfoPageDiskRead	I/O delay of reading the LSN info page when read on standby in ultimate RTO mode is enabled.	Read on standby in ultimate RTO mode is enabled.	Enhance the I/O capability.
BlkInfoMetaDiskRead	I/O delay of reading the block info page when read on standby in ultimate RTO mode is enabled.	Read on standby in ultimate RTO mode is enabled.	Enhance the I/O capability.
BasePageDiskRead	I/O delay of reading the base page when read on standby in ultimate RTO mode is enabled.	Read on standby in ultimate RTO mode is enabled.	Enhance the I/O capability.
LockFileAddToDataDirRead	Reads the **postmaster.pid** file.	The **postmaster.pid** file has been read concurrently.	Reduce service concurrency.
LockFileAddToDataDirSync	Flushes the **postmaster.pid** file to a disk.	The **postmaster.pid** file has been flushed concurrently.	Reduce service concurrency.
LockFileAddToDataDirWrite	Writes PID information into the **postmaster.pid** file.	The **postmaster.pid** file has been written concurrently.	Reduce service concurrency.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
LockFileCreateRead	Reads the LockFile file **%s.lock**.	The LockFile file has been read concurrently.	Reduce service concurrency.
LockFileCreateSync	Flushes the LockFile file **%s.lock** to a disk.	The LockFile file has been flushed concurrently.	Reduce service concurrency.
LockFileCreateWRITE	Writes PID information into the LockFile file **%s.lock**.	The LockFile file has been written concurrently.	Reduce service concurrency.
ReplicationSlotRead	Reads a streaming replication slot file This operation is performed during a restart.	Slot persistence information has been read.	Used only when the process is started and the replication slot information is loaded. No optimization is required.
ReplicationSlotRestoreSync	Flushes a streaming replication slot file to a disk This operation is performed during a restart.	Replication slot persistence information has been synchronized from disks.	Used only when the process is started and the replication slot information is loaded. No optimization is required.
ReplicationSlotSync	Flushes a temporary streaming replication slot file to a disk during checkpoint execution.	Replication slot information has been flushed to a disk.	Used only when the replication slot status is created or refreshed. No optimization is required.
ReplicationSlotWrite	Writes a temporary streaming replication slot file during checkpoint execution.	Replication slot information has been written to a temporary file.	Used only when the replication slot status is created or refreshed. No optimization is required.
SLRUFlushSync	Flushes the **pg_clog**, **pg_subtrans**, and **pg_multixact** files to a disk, mainly during checkpoint execution and database shutdown.	The checkpoint logic has been triggered.	Decrease the checkpoint interval and check the I/O performance.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
SLRURead	Reads the **pg_clog**, **pg_subtrans**, and **pg_multixact** files.	The Clog and CSNlog information is missing in the memory and needs to be loaded from the disk.	Increase the **shared_buffers** size and check the I/O performance.
SLRUSync	Writes dirty pages into the **pg_clog**, **pg_subtrans**, and **pg_multixact** files, and flushes the files to a disk, mainly during checkpoint execution and database shutdown.	The checkpoint logic has been triggered.	Decrease the checkpoint interval and check the I/O performance.
SLRUWrite	Writes the **pg_clog**, **pg_subtrans**, and **pg_multixact** files.	The checkpoint logic has been triggered.	Decrease the checkpoint interval and check the I/O performance.
Twophase FileRead	Reads the **pg_twophase** file, mainly during two-phase transaction commit and restoration.	The **pg_twophase** information is missing in the memory and needs to be loaded from the disk.	Increase the **shared_buffers** size and check the I/O performance.
Twophase FileSync	Flushes the **pg_twophase** file to a disk, mainly during two-phase transaction commit and restoration.	During a two-phase transaction, dirty pages are flushed to disks.	Decrease the checkpoint interval and check the I/O performance.
Twophase FileWrite	Writes the **pg_twophase** file, mainly during two-phase transaction commit and restoration.	Disk files have been rebuilt during a two-phase transaction.	Decrease the checkpoint interval and check the I/O performance.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
UndoFileExtend	Extends an undo file.	DML generates undo records. When the space is insufficient, the undo file needs to be expanded to store new undo records.	Reduce DML operations on the Ustore table.
UndoFilePrefetch	Asynchronously reads undo files.	When the system supports posix_fadvise () calls, the database pre-reads the scan to improve performance.	Reduce DML operations on the Ustore table.
UndoFileRead	Reads an undo file.	When a service needs to access a historical tuple version or a transaction needs to be rolled back, the undo record is read.	Reduce DML operations on the Ustore table.
UndoFileWrite	Writes an undo file.	DML generates undo records. The undo records are written to the buffer and then flushed to disks in the pagewriter thread. This wait event is generated in the flushing process.	Reduce DML operations on the Ustore table.
UndoFileSync	Synchronously flushes undo files to disk.	Undo records have been synchronously flushed to disk.	Reduce DML operations on the Ustore table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
UndoFileUnlink	Deletes an undo file.	Undo records are generated when DML operations are performed on each Ustore table. Unnecessary undo records must be deleted in a timely manner to prevent the undo space from increasing.	Reduce DML operations on the Ustore table.
UndoSegFileExtend	Extends undo segment-page files.	DML generates undo records. When the space is insufficient, the undo segment-page file needs to be expanded to store new undo records.	Reduce DML operations on the Ustore table.
UndoSegReadDisk	Reads undo segment-page files.	When a service needs to access a historical tuple version or a transaction needs to be rolled back, the undo record is read.	Reduce DML operations on the Ustore table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
UndoSegWriteDisk	Writes undo segment-page files.	DML generates undo records. The undo records are written to the buffer and then flushed to disks in the pagewriter thread. This wait event is generated in the flushing process.	Reduce DML operations on the Ustore table.
UndoSegSync	Synchronously flushes undo segment-page files to disk.	Undo records have been synchronously flushed to disk.	Reduce DML operations on the Ustore table.
UndoMetaSync	Synchronously flushes undo metadata files to disk.	The undo metadata is flushed to disk with each checkpoint. Ensure that the undo metadata is flushed in real time after the checkpoint is updated.	Increase the value of **incremental_checkpoint_timeout**.
WALBootstrapSync	Flushes initialized WAL files to a disk during database initialization.	The database has been initialized.	Triggered only during initialization. No optimization needed.
WALBootstrapWrite	Writes initialized WAL files during database initialization.	The database has been initialized.	Triggered only during initialization. No optimization needed.
WALCopyRead	Records read operations generated when existing WAL files are read for replication after archiving and restoration.	Archive restoration has been triggered.	Related to the I/O performance, which can be optimized on the hard disk.
WALCopySync	Flushes replicated WAL files to a disk after archive restoration.	Archive restoration has been triggered.	Related to the I/O performance, which can be optimized on the hard disk.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
WALCopyWrite	Records write operations generated when existing WAL files are read for replication after archiving and restoration.	Archive restoration has been triggered.	Related to the I/O performance, which can be optimized on the hard disk.
WALInitSync	Flushes a newly initialized WAL file to a disk during log recycling or writing.	A log file has been created.	Related to the I/O performance, which can be optimized on the hard disk.
WALInitWrite	Initializes a newly created WAL file to **0** during log recycling or writing.	A log file has been created.	Related to the I/O performance, which can be optimized on the hard disk.
WALRead	Reads data from Xlogs during redo operations on two-phase files.	Log transfer or logical decoding has been triggered.	Related to the I/O performance, which can be optimized on the hard disk.
ShareStorageWalRead	Reads log files from the shared disk.	The WAL read speed of the shared storage is slow. As a result, the read operation is time-consuming.	Optimize the network and storage performance, or increase the read parallelism of WAL files.
ShareStorageWalWrite	Writes log files to the shared disk.	The WAL write speed of the shared storage is slow. As a result, the write operation is time-consuming.	Optimize the storage performance or adjust the WAL generation configuration to reduce the waiting time of write operations.
ShareStorageCtlInfoRead	Reads control information from the shared disk.	The speed of reading the control information of the shared storage is slow. As a result, the read operation is time-consuming.	Optimize the storage performance or reduce the frequency of reading control information.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
ShareStorageCtlInfoWrite	Writes control information to the shared disk.	The write speed of the shared storage control information is slow. As a result, the write operation is time-consuming.	Optimize the storage performance or reduce the frequency of writing control information.
WALSyncMethodAssign	Flushes all open WAL files to a disk.	The **wal_sync_method** parameter has been modified.	Related to the I/O performance, which can be optimized on the hard disk.
WALWrite	Writes a WAL file.	Xlog files are written.	Related to the I/O performance, which can be optimized on the hard disk.
WALBufferAccess	Accesses WAL buffer. (To ensure performance, only the number of access times is counted in the kernel code, and the access duration is not counted.)	WAL records are inserted.	The time consumption is not counted, and no optimization is required.
WALBufferFull	Writes WAL files when the WAL buffer is full.	The WAL buffer is full and waits for flushing.	Increase the wal_buffers size.
DoubleWriteFileRead	Waits for reading the doublewrite file.	During database startup, pages in the doublewrite file are scanned and half-write pages are repaired.	Reduce the number of database startup and stop operations to avoid half-write.
DoubleWriteFileWrite	Waits for writing the doublewrite file.	The checkpoint logic has been triggered.	Reduce checkpoints.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
DWSingleFlushGetPos	Searches for an available location for a single-page doublewrite file.	When a full checkpoint is performed at an incremental checkpoint, the single-page doublewrite file is truncated.	Check the manual checkpoint operation.
DWSingleFlushWrite	Flushes a single-page doublewrite file to a disk.	A single-page doublewrite file is written during dirty page eviction.	Reduce page eviction and increase the value of **shared_buffers**.
CkptWaitPageWriterFlush	Waits for page refreshing during full checkpoint execution.	The dirty page refresh speed is slow. As a result, the checkpoint operation is blocked.	Optimize the I/O performance or increase the number of writers to improve the write efficiency.
CkptWaitPageWriterSync	Waits for the modified file to be synchronized to the disk before checkpointing.	The dirty page synchronization speed is slow. As a result, the checkpoint operation is blocked.	Optimize the I/O performance or adjust the checkpoint configuration to reduce the synchronization waiting time.
CkptWaitCommitTransactionFinish	Waits until all transactions are committed before checkpointing.	Checkpoints need to wait for all active transactions to be committed to ensure data consistency.	Optimize the transaction processing process, reduce long transactions, or adjust the frequency of triggering checkpoints.
PredoProcessPending	Waits for the replay of other records to complete during parallel log replay.	Logs to be synchronized, such as DDL-related logs, are displayed.	Reduce DDL operations.
PredoApply	Waits for the replay of other threads to the LSN of the current thread during parallel log replay.	Logs to be synchronized, such as DDL-related logs, are displayed.	Reduce DDL operations.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
DisableConnectFileRead	Reads the HA lock fragment logic file.	HA build is successful, and the primary and standby configuration files are synchronized.	Related to the I/O performance, which can be optimized on the hard disk.
DisableConnectFileSync	Forcibly flushes the HA lock fragment logic file to disks.	HA build is successful, and the primary and standby configuration files are synchronized.	Related to the I/O performance, which can be optimized on the hard disk.
DisableConnectFileWrite	Writes the HA lock fragment logic file.	HA build is successful, and the primary and standby configuration files are synchronized.	Related to the I/O performance, which can be optimized on the hard disk.
MPFL_INIT	Initiates **max_page_flush_lsn**.	The control file has been initialized.	Related to the I/O performance, which can be optimized on the hard disk.
MPFL_READ	Reads **max_page_flush_lsn**.	The control file has been read.	Related to the I/O performance, which can be optimized on the hard disk.
MPFL_WRITE	Writes **max_page_flush_lsn**.	The control file has been written.	Related to the I/O performance, which can be optimized on the hard disk.
LOGCTRL_SLEEP	Waits for the standby node to catch up with replay.	Enable flow control.	Disable flow control.
WALRcvBufFull	The buffer is full when walreceiver of the standby node receives Xlogs and stores them in the buffer.	The capacity of the WAL receive buffer is insufficient. As a result, the received WALs cannot be processed in a timely manner.	Related to the I/O performance, which can be optimized on the hard disk.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
WALFileSync	Flushes the WAL file to a disk.	After the WAL write operation is performed, you need to ensure that the data is persistent. Therefore, file synchronization is waited.	Related to the I/O performance, which can be optimized on the hard disk.
VlogInitSync	Waits for initializing the Vlog file.	A new Vlog file is created after the page verification persistence is enabled.	Related to the I/O performance, which can be optimized on the hard disk.
VlogWrite	Waits for writing Vlog records to a file.	Vlog records are written to a file after the page verification persistence is enabled.	Related to the I/O performance, which can be optimized on the hard disk.
OBSWrite	Deprecated.	Deprecated.	Deprecated.
invalid io event	Unexpected read/write event.	An unexpected read/write event occurs.	Contact Huawei technical support.

Table 7-473 describes the corresponding wait events when **wait_status** is **acquire lock**.

Table 7-473 List of wait events corresponding to transaction locks

wait_event	Description	Cause	Suggestions
relation	Adds a lock to a table.	A conflict occurs when the service statement obtains the table lock.	Do not perform DML and DDL operations on the same table concurrently.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
extend	Adds a lock to a table being scaled out.	A conflict occurs when the same table is frequently extended.	Do not concurrently insert too much data into a single table
partition	Adds a lock to a partitioned table.	A conflict occurs when the service statement obtains the partitioned table lock.	Do not perform DML and DDL operations on the same partitioned table concurrently.
partition_seq	Adds a lock to a partition of a partitioned table.	A conflict occurs when the service statement obtains the partitioned table lock of the partitioned table.	Do not perform DML and DDL operations concurrently on partitions of the same partitioned table.
page	Adds a lock to a table page.	A conflict occurs when service statements are used to add, delete, or modify vector metadata.	Reduce concurrent DML operations on vector columns in the vector table.
tuple	Adds a lock to a tuple on a page.	A tuple is locked when DML is executed.	Reduce concurrent DML operations.
transactionid	Adds a lock to a transaction ID.	A transaction has been concurrently accessed.	Reduce concurrent access to the same transaction.
virtualxid	Adds a lock to a virtual transaction ID.	A virtual transaction has been concurrently accessed.	Reduce concurrent access to the same virtual transaction.
object	Adds a lock to an object.	The internal object lock of the database conflicts.	Analyze the specific process code.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
advisory	Adds an advisory lock.	A lock conflict occurs when the advisory lock function is called. For example, if session 1 calls the pg_advisory_lock_int8 function and does not release the lock for a long time, session 2 enters the waiting state when calling the same parameter of the pg_advisory_lock_int8 function.	Optimize the service process to prevent lock conflicts.
filenode	Adds a lock to a file name and controls the concurrent sequence of file-level operations.	When multiple sessions concurrently access the same file, the system needs to lock the file to control the concurrency sequence and prevent data conflicts.	Optimize the access logic to reduce concurrent operations on the same file.
subtransactionid	Adds a lock to a sub-transaction ID.	A sub-transaction has been concurrently accessed.	Reduce concurrent access to the same sub-transaction.
tuple_uid	Adds a lock to the UID hidden column in the tuple header.	DML operations have been performed on the Ustore table.	Reduce concurrent DML operations on the Ustore table.
spectoken	Protects the upsert process.	The upsert operation is concurrently executed in a transaction.	Prevent transactions from executing upsert concurrently or optimize the upsert performance.
bucket	Adds a bucket lock.	A conflict occurs when the service statement obtains the bucket lock.	-

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

wait_event	Description	Cause	Suggestions
vector_ccbuild	Locks the tuple of an index.	When a vector index is created, the tuple of the index is locked.	Reduce concurrent creation operations.
cstore_freespace	Deprecated.	Deprecated.	Deprecated.
userlock	Deprecated.	Deprecated.	Deprecated.

7.3.13.188 PG_TIMEZONE_ABBREVS

PG_TIMEZONE_ABBREVS displays information about all available time zones. For details about the columns, see **Table 7-474**.

Table 7-474 PG_TIMEZONE_ABBREVS columns

Name	Type	Description
abbrev	text	Time zone name abbreviation.
utc_offset	interval	Offset from UTC.
is_dst	boolean	Specifies whether the DST is being used. If the DST is being used, the value is **t**; otherwise, the value is **f**.

7.3.13.189 PG_TIMEZONE_NAMES

PG_TIMEZONE_NAMES displays all time zone names that can be recognized by SET TIMEZONE, along with their abbreviations, UTC offsets, and daylight saving time (DST) statuses. For details about the columns, see **Table 7-475**.

Table 7-475 PG_TIMEZONE_NAMES columns

Name	Type	Description
name	text	Name of the time zone.
abbrev	text	Abbreviation of the ime zone name.
utc_offset	interval	Offset from UTC.
is_dst	boolean	Specifies whether the DST is being used. If the DST is being used, the value is **t**; otherwise, the value is **f**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.190 PG_TOTAL_MEMORY_DETAIL

PG_TOTAL_MEMORY_DETAIL displays memory usage of a node in the database. If the GUC parameter **enable_memory_limit** is set to **off**, this view is unavailable. For details about the columns, see **Table 7-476**.

Table 7-476 PG_TOTAL_MEMORY_DETAIL columns

Name	Type	Description
nodename	text	Node name
memorytype	text	Memory name
memorymbytes	integer	Size of the used memory, in MB

7.3.13.191 PG_TOTAL_USER_RESOURCE_INFO

PG_TOTAL_USER_RESOURCE_INFO displays resource usage of all users. Only administrators can query this view. This view is valid only when the GUC parameter **use_workload_manager** is set to **on**. I/O monitoring items are valid only when **enable_logical_io_statistics** is set to **on**. For details about the columns, see **Table 7-477**.

Table 7-477 PG_TOTAL_USER_RESOURCE_INFO columns

Name	Type	Description
username	name	Username.
used_memory	integer	This column is deprecated in the current version. The query result is meaningless.
total_memory	integer	This column is deprecated in the current version. The query result is meaningless.
used_cpu	double precision	This column is deprecated in the current version. The query result is meaningless.
total_cpu	integer	This column is deprecated in the current version. The query result is meaningless.
used_space	bigint	Used permanent table storage space, in KB.
total_space	bigint	Available permanent table storage space, in KB (**-1** if the storage space is not limited).
used_temp_space	bigint	Used temporary space, in KB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
total_temp_space	bigint	Total available temporary space, in KB (**-1** if the temporary space is not limited).
used_spill_space	bigint	Size of the used operator flushing space, in KB.
total_spill_space	bigint	Total size of the available operator flushing space, in KB (**-1** if the space is not limited).
read_kbytes	bigint	CN: total bytes read by the user's complex jobs on all DNs in the last 5 seconds, in KB DN: total bytes read by the user's complex jobs from the instance startup time to the current time, in KB
write_kbytes	bigint	CN: total bytes written by the user's complex jobs on all DNs in the last 5 seconds, in KB DN: total bytes written by the user's complex jobs from the instance startup time to the current time, in KB
read_counts	bigint	CN: total number of read times of the user's complex jobs on all DNs in the last 5 seconds DN: total number of read times of the user's complex jobs from the instance startup time to the current time
write_counts	bigint	CN: total number of write times of the user's complex jobs on all DNs in the last 5 seconds DN: total number of write times of the user's complex jobs from the instance startup time to the current time
read_speed	double precision	CN: average read rate of the user's complex jobs on a single DN in the last 5 seconds, in KB/s DN: average read rate of the user's complex jobs on the DN in the last 5 seconds, in KB/s

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
write_speed	double precision	CN: average write rate of the user's complex jobs on a single DN in the last 5 seconds, in KB/s DN: average write rate of the user's complex jobs on a single DN in the last 5 seconds, in KB/s

NOTE

- There may be a ±5% difference between the used disk space calculated in the PG_TOTAL_USER_RESOURCE_INFO view and the actual used disk space. You can run the **SELECT gs_wlm_readjust_user_space(0)** statement to correct the difference. The database also periodically (every seven days) corrects the space used by users. If data is inserted into the table during the execution of this statement, the statistics may be inaccurate.
- After DROP DATABASE is performed, the user space information collected in the PG_TOTAL_USER_RESOURCE_INFO view is not cleared immediately. When the CPU usage is high, it takes one to two hours to clear the information and restore the accuracy of the statistics.

7.3.13.192 PG_TOTAL_USER_RESOURCE_INFO_OID

PG_TOTAL_USER_RESOURCE_INFO_OID displays resource usage of all users. Only administrators can query this view. This view is valid only when the GUC parameter **use_workload_manager** is set to **on**. For details about the columns, see **Table 7-478**.

Table 7-478 PG_TOTAL_USER_RESOURCE_INFO_OID columns

Name	Type	Description
userid	oid	User ID.
used_memory	integer	This column is deprecated in the current version. The query result is meaningless.
total_memory	integer	This column is deprecated in the current version. The query result is meaningless.
used_cpu	double precision	This column is deprecated in the current version. The query result is meaningless.
total_cpu	integer	This column is deprecated in the current version. The query result is meaningless.
used_space	bigint	Used storage space, in KB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
total_space	bigint	Available storage space (unit: KB). The value **-1** indicates that the space is not limited.
used_temp_space	bigint	Used temporary storage space, in KB.
total_temp_space	bigint	Total available temporary space, in KB (**-1** if the temporary space is not limited).
used_spill_space	bigint	Used disk space for spilling, in KB.
total_spill_space	bigint	Total available disk space for spilling, in KB. The value **-1** indicates that the space is not limited.
read_kbytes	bigint	Amount of data read from the disk, in KB.
write_kbytes	bigint	Amount of data written to the disk, in KB.
read_counts	bigint	Number of disk read times.
write_counts	bigint	Number of disk write times.
read_speed	double precision	Disk read rate, in B/ms.
write_speed	double precision	Disk write rate, in B/ms.

7.3.13.193 PG_VARIABLE_INFO

PG_VARIABLE_INFO records information about transaction IDs and OIDs of the current node in a cluster. For details about the columns, see **Table 7-479**.

Table 7-479 PG_VARIABLE_INFO columns

Name	Type	Description
node_name	text	Node name.
next_oid	oid	OID generated next time for the node.
next_xid	xid	Transaction ID generated next time for the node.
oldest_xid	xid	Oldest transaction ID on the node.
xid_vac_limit	xid	Critical point that triggers forcible autovacuum.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
oldest_xid_db	oid	OID of the database that has the minimum datafrozenxid on the node.
last_extend_csn_logpage	xid	Number of the last extended CSNlog page.
start_extend_csn_logpage	xid	Number of the page from which csnlog extending starts.
next_commit_seqno	xid	CSN generated next time for the node.
latest_completed_xid	xid	Latest transaction ID on the node after the transaction commit or rollback.
startup_max_xid	xid	Last transaction ID before the node is powered off.

7.3.13.194 PG_VIEWS

PG_VIEWS displays useful information about each view in the database. For details about the columns, see **Table 7-480**.

Table 7-480 PG_VIEWS columns

Name	Type	Reference	Description
schemaname	name	**nspname** in **PG_NAMESPACE**	Schema name of a view.
viewname	name	**relname** in **PG_CLASS**	View name.
viewowner	name	**PG_AUTHID**.rolname	Owner of the view
definition	text	-	Definition of the view.

7.3.13.195 PGXC_COPY_ERROR_LOG

The PGXC_COPY_ERROR_LOG system view is used to capture error rows related to the COPY FROM procedure for future reference.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-481 PGXC_COPY_ERROR_LOG columns

Name	Type	Description
relname	text	Table name, in the form of *Schema name*.*Table name*.
begintime	timestamp with time zone	Time when a data format error was reported.
filename	text	Name of the source data file where a data format error occurs.
lineno	bigint	Number of the row where a data format error occurs in a source data file.
rawrecord	text	Raw record of a data format error in the source data file.
detail	text	Error details.
custom_id	text	Value of **copy_custom_id** in the COPY syntax, which indicates the information about the imported custom identifier.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- When the pg_catalog.pgxc_copy_error_log system view is queried, data in the pg_catalog.gs_copy_error_log table on all CNs and DNs is displayed. It is equivalent to running the following statement:
  ```
  SELECT * FROM pg_catalog.pgxc_copy_error_func(false, false);
  ```
- By default, the view does not perform the UNION ALL operation on the pg_catalog.gs_copy_error_old table. To be compatible with the queried data in the pg_catalog.gs_copy_error_old table, run the following statement:
  ```
  SELECT * FROM pg_catalog.pgxc_copy_error_func(true, false);
  ```
- In this version and later versions, pgxc_copy_error_log is changed from a system catalog to a system view. Therefore, the TRUNCATE operation cannot be performed on this view, otherwise, an error message is displayed indicating that the operation cannot be performed on the view. You can clear the copy-related error data as follows:
  - In non-separation-of-duties scenarios, the SYSADMIN needs to run the following statement to clear the data:
    ```
    TRUNCATE TABLE pg_catalog.gs_copy_error_log;
    ```
  - In separation-of-duties scenarios, clear the pgxc_copy_error_log table in the schema with the same name as a common user.
- In separation-of-duties scenarios, error logs are not recorded in pg_catalog.pgxc_copy_error_log or pg_catalog.gs_copy_error_log. If error information needs to be recorded for the **load_user** user, execute the following statement to create the error log in the schema with the same name as the user:
  ```
  CREATE TABLE load_user.pgxc_copy_error_log (relname varchar, begintime timestamptz, filename varchar, lineno int8, rawrecord text, detail text);
  CREATE INDEX copy_error_log_relname_idx ON load_user.pgxc_copy_error_log(relname);
  ```

  (Optional) Add the **custom_id** column to the table.

  ```
  ALTER TABLE load_user.pgxc_copy_error_log ADD COLUMN custom_id varchar(64);
  ```

7.3.13.196 PGXC_GET_STAT_ALL_TABLES

PGXC_GET_STAT_ALL_TABLES displays information about insertion, update, and deletion operations on tables and the dirty page rate of tables. Before running **VACUUM FULL** to a system catalog with a high dirty page rate, ensure that no user is performing operations on it. This view is a new feature of GaussDB, and will not collect statistics on insertion, update, and deletion operations in earlier versions. You are advised to run **VACUUM FULL** to tables (excluding system catalogs) whose dirty page rate exceeds 30% or run it based on service scenarios. Only users with the SYSADMIN or MONADMIN permissions can view the information. For details about the columns, see **Table 7-482**.

Table 7-482 PGXC_GET_STAT_ALL_TABLES columns

Name	Type	Description
relid	oid	Table OID
relname	name	Table name.
schemaname	name	Name of the schema that contains a table.
n_tup_ins	numeric	Number of inserted tuples.
n_tup_upd	numeric	Number of updated tuples.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
n_tup_del	numeric	Number of deleted tuples.
n_live_tup	numeric	Number of live tuples.
n_dead_tup	numeric	Number of dead tuples. It indicates the number of inactive row pointers in Ustore tables.
dirty_page_rate	numeric(5,2)	Dirty page rate (%) of a table.

7.3.13.197 PGXC_GET_TABLE_SKEWNESS

PGXC_GET_TABLE_SKEWNESS displays the data skew on tables in the current database. Only users with the SYSADMIN permission can access. For details about the columns, see **Table 7-483**.

Table 7-483 PGXC_GET_TABLE_SKEWNESS columns

Name	Type	Description
schemaname	name	Name of the schema where the table is located.
tablename	name	Table name.
totalsize	numeric	Total size of the table, in bytes.
avgsize	numeric(1000,0)	Average table size (total table size divided by the number of DNs), which is the ideal size of tables distributed on each DN.
maxratio	numeric(4,3)	Ratio of the maximum table size on a single DN to the total table size.
minratio	numeric(4,3)	Ratio of the minimum table size on a single DN to the total table size.
skewsize	bigint	Table skew rate (the maximum table size on a single DN minus the minimum table size on a single DN).
skewratio	numeric(4,3)	Table skew rate (skew size divided by total table size).
skewstddev	numeric(1000,0)	Standard deviation of table distribution. (For two tables of the same size, a larger deviation indicates a more severe skew.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.198 PGXC_NODE_ENV

PGXC_NODE_ENV displays environmental variables of all nodes in a cluster. Only users with the SYSADMIN or MONADMIN permission can query this view. For details about the columns, see **Table 7-484**.

Table 7-484 PGXC_NODE_ENV columns

Name	Type	Description
node_name1	text	Name of a node in a cluster.
host1	text	Host names of the nodes in the cluster.
process1	integer	Process IDs of the nodes in the cluster.
port1	integer	Port numbers of the nodes in the cluster.
installpath1	text	Installation directory of the nodes in the cluster.
datapath1	text	Data directory of the nodes in the cluster.
log_directory1	text	Log directory of the nodes in the cluster.

7.3.13.199 PGXC_OS_THREADS

PGXC_OS_THREADS displays thread status information under all normal nodes in the current cluster. Only users with the SYSADMIN or MONADMIN permission can query this view. For details about the columns, see **Table 7-485**.

Table 7-485 PGXC_OS_THREADS columns

Name	Type	Description
node_name	text	Names of normal nodes in a cluster.
pid	bigint	IDs of running threads among normal node processes in the current cluster.
lwpid	integer	Lightweight thread ID corresponding to the PID.
thread_name	text	Thread name corresponding to the PID.
creation_time	timestamp with time zone	Thread creation time corresponding to the PID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
dbid	oid	ID of the database to which a thread belongs.

7.3.13.200 PGXC_PREPARED_XACTS

PGXC_PREPARED_XACTS displays two-phase transactions in the prepared phase. Only users with the system admin or monitor admin permission can view the information. For details about the columns, see **Table 7-486**.

Table 7-486 PGXC_PREPARED_XACTS columns

Name	Type	Description
pgxc_prepared_xact	text	Displays the two-phase transaction in the prepared phase.

7.3.13.201 PGXC_RUNNING_XACTS

PGXC_RUNNING_XACTS displays information about running transactions on each node in the cluster. The content is the same as that displayed by PG_RUNNING_XACTS. Only users with the system admin or monitor admin permission can view the information. For details about the columns, see **Table 7-487**.

Table 7-487 PGXC_RUNNING_XACTS columns

Name	Type	Description
handle	integer	Handle corresponding to the transaction in GTM.
gxid	xid	Transaction ID.
state	tinyint	Transaction status. (**3**: prepared; **0**: starting.)
node	text	Node name.
xmin	xid	Minimum transaction ID on the node.
vacuum	boolean	Specifies whether the current transaction is lazy vacuum. • **t** (true): yes. • **f** (false): no.
timeline	bigint	Number of database restarts.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
prepare_xid	xid	Transaction ID in the **prepared** state. If the status is not **prepared**, the value is **0**.
pid	bigint	Thread ID corresponding to the transaction.
next_xid	xid	Transaction ID sent from a CN to a DN.
dbid	oid	ID of the database corresponding to the transaction.

7.3.13.202 PGXC_STAT_ACTIVITY

PGXC_STAT_ACTIVITY displays information about the current user's queries on all CNs in the current cluster. Only users with the MONADMIN or SYSADMIN permission can view the view. Three columns are added to version 505.0.0. During the gray upgrade observation period, an error is reported. You need to use PG_STAT_ACTIVITY to directly connect to the target node for query. For details about the columns, see **Table 7-488**.

Table 7-488 PGXC_STAT_ACTIVITY columns

Name	Type	Description
coorname	text	Name of a CN in the current cluster.
datid	oid	OID of the database that the user session connects to in the background.
datname	text	Name of the database that the user session connects to in the background.
pid	bigint	Background thread ID.
sessionid	bigint	Session ID.
usesysid	oid	OID of the user logged in to the background.
usename	text	Name of the user logged in to the background.
application_name	text	Name of the application connected to the background.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
client_addr	inet	IP address of the client connected to the background. If this column is **null**, either the client is connected via a Unix socket on the server machine or this is an internal process such as autovacuum.
client_hostname	text	Host name of the connected client, as reported by a reverse DNS lookup of **client_addr**. This column will be non-null for IP connections only when **log_hostname** is enabled.
client_port	integer	TCP port number that the client uses for communication with the background (**-1** if a Unix socket is used).
backend_start	timestamp with time zone	Time when this process was started, that is, when the client connected to the server.
xact_start	timestamp with time zone	Time when the current transaction was started (**null** if no transaction is active). If the current query is the first of its transaction, this column is equal to the **query_start** column.
query_start	timestamp with time zone	Time when the currently active query was started, or if **state** is not **active**, when the last query was started.
state_change	timestamp with time zone	Time when the last **status** was changed.
waiting	boolean	Specifies whether the background is currently waiting for a lock. If the background is currently waiting for a lock, the value is **true**. Otherwise, the value is **false**. If a lock is being waited for on a DN, determine the DN based on the **wait_status** column in the pg_thread_wait_status view and query the **waiting** column in pg_stat_activity on the DN to check whether a lock wait occurs.
enqueue	text	Queuing status of a statement. Its value can be: • **waiting in queue**: The statement is in the queue. • **Empty**: The statement is running.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
state	text	Background status. Its value can be: • **active**: The background is executing a query. • **idle**: The background is waiting for a new client command. • **idle in transaction**: The background is in a transaction, but there is no statement being executed in the transaction. • **idle in transaction** (aborted): The background is in a transaction, but there are statements failed in the transaction. • **fastpath function call**: The background is executing a fast-path function. • **disabled**: This state is reported if **track_activities** is disabled in this background. **NOTE** Only system administrators can view the session status of their accounts. The state information of other accounts is empty. For example, after user **judy** is connected to the database, the state information of user **joe** and the initial user **omm** in **pgxc_stat_activity** is empty. SELECT datname, usename, usesysid, state,pid FROM pgxc_stat_activity; datname | usename | usesysid | state | pid ----------+---------+----------+--------+----------------- testdb | omm | 10 | | 139968752121616 testdb | omm | 10 | | 139968903116560 db_tpcds | judy | 16398 | active | 139968391403280 testdb | omm | 10 | | 139968643069712 testdb | omm | 10 | | 139968680818448 testdb | joe | 16390 | | 139968563377936 (6 rows)
resource_pool	name	Resource pool used by the user.
query_id	bigint	ID of a query statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
query	text	Latest query in the background. If **state** is **active**, this column shows the ongoing query. In all other states, it shows the last query that was executed.
connection_info	text	A string in JSON format recording the driver type, driver version, driver deployment path, and process owner of the connected database. For details, see the GUC parameter **connection_info**.
global_sessionid	text	Global session ID.
unique_sql_id	bigint	Unique SQL statement ID.
trace_id	text	Driver-specific trace ID, which is associated with an application request.
top_xid	xid	Top-level transaction ID of a transaction.
current_xid	xid	Current transaction ID of a transaction.
xlog_quantity	bigint	Size of Xlogs currently used by a transaction, in bytes.

7.3.13.203 PGXC_STAT_BAD_BLOCK

PGXC_STAT_BAD_BLOCK displays statistics about page verification failures after all nodes in a cluster are started. Only users with the SYSADMIN or MONADMIN permission can query this view. For details about the columns, see **Table 7-489**.

Table 7-489 PGXC_STAT_BAD_BLOCK columns

Name	Type	Description
nodename	text	Node name.
databaseid	integer	Database OID.
tablespaceid	integer	Tablespace OID.
relfilenode	integer	File object ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
forknum	integer	File type. The options are as follows: • **0**: main data file. • **1**: FSM file. • **2**: VM file. • **3**: BCM file.
error_count	integer	Number of verification failures.
first_time	timestamp with time zone	Time of the first verification failure.
last_time	timestamp with time zone	Time of the latest verification failure.

7.3.13.204 PGXC_SQL_COUNT

PGXC_SQL_COUNT displays node-level and user-level statistical results for the SELECT, INSERT, UPDATE, DELETE, and MERGE INTO statements in real time, identifies query types with heavy load, and measures the capability of a cluster or node to perform a specific type of query. For example, you can calculate QPS based on the quantities of the five types of SQL statements at certain time points. **USER1 SELECT** is counted as **X1** at T1 and as **X2** at T2. The **SELECT** QPS of the user can be calculated as follows: (X2 – X1)/(T2 – T1). In this way, the system can draw cluster-user-level QPS curve graphs and determine cluster throughput, tracing changes in the service load of each user. If there are drastic changes, the system can locate the specific statement type (such as SELECT, INSERT, UPDATE, DELETE, and MERGE INTO). You can also observe QPS curves to determine the time points when problems occur and then locate the problems using other tools. The curves provide a basis for optimizing cluster performance and locating problems. Only users with the SYSADMIN or MONADMIN permission can query this view. The query can be performed only on the CN. The **execute direct on (dn)'select * from PGXC_SQL_COUNT';** statement is not supported.

Columns in the PGXC_SQL_COUNT view are the same as those in the GS_SQL_COUNT view. For details about the columns, see **GS_SQL_COUNT**.

NOTE

If a MERGE INTO statement can be pushed down and a DN receives it, the statement will be counted on the DN and the value of the **mergeinto_count** column will be incremented by 1. If pushdown is not allowed, the DN will receive an UPDATE or INSERT statement. In this case, the **update_count** or **insert_count** column will be incremented by 1.

7.3.13.205 PGXC_THREAD_WAIT_STATUS

PGXC_THREAD_WAIT_STATUS displays all the call layer hierarchy relationship between threads of the SQL statements on all the nodes in a cluster, and the waiting status of the block for each thread, so that you can easily locate the causes of process response failures and similar phenomena. For details about the columns, see **Table 7-490**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The definitions of PGXC_THREAD_WAIT_STATUS and PG_THREAD_WAIT_STATUS are the same, because the essence of PGXC_THREAD_WAIT_STATUS is the query summary of PG_THREAD_WAIT_STATUS on each node in the cluster.

Table 7-490 PGXC_THREAD_WAIT_STATUS columns

Name	Type	Description
node_name	text	Current node name.
db_name	text	Database name.
thread_name	text	Thread name.
query_id	bigint	Query ID. It is equivalent to **debug_query_id**.
tid	bigint	Thread ID of the current thread.
sessionid	bigint	Session ID.
lwtid	integer	Lightweight thread ID of the current thread.
psessionid	bigint	Session ID.
tlevel	integer	Level of the streaming thread.
smpid	integer	Parallel thread ID.
wait_status	text	Waiting status of the current thread. For details, see **Table 7-470**.
wait_event	text	If **wait_status** is **acquire lock**, **acquire lwlock**, or **wait io**, this column describes the lock, lightweight lock, or I/O information. Otherwise, this column is empty.
locktag	text	Information about the lock that the current thread is waiting for.
lockmode	text	Lock mode that the current thread is waiting to obtain.
block_sessionid	bigint	ID of the session that blocks the current thread from obtaining the lock.
global_sessionid	text	Global session ID.

Example:

Assume that you run a statement on coordinator1, and no response is returned after a long period of time. In this case, establish another connection to coordinator1 to check the thread status on it.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
gaussdb=# SELECT * FROM pg_thread_wait_status WHERE query_id > 0;
-[ RECORD 1 ]----+-------------------------------------
node_name        | cn_5001
db_name          | tpcc_row
thread_name      | PostgreSQL JDBC Driver
query_id         | 72620544050065400
tid              | 140650239031040
sessionid        | 11680
lwtid            | 26762
psessionid       |
tlevel           | 0
smpid            | 0
wait_status      | wait node: dn_6007_6008_6009, total 1
wait_event       | wait node
locktag          |
lockmode         |
block_sessionid  |
global_sessionid | 1120683504:11680#0
(1 rows)
gaussdb=# SELECT * FROM pgxc_thread_wait_status WHERE query_id > 0;
-[ RECORD 1 ]----+-------------------------------------------------
node_name        | cn_5001
db_name          | tpcc_row
thread_name      | PostgreSQL JDBC Driver
query_id         | 72620544050081616
tid              | 140648290055936
sessionid        | 11680
lwtid            | 26839
psessionid       |
tlevel           | 0
smpid            | 0
wait_status      | wait node: dn_6004_6005_6006, total 2
wait_event       | wait node
locktag          |
lockmode         |
block_sessionid  |
global_sessionid | 1120683504:11680#0
(1 rows)
```

7.3.13.206 PGXC_TOTAL_MEMORY_DETAIL

PGXC_TOTAL_MEMORY_DETAIL displays memory usage in the cluster. The query can be performed only on the CN. The EXECUTE DIRECT ON (dn) 'SELECT * FROM PGXC_TOTAL_MEMORY_DETAIL'; statement is not supported. Only users with the SYSADMIN or MONADMIN permission can query this view. If the GUC parameter **enable_memory_limit** is set to **off**, this view is unavailable. For details about the columns, see **Table 7-491**.

Table 7-491 PGXC_TOTAL_MEMORY_DETAIL columns

Name	Type	Description
nodename	text	Node name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
memorytype	text	Memory name. • **max_process_memory**: memory occupied by the GaussDB instance • **process_used_memory**: memory occupied by the GaussDB process. • **max_dynamic_memory**: maximum dynamic memory. • **dynamic_used_memory**: used dynamic memory. • **dynamic_peak_memory**: dynamic memory peak. • **dynamic_used_shrctx**: maximum dynamic shared memory context. • **dynamic_peak_shrctx**: dynamic peak value of the shared memory context. • **max_backend_memory**: maximum memory that can be used when the HA port is used to execute services. • **backend_used_memory**: memory that has been used when the HA port is used to execute services. • **high_backend_thread_memory**: memory used by high-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **high_backend_thread_max_memory**: memory reserved for high-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **medium_backend_thread_memory**: memory used by medium-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **medium_backend_thread_max_memory**: memory reserved for medium-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
		• **low_backend_thread_memory**: memory used by low-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **low_backend_thread_max_memory**: memory reserved for low-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **max_shared_memory**: maximum shared memory. • **shared_used_memory**: used shared memory. • **max_sctpcomm_memory**: maximum memory allowed for the communications library. • **sctpcomm_used_memory**: memory used by the communications library. • **sctpcomm_peak_memory**: memory peak of the communications library. • **other_used_memory**: other used memory. • **llvm_used_memory**: memory occupied by the expression IR that is generated by Codegen and is not released.
memorymbytes	integer	Size of the used memory (unit: MB).

7.3.13.207 PGXC_VARIABLE_INFO

PGXC_VARIABLE_INFO records information about transaction IDs and OIDs of all nodes in a cluster. Only users with the SYSADMIN or MONADMIN permission can query this view. The query can be performed only on the CN. The **execute direct on (dn)'select * from PGXC_VARIABLE_INFO';** statement is not supported. For details about the columns, see **Table 7-492**.

Table 7-492 PGXC_VARIABLE_INFO columns

Name	Type	Description
node_name	text	Node name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
next_oid	oid	OID generated next time on the node.
next_xid	xid	Transaction ID generated next time on the node.
oldest_xid	xid	Oldest transaction ID on the node.
xid_vac_limit	xid	Critical point (transaction ID) that triggers forcible autovacuum.
oldest_xid_db	oid	OID of the database that has the minimum datafrozenxid on the node.
last_extend_csn_logpage	xid	Number of the last extended CSNlog page.
start_extend_csn_logpage	xid	Number of the page from which CSNlog extending starts.
next_commit_seqno	xid	CSN generated next time for the node.
latest_completed_xid	xid	Latest transaction ID on the node after the transaction commit or rollback.
startup_max_xid	xid	Last transaction ID before the node is powered off.

7.3.13.208 PLAN_TABLE

PLAN_TABLE displays plan information collected by EXPLAIN PLAN. Plan information is in a session-level lifecycle. After a session exits, the data will be deleted. Data is isolated between sessions and between users. This view exists in both PG_CATALOG and SYS schemas.

Table 7-493 PLAN_TABLE columns

Name	Type	Description
statement_id	character varying(30)	Query tag specified by a user.
plan_id	bigint	Query ID.
id	integer	ID of each operator in a generated plan.
operation	character varying(30)	Operation description of an operator in a plan.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
options	character varying(255)	Operation action.
object_name	name	Object name corresponding to the operation, which is not the object alias used in the query. The object name is defined by users.
object_type	character varying(30)	Object type.
object_owner	name	Schema to which the object belongs. It is defined by users.
projection	character varying(4000)	Returned column information.
cost	double precision	Execution cost estimated by the optimizer for an operator.
cardinality	double precision	Number of rows estimated by the optimizer for an operator.
remarks	character varying(4000)	Not supported yet. Its value is **NULL**.
timestamp	date	Not supported yet. Its value is **NULL**. **NOTE** During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of this column is changed to **datea**.
object_node	character varying(128)	Not supported yet. Its value is **NULL**.
object_alias	character varying(261)	Not supported yet. Its value is **NULL**.
object_instance	numeric	Not supported yet. Its value is **NULL**.
optimizer	character varying(255)	Not supported yet. Its value is **NULL**.
search_columns	numeric	Not supported yet. Its value is **NULL**.
parent_id	numeric	Not supported yet. Its value is **NULL**.
depth	numeric	Not supported yet. Its value is **NULL**.
position	numeric	Not supported yet. Its value is **NULL**.
bytes	numeric	Not supported yet. Its value is **NULL**.
other_tag	character varying(255)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
partition_start	character varying(255)	Not supported yet. Its value is **NULL**.
partition_stop	character varying(255)	Not supported yet. Its value is **NULL**.
partition_id	numeric	Not supported yet. Its value is **NULL**.
other	character varying	Not supported yet. Its value is **NULL**.
other_xml	clob	Not supported yet. Its value is **NULL**.
distribution	character varying(20)	Not supported yet. Its value is **NULL**.
cpu_cost	numeric	Not supported yet. Its value is **NULL**.
io_cost	numeric	Not supported yet. Its value is **NULL**.
temp_space	numeric	Not supported yet. Its value is **NULL**.
access_predicates	character varying(4000)	Not supported yet. Its value is **NULL**.
filter_predicates	character varying(4000)	Not supported yet. Its value is **NULL**.
time	numeric	Not supported yet. Its value is **NULL**.
qblock_name	character varying(128)	Not supported yet. Its value is **NULL**.

NOTE

- A valid **object_type** value consists of a relkind type defined in **PG_CLASS** (**TABLE** for ordinary tables, **INDEX**, **SEQUENCE**, **VIEW**, **FOREIGN TABLE**, **COMPOSITE TYPE**, or **TOASTVALUE** for TOAST tables) and the rtekind type used in the plan (**SUBQUERY**, **JOIN**, **FUNCTION**, **VALUES**, **CTE**, or **REMOTE_QUERY**).
- For RangeTableEntry (RTE), **object_owner** is the object description used in the plan. Non-user-defined objects do not have **object_owner**.
- Information in the **statement_id**, **object_name**, **object_owner**, and **projection** columns is stored in letter cases specified by users and information in other columns is stored in uppercase.
- **PLAN_TABLE** supports only SELECT and DELETE and does not support other DML operations.

7.3.13.209 PV_FILE_STAT

PV_FILE_STAT records statistics about data file I/O to indicate I/O performance and detect performance problems such as abnormal I/O operations. For details about the columns, see **Table 7-494**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-494 PV_FILE_STAT columns

Name	Type	Description
filenum	oid	File ID
dbid	oid	Database ID
spcid	oid	Tablespace ID
phyrds	bigint	Number of times of reading physical files
phywrts	bigint	Number of times of writing into physical files
phyblkrd	bigint	Number of times of reading physical file blocks
phyblkwrt	bigint	Number of times of writing into physical file blocks
readtim	bigint	Total duration of reading, in microseconds
writetim	bigint	Total duration of writing, in microseconds
avgiotim	bigint	Average duration of reading and writing, in microseconds
lstiotim	bigint	Duration of the last file reading, in microseconds
miniotim	bigint	Minimum duration of reading and writing, in microseconds
maxiowtm	bigint	Maximum duration of reading and writing, in microseconds

7.3.13.210 PV_INSTANCE_TIME

Records time consumption information of the current node. For details about the columns, see **Table 7-495**. The time consumption information is classified into the following types:

- DB_TIME: effective time spent by jobs in multi-core scenarios
- CPU_TIME: CPU time spent
- EXECUTION_TIME: time spent in executors
- PARSE_TIME: time spent in parsing SQL statements
- PLAN_TIME: time spent on generating plans
- REWRITE_TIME: time spent for SQL rewriting
- PL_EXECUTION_TIME: execution time of the PL/SQL stored procedure

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- PL_COMPILATION_TIME: compilation time of the PL/SQL stored procedure
- NET_SEND_TIME: time spent on the network
- DATA_IO_TIME: I/O time spent

Table 7-495 PV_INSTANCE_TIME columns

Name	Category	Description
stat_id	integer	Statistics ID
stat_name	text	Type name
value	bigint	Time value, in µs

7.3.13.211 PV_OS_RUN_INFO

PV_OS_RUN_INFO displays the running status of the OS. For details about the columns, see **Table 7-496**.

Table 7-496 PV_OS_RUN_INFO columns

Name	Type	Description
id	integer	ID
name	text	Name of the OS running status.
value	numeric	Value of the OS running status.
comments	text	Remarks of the OS running status.
cumulative	boolean	Specifies whether the value of the OS running status is cumulative.

7.3.13.212 PV_REDO_STAT

PV_REDO_STAT displays the log replay of session threads. For details about the columns, see **Table 7-497**.

Table 7-497 PV_REDO_STAT columns

Name	Type	Description
phywrts	bigint	Number of times that data is written during log replay
phyblkwrt	bigint	Number of data blocks written during log replay
writetim	bigint	Total time required for writing data during log replay.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
avgiotim	bigint	Average time required for writing data during log replay.
lstiotim	bigint	Time consumed by the last data write operation during log replay
miniotim	bigint	Minimum time consumed by a single data write operation during log replay
maxiowtm	bigint	Maximum time consumed by a single data write operation during log replay

7.3.13.213 PV_SESSION_MEMORY

PV_SESSION_MEMORY collects statistics about memory usage at the session level, including all the memory allocated to gaussdb and Stream threads on DNs for jobs currently executed by users. For details about the columns, see **Table 7-498**.

Table 7-498 PV_SESSION_MEMORY columns

Name	Type	Description
sessid	text	Thread start time and ID.
init_mem	integer	Memory allocated to the currently executed jobs before they enter the executor, in MB.
used_mem	integer	Memory allocated to the currently executed jobs, in MB.
peak_mem	integer	Peak memory allocated to the currently executed jobs, in MB.

7.3.13.214 PV_SESSION_MEMORY_CONTEXT

PV_SESSION_MEMORY_CONTEXT displays the memory usage of all sessions based on the MemoryContext node. This view is valid only when **enable_thread_pool** is set to **on**.

The memory context **TempSmallContextGroup** collects information about all memory contexts whose value in the **totalsize** column is less than 8192 bytes in the current thread, and the number of the collected memory contexts is recorded in the **usedsize** column. Therefore, the **totalsize** and **freesize** columns for **TempSmallContextGroup** in the view display the corresponding information about all the memory contexts whose value in the **totalsize** column is less than 8192 bytes in the current thread, and the **usedsize** column displays the number of these memory contexts. For details about the columns, see **Table 7-499**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-499 PV_SESSION_MEMORY_CONTEXT columns

Name	Type	Description
sessid	text	Session start time+session ID (character string: *timestamp.sessionid*).
threadid	bigint	ID of the thread bound to a session (**-1** if no thread is bound).
contextname	text	Name of the memory context.
level	smallint	Hierarchy of the memory context.
parent	text	Name of the parent memory context.
totalsize	bigint	Total size of the memory context, in bytes.
freesize	bigint	Total size of released memory in the current memory context, in bytes.
usedsize	bigint	Size of used memory in the memory context, in bytes. For **TempSmallContextGroup**, this parameter specifies the number of collected memory contexts.

NOTICE

This view is an O&M view and is used to locate memory problems. Do not query this view concurrently. If you query this view concurrently, the waiting time for new connections increases as the number of concurrent connections increases. As a result, new connections cannot be connected for a long time.

7.3.13.215 PV_SESSION_MEMORY_DETAIL

PV_SESSION_MEMORY_DETAIL displays the memory usage of sessions based on the MemoryContext node. When **enable_thread_pool** is set to **on**, this view contains memory usage of all threads and sessions.

The memory context **TempSmallContextGroup** collects information about all memory contexts whose value in the **totalsize** column is less than 8192 bytes in the current thread, and the number of the collected memory contexts is recorded in the **usedsize** column. Therefore, the **totalsize** and **freesize** columns for **TempSmallContextGroup** in the view display the corresponding information about all the memory contexts whose value in the **totalsize** column is less than 8192 bytes in the current thread, and the **usedsize** column displays the number of these memory contexts.

You can run the SELECT * FROM PV_SESSION_MEMCTX_DETAIL(*threadid*, ''); statement to record information about all memory contexts of a thread into the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

threadid_timestamp.**log** file in the *$GAUSSLOG*/**gs_log/***${node_name}*/**dumpmem** directory. *threadid* can be obtained from **sessid** in Table 7-500.

Table 7-500 PV_SESSION_MEMORY_DETAIL columns

Name	Type	Description
sessid	text	• When the thread pool is disabled (**enable_thread_pool = off**), this column indicates the thread start time + session ID (string: **timestamp.sessionid**). • When the thread pool is enabled (**enable_thread_pool = on**): If the memory context is at the thread level, this column indicates the thread start time + thread ID (string: **timestamp.threadid**). If the memory context is at the session level, the column indicates the thread start time + session ID (string: **timestamp.sessionid**).
sesstype	text	Thread name.
contextname	text	Name of the memory context.
level	smallint	Hierarchy of the memory context.
parent	text	Name of the parent memory context.
totalsize	bigint	Total size of the memory context, in bytes.
freesize	bigint	Total size of released memory in the current memory context, in bytes.
usedsize	bigint	Size of used memory in the memory context, in bytes. For **TempSmallContextGroup**, this parameter specifies the number of collected memory contexts.

NOTICE

This view is an O&M view and is used to locate memory problems. Do not query this view concurrently. If you query this view concurrently, the waiting time for new connections increases as the number of concurrent connections increases. As a result, new connections cannot be connected for a long time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

7.3.13.216 PV_SESSION_STAT

PV_SESSION_STAT collects statistics about session states based on session threads or the AutoVacuum thread. For details about the columns, see **Table 7-501**.

Table 7-501 PV_SESSION_STAT columns

Name	Type	Description
sessid	text	Thread ID and start time
statid	integer	Statistics ID
statname	text	Name of the statistics session
statunit	text	Unit of the statistics session
value	bigint	Value of the statistics session

7.3.13.217 PV_SESSION_TIME

PV_SESSION_TIME displays the running time of session threads and time consumed in each execution phase. For details about the columns, see **Table 7-502**.

Table 7-502 PV_SESSION_TIME columns

Name	Type	Description
sessid	text	Thread ID and start time
stat_id	integer	Statistics ID

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
stat_name	text	Name of the session type • **DB_TIME**: effective time spent by jobs in multi-core scenarios • **CPU_TIME**: CPU time spent • **EXECUTION_TIME**: time spent within executors • **PARSE_TIME**: time spent in parsing SQL statements • **PLAN_TIME**: time spent on generating plans • **REWRITE_TIME**: time spent on rewriting SQL statements • **PL_EXECUTION_TIME**: execution time of the PL/pgSQL stored procedure • **PL_COMPILATION_TIME**: compilation time of the PL/pgSQL stored procedure • **NET_SEND_TIME**: time spent on the network. • **DATA_IO_TIME**: I/O time spent
value	bigint	Session value

7.3.13.218 PV_THREAD_MEMORY_CONTEXT

PV_THREAD_MEMORY_CONTEXT displays the memory usage of all threads based on the MemoryContext node. This view is equivalent to the **PV_SESSION_MEMORY_DETAIL** view when **enable_thread_pool** is set to **off**. For details about the columns, see **Table 7-503**.

The memory context **TempSmallContextGroup** collects information about all memory contexts whose value in the **totalsize** column is less than 8192 bytes in the current thread, and the number of the collected memory contexts is recorded in the **usedsize** column. Therefore, the **totalsize** and **freesize** columns for **TempSmallContextGroup** in the view display the corresponding information about all the memory contexts whose value in the **totalsize** column is less than 8192 bytes in the current thread, and the **usedsize** column displays the number of these memory contexts.

Table 7-503 PV_THREAD_MEMORY_CONTEXT columns

Name	Type	Description
threadid	text	Thread start time and ID (string: *timestamp.sessionid*).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
tid	bigint	Thread ID.
thrdtype	text	Thread type It can be any thread type in the system, such as wlmmonitor.
contextname	text	Name of the memory context.
level	smallint	Hierarchy of the memory context.
parent	text	Name of the parent memory context.
totalsize	bigint	Total size of the memory context, in bytes.
freesize	bigint	Total size of released memory in the current memory context, in bytes.
usedsize	bigint	Size of used memory in the memory context, in bytes. For **TempSmallContextGroup**, this parameter specifies the number of collected memory contexts.

7.3.13.219 PV_TOTAL_MEMORY_DETAIL

PV_TOTAL_MEMORY_DETAIL displays statistics about memory usage of the current database node in the unit of MB. If the GUC parameter **enable_memory_limit** is set to **off**, this view is unavailable. For details about the columns, see **Table 7-504**.

Table 7-504 PV_TOTAL_MEMORY_DETAIL columns

Name	Type	Description
nodename	text	Node name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
memorytype	text	Memory type. The value must be one of the following: • **max_process_memory**: memory occupied by a GaussDB cluster instance. • **process_used_memory**: memory occupied by a GaussDB process. • **max_dynamic_memory**: maximum dynamic memory. • **dynamic_used_memory**: used dynamic memory. • **dynamic_peak_memory**: dynamic peak memory. • **dynamic_used_shrctx**: context of the used dynamic shared memory. • **dynamic_peak_shrctx**: dynamic peak value of the shared memory context. • **max_backend_memory**: maximum memory that can be used when the HA port is used to execute services. • **backend_used_memory**: memory that has been used when the HA port is used to execute services. • **high_backend_thread_memory**: memory used by high-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **high_backend_thread_max_memory**: memory reserved for high-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **medium_backend_thread_memory**: memory used by medium-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **medium_backend_thread_max_memory**: memory reserved for medium-priority threads. This is valid only when

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
		enable_threads_memory_reserve is set to **on**. • **low_backend_thread_memory**: memory used by low-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **low_backend_thread_max_memory**: memory reserved for low-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **max_shared_memory**: maximum shared memory. • **shared_used_memory**: used shared memory. • **max_sctpcomm_memory**: maximum memory allowed for the communications library. • **sctpcomm_used_memory**: memory used by the communications library. • **sctpcomm_peak_memory**: memory peak of the communications library. • **other_used_memory**: other used memory. • **llvm_used_memory**: memory occupied by the expression IR that is generated by Codegen and is not released.
memorymbytes	integer	Size of allocated memory-typed memory.

7.3.13.220 SYS_DUMMY

SYS_DUMMY is automatically created by the database based on the data dictionary. It has only one text column in only one row for storing expression calculation results. It is accessible to all users. This view exists in both PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-505**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-505 SYS_DUMMY columns

Name	Type	Description
dummy	text	Expression calculation result

7.3.13.221 V_INSTANCE

V_INSTANCE displays instance information in current database. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-506**.

Table 7-506 V_INSTANCE columns

Name	Type	Description
instance_number	oid	OID of the current database.
instance_name	character varying(16)	Name of the current database.
host_name	character varying(64)	Host name
version	character varying(17)	Not supported yet. Its value is **NULL**.
version_legacy	character varying(17)	Not supported yet. Its value is **NULL**.
version_full	character varying(17)	Not supported yet. Its value is **NULL**.
startup_time	date	Not supported yet. Its value is **NULL**.
status	character varying(12)	Not supported yet. Its value is **NULL**.
parallel	character varying(3)	Not supported yet. Its value is **NULL**.
thread#	numeric	Not supported yet. Its value is **NULL**.
archiver	character varying(7)	Not supported yet. Its value is **NULL**.
log_switch_wait	character varying(15)	Not supported yet. Its value is **NULL**.
logins	character varying(10)	Not supported yet. Its value is **NULL**.
shutdown_pending	character varying(3)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
database_status	character varying(17)	Not supported yet. Its value is **NULL**.
instance_role	character varying(18)	Not supported yet. Its value is **NULL**.
active_state	character varying(9)	Not supported yet. Its value is **NULL**.
blocked	character varying(3)	Not supported yet. Its value is **NULL**.
con_id	numeric	Not supported yet. Its value is **NULL**.
instance_mode	character varying(11)	Not supported yet. Its value is **NULL**.
edition	character varying(7)	Not supported yet. Its value is **NULL**.
family	character varying(80)	Not supported yet. Its value is **NULL**.
database_type	character varying(15)	Not supported yet. Its value is **NULL**.

7.3.13.222 V_MYSTAT

V_MYSTAT displays statistics about all sessions in the database. Only system administrators can access this view. Common users can access this view only after being authorized. This view exists in both the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-507**.

Table 7-507 V_MYSTAT columns

Name	Type	Description
sid	numeric	Current session ID.
statistic#	numeric	Not supported yet. Its value is **NULL**.
value	numeric	Not supported yet. Its value is **NULL**.
con_id	numeric	Not supported yet. Its value is **NULL**.

7.3.13.223 V_SESSION

V_SESSION displays information about all current sessions. Only administrators can access this view. Common users can access this view only after being authorized. This view exists in both PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-508**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-508 V_SESSION columns

Name	Type	Description
saddr	raw	Not supported yet. Its value is **NULL**.
sid	bigint	Session ID.
serial#	integer	Sequence number of the active background thread, which is **0** in GaussDB
audsid	numeric	Not supported yet. Its value is **NULL**.
paddr	raw	Not supported yet. Its value is **NULL**.
schema#	numeric	Not supported yet. Its value is **NULL**.
schemaname	name	Name of the user logged in to the background.
user#	oid	OID of the user logged in to the background thread. The OID is **0** if the background thread is a global auxiliary thread.
username	name	Name of the user logged in to the background process. **username** is null if the background thread is a global auxiliary thread.
command	numeric	Not supported yet. Its value is **NULL**.
ownerid	numeric	Not supported yet. Its value is **NULL**.
taddr	character varying(16)	Not supported yet. Its value is **NULL**.
lockwait	character varying(16)	Not supported yet. Its value is **NULL**.
machine	text	Host name of the connected client, as reported by a reverse DNS lookup of **client_addr**. This column will be non-null only for IP connections and only when **log_hostname** is enabled.
sql_id	bigint	ID of a query statement.
client_info	text	Client information.
event	text	Queuing status of a statement. The value must be one of the following: • **waiting in queue**: The statement is in the queue. • **Empty**: The statement is running.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
sql_exec_start	timestamp with time zone	Time when the currently active query was started, or time when the last query was started if the value of **state** is not **active**.
program	text	Name of the application connected to the background.
status	text	Overall status of this background. The value must be one of the following: • **active**: The background is executing a query. • **idle**: The background is waiting for a new client command. • **idle in transaction**: The background is in a transaction, but there is no statement being executed in the transaction. • **idle in transaction (aborted)**: The background is in a transaction, but there are statements failed in the transaction. • **fastpath function call**: The background is executing a fast-path function. • **disabled**: This state is reported if **track_activities** is disabled in this background.
server	character varying(9)	Not supported yet. Its value is **NULL**.
pdml_status	character varying(8)	Specifies whether to enable a DML parallel execution in the current session.
port	numeric	Port number of the current session.
process	character varying(24)	Process ID of the current session.
logon_time	date	Login time of the current session. **TIMESTAMP (0) WITHOUT TIME ZONE** is displayed in an O-compatible database.
last_call_et	integer	Duration when the status of the current session changes last time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
osuser	character varying(128)	Not supported yet. Its value is **NULL**.
terminal	character varying(30)	Not supported yet. Its value is **NULL**.
type	character varying(10)	Not supported yet. Its value is **NULL**.
sql_address	raw	Not supported yet. Its value is **NULL**.
sql_hash_value	numeric	Not supported yet. Its value is **NULL**.
sql_child_number	numeric	Not supported yet. Its value is **NULL**.
sql_exec_id	numeric	Not supported yet. Its value is **NULL**.
prev_sql_addr	raw	Not supported yet. Its value is **NULL**.
prev_hash_value	numeric	Not supported yet. Its value is **NULL**.
prev_sql_id	character varying(13)	Not supported yet. Its value is **NULL**.
prev_child_number	numeric	Not supported yet. Its value is **NULL**.
prev_exec_start	date	Not supported yet. Its value is **NULL**. • **TIMESTAMP (0) WITHOUT TIME ZONE** is displayed in an O-compatible database.
prev_exec_id	numeric	Not supported yet. Its value is **NULL**.
plsql_entry_object_id	numeric	Not supported yet. Its value is **NULL**.
plsql_entry_subprogram_id	numeric	Not supported yet. Its value is **NULL**.
plsql_object_id	numeric	Not supported yet. Its value is **NULL**.
plsql_subprogram_id	numeric	Not supported yet. Its value is **NULL**.
module	character varying(64)	Not supported yet. Its value is **NULL**.
module_hash	numeric	Not supported yet. Its value is **NULL**.
action	character varying(64)	Not supported yet. Its value is **NULL**.
action_hash	numeric	Not supported yet. Its value is **NULL**.
fixed_table_sequence	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
row_wait_obj#	numeric	Not supported yet. Its value is **NULL**.
row_wait_file#	numeric	Not supported yet. Its value is **NULL**.
row_wait_block#	numeric	Not supported yet. Its value is **NULL**.
row_wait_row#	numeric	Not supported yet. Its value is **NULL**.
top_level_call#	numeric	Not supported yet. Its value is **NULL**.
pdml_enabled	character varying(3)	Not supported yet. Its value is **NULL**.
failover_type	character varying(13)	Not supported yet. Its value is **NULL**.
failover_method	character varying(10)	Not supported yet. Its value is **NULL**.
failed_over	character varying(3)	Not supported yet. Its value is **NULL**.
resource_consumer_group	character varying(32)	Not supported yet. Its value is **NULL**.
pddl_status	character varying(8)	Not supported yet. Its value is **NULL**.
pq_status	character varying(8)	Not supported yet. Its value is **NULL**.
current_queue_duration	numeric	Not supported yet. Its value is **NULL**.
client_identifier	character varying(64)	Not supported yet. Its value is **NULL**.
blocking_session_status	character varying(11)	Not supported yet. Its value is **NULL**.
blocking_instance	numeric	Not supported yet. Its value is **NULL**.
blocking_session	numeric	Not supported yet. Its value is **NULL**.
final_blocking_session_status	character varying(11)	Not supported yet. Its value is **NULL**.
final_blocking_instance	numeric	Not supported yet. Its value is **NULL**.
final_blocking_session	numeric	Not supported yet. Its value is **NULL**.
seq#	numeric	Not supported yet. Its value is **NULL**.
event#	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
p1text	character varying(64)	Not supported yet. Its value is **NULL**.
p1	numeric	Not supported yet. Its value is **NULL**.
p1raw	raw	Not supported yet. Its value is **NULL**.
p2text	character varying(64)	Not supported yet. Its value is **NULL**.
p2	numeric	Not supported yet. Its value is **NULL**.
p2raw	raw	Not supported yet. Its value is **NULL**.
p3text	character varying(64)	Not supported yet. Its value is **NULL**.
p3	numeric	Not supported yet. Its value is **NULL**.
p3raw	raw	Not supported yet. Its value is **NULL**.
wait_class_id	numeric	Not supported yet. Its value is **NULL**.
wait_class#	numeric	Not supported yet. Its value is **NULL**.
wait_class	character varying(64)	Not supported yet. Its value is **NULL**.
wait_time	numeric	Not supported yet. Its value is **NULL**.
seconds_in_wait	numeric	Not supported yet. Its value is **NULL**.
state	character varying(19)	Not supported yet. Its value is **NULL**.
wait_time_micro	numeric	Not supported yet. Its value is **NULL**.
time_remaining_micro	numeric	Not supported yet. Its value is **NULL**.
time_since_last_wait_micro	numeric	Not supported yet. Its value is **NULL**.
service_name	character varying(64)	Not supported yet. Its value is **NULL**.
sql_trace	character varying(8)	Not supported yet. Its value is **NULL**.
sql_trace_waits	character varying(5)	Not supported yet. Its value is **NULL**.
sql_trace_binds	character varying(5)	Not supported yet. Its value is **NULL**.
sql_trace_plan_stats	character varying(10)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
session_edition_id	numeric	Not supported yet. Its value is **NULL**.
creator_addr	raw	Not supported yet. Its value is **NULL**.
creator_serial#	numeric	Not supported yet. Its value is **NULL**.
ecid	character varying(64)	Not supported yet. Its value is **NULL**.
sql_translation_profile_id	numeric	Not supported yet. Its value is **NULL**.
pga_tunable_mem	numeric	Not supported yet. Its value is **NULL**.
shard_ddl_status	character varying(8)	Not supported yet. Its value is **NULL**.
con_id	numeric	Not supported yet. Its value is **NULL**.
external_name	character varying(1024)	Not supported yet. Its value is **NULL**.
plsql_debugger_connected	character varying(5)	Not supported yet. Its value is **NULL**.

NOTE

During database initialization, if the GUC parameter **mapping_date_to_datea** is set to **on**, the type of the **logon_time** column in the view is changed to **datea**.

7.3.13.224 V$GLOBAL_OPEN_CURSOR

V$GLOBAL_OPEN_CURSOR displays information about all opened cursors on all current nodes. For details about the columns, see **Table 7-509**.

Table 7-509 V$GLOBAL_OPEN_CURSOR columns

Name	Type	Description
node_name	name	Node name.
saddr	raw	Not supported yet. Its value is **NULL**.
sid	numeric	Session ID.
user_name	character varying(128)	Username.
address	raw	Not supported yet. Its value is **NULL**.
hash_value	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
sql_id	character varying(13)	ID of a query statement.
sql_text	character varying(60)	First 60 bytes of the SQL text of a cursor.
last_sql_active_time	date	Not supported yet. Its value is **NULL**. **TIMESTAMP (0) WITHOUT TIME ZONE** is displayed in an O-compatible database.
sql_exec_id	numeric	Not supported yet. Its value is **NULL**.
cursor_type	character varying(64)	Cursor type. • **OPEN-PL/SQL**: opened PL/SQL cursor. • **OPEN**: other open cursors.
child_address	raw	Not supported yet. Its value is **NULL**.
con_id	numeric	Not supported yet. Its value is **NULL**.
dbid	oid	ID of the database to which the cursor belongs.

7.3.13.225 V$GLOBAL_TRANSACTION

V$GLOBAL_TRANSACTION displays information about currently active global transactions. By default, only users with the SYSADMIN permission can access the system view. Common users must be authorized to access the system view. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-510**.

Table 7-510 V$GLOBAL_TRANSACTION columns

Name	Type	Description
formatid	numeric	Global transaction format identifier. Not supported yet. Its value is **NULL**.
globalid	raw	Global transaction identifier.
branchid	raw	Global transaction branch identifier. Not supported yet. Its value is **NULL**. Each individual transaction of a global transaction is called a branch.
branches	numeric	Total number of global transaction branches.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
refcount	numeric	Number of siblings for the global transaction (must be the same as branches).
preparecount	numeric	Number of prepared global transaction branches. If the value of **system_view_version** is greater than **0** and no prepared global transaction branch exists, the value is **0**. Otherwise, the value is **NULL**.
state	character varying(38)	Status of the global transaction branch.
flags	numeric	Numeric representation of the status.
coupling	character varying(15)	Specifies whether the branch is free (`FREE`), loosely coupled (`LOOSELY COUPLED`), or tightly coupled (TIGHTLY COUPLED). Not supported yet. Its value is **NULL**.
con_id	numeric	ID of the container related to data. Not supported. The value is **0**.

7.3.13.226 V$LOCK

V$LOCK displays information about locks held by open transactions. By default, only users with the SYSADMIN permission can access the system view. Common users must be authorized to access the system view. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-511**.

Table 7-511 V$LOCK columns

Name	Type	Description
addr	raw	Not supported yet. Its value is **NULL**.
kaddr	raw	Not supported yet. Its value is **NULL**.
sid	numeric	ID of the session that owns the lock.
type	character varying(2)	TM or TX. • TM: corresponding relation lock in pg_locks. • TX: other locks.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
id1	numeric	• TM lock: **relationId** corresponding to the object ID. • TX lock: not supported. The value is **0**.
id2	numeric	• TM lock: The default value is **0**. • TX lock: not supported. The value is **0**.
lmode	numeric	For details, see **Table 7-512**.
request	numeric	For details, see **Table 7-512**.
ctime	numeric	Not supported yet. Its value is **NULL**.
block	numeric	Specifies whether it is blocked by other sessions. **1**: blocked; **0**: not blocked.
con_id	numeric	Not supported yet. Its value is **NULL**.
blocksid	bigint	ID of another thread blocked by the current thread.

Table 7-512 Lock mode

Level	GaussDB
-	NULL
0	INVALID
1	AccessShare
2	RowShare
3	RowExclusive
4	ShareUpdateExclusive
5	Share
6	ShareRowExclusive
7	Exclusive
8	AccessExclusive

7.3.13.227 V$NLS_PARAMETERS

V$NLS_PARAMETERS displays the National Language Support (NLS) parameters and parameter values configured for the database. This view exists in the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

PG_CATALOG and SYS schemas and all users can access this view. For details about the columns, see **Table 7-513**.

Table 7-513 V$NLS_PARAMETERS columns

Name	Type	Description
parameter	character varying(64)	NLS parameter name.
value	character varying(64)	NLS parameter value.
con_id	numeric	Not supported. The value is **0**.

7.3.13.228 V$OPEN_CURSOR

V$OPEN_CURSOR displays information about cursors opened by all sessions on the current node. By default, only users with the SYSADMIN permission can access the system view. Common users must be authorized to access the system view. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-514**.

Table 7-514 V$OPEN_CURSOR columns

Name	Type	Description
saddr	raw	Not supported yet. Its value is **NULL**.
sid	numeric	Session ID.
user_name	character varying(128)	Username.
address	raw	Not supported yet. Its value is **NULL**.
hash_value	numeric	Not supported yet. Its value is **NULL**.
sql_id	character varying(13)	ID of a query statement.
sql_text	character varying(60)	First 60 bytes of the SQL text of a cursor.
last_sql_active_time	date	Not supported yet. Its value is **NULL**. **TIMESTAMP (0) WITHOUT TIME ZONE** is displayed in an O-compatible database.
sql_exec_id	numeric	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
cursor_type	character varying(64)	Cursor type. • **OPEN-PL/SQL**: opened PL/SQL cursor. • **OPEN**: other open cursors.
child_address	raw	Not supported yet. Its value is **NULL**.
con_id	numeric	Not supported yet. Its value is **NULL**.
dbid	oid	ID of the database to which the cursor belongs.

7.3.13.229 V$SESSION_WAIT

V$SESSION_WAIT displays the current wait event or the last wait event of each session of each user. By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-515**.

Table 7-515 V$SESSION_WAIT columns

Name	Type	Description
sid	numeric	Session ID, which is mapped to the **V$SESSION.SID** column.
seq#	numeric	Not supported yet. Its value is **NULL**.
event	character varying(64)	If the session is waiting, the resource or event that is waiting for is displayed. If the session is not waiting, the resource or event that is waiting for the last time is displayed.
p1text	character varying(64)	Not supported yet. Its value is **NULL**.
p1	numeric	Not supported yet. Its value is **NULL**.
p1raw	raw	Not supported yet. Its value is **NULL**.
p2text	character varying(64)	Not supported yet. Its value is **NULL**.
p2	numeric	Not supported yet. Its value is **NULL**.
p2raw	raw	Not supported yet. Its value is **NULL**.
p3text	character varying(64)	Not supported yet. Its value is **NULL**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
p3	numeric	Not supported yet. Its value is **NULL**.
p3raw	raw	Not supported yet. Its value is **NULL**.
wait_class_id	numeric	Not supported yet. Its value is **NULL**.
wait_class#	numeric	Not supported yet. Its value is **NULL**.
wait_class	character varying(64)	Name of a wait event type
wait_time	numeric	If the session is waiting, the value is **0**. If the session is not waiting, the options are as follows: ● **>0**: duration of the last wait (unit: centisecond). ● **–1**: The duration of the last wait is less than a centisecond. ● **–2**: The **TIMED_STATISTICS** parameter is set to **false**. This column has been deprecated and replaced by **WAIT_TIME_MICRO** and **STATE**.
second_in_wait	numeric	If the session is waiting, the value is the amount of time to wait. If the session is not waiting, the value is the amount of time since the last wait started. This column has been deprecated and replaced by the **WAIT_TIME_MICRO** and **TIME_SINCE_LAST_WAIT_MICRO** columns.
state	character varying(64)	Waiting state. The options are as follows: ● **WAITING**: The session is waiting. ● **WAITED UNKNOWN TIME**: The duration of the last wait is unknown. This value is used when **TIMED_STATISTICS** is set to **false**. ● **WAITED SHORT TIME**: The duration of the last wait is less than a centisecond. ● **WAITED KNOWN TIME**: duration of the last wait specified in the **WAIT_TIME** column.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
wait_time_micro	numeric	Wait time, in microseconds. If the session is waiting, the value is the time spent in the current wait. If the session is not waiting, the value is the time spent for the last wait.
time_remaining_micro	numeric	Not supported yet. Its value is **NULL**.
time_since_last_wait_micro	numeric	Time elapsed since the last wait ended, in microseconds. If the session is waiting, the value is **0**.
con_id	numeric	Not supported. The value is **0**.

7.3.13.230 V$SYSSTAT

V$SYSSTAT displays the resource usage of the entire database instance since the database instance is started. By default, only the initial user or MONADMIN can access the database. Other users can access the database only after being granted the MONADMIN permission. This view exists in the PG_CATALOG and SYS schemas. Contact an administrator to enable the memory resource management function in the Arm environment. For details about the columns, see **Table 7-516**.

Table 7-516 V$SYSSTAT columns

Name	Type	Description
statistic#	numeric	Statistics ID.
name	character varying(64)	Statistical item name.
class	numeric	Not supported yet. Its value is **NULL**.
value	numeric	Statistical item value.
stat_id	numeric	Not supported yet. Its value is **NULL**.
con_id	numeric	Not supported. The value is **0**.

7.3.13.231 V$SYSTEM_EVENT

V$SYSTEM_EVENT displays information about all the wait events (summary of each wait event since the instance is started). By default, only users with the SYSADMIN permission can access this view. Common users can access the view only after being authorized. This view exists in the PG_CATALOG and SYS schemas. For details about the columns, see **Table 7-517**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 7-517 V$SYSTEM_EVENT columns

Name	Type	Description
event	character varying(64)	Wait event name
total_waits	numeric	Total number of wait events
total_timeouts	numeric	Total number of timeout events
time_waited	numeric	Total time (in centiseconds) of wait events
average_wait	numeric	Average time (in centiseconds) of wait events
time_waited_micro	numeric	Total time (in microseconds) of wait events
total_waits_fg	numeric	Not supported yet. Its value is **NULL**.
total_timeouts_fg	numeric	Not supported yet. Its value is **NULL**.
time_waited_fg	numeric	Not supported yet. Its value is **NULL**.
average_wait_fg	numeric	Not supported yet. Its value is **NULL**.
time_waited_micro_fg	numeric	Not supported yet. Its value is **NULL**.
event_id	numeric	Not supported yet. Its value is **NULL**.
wait_class_id	numeric	Not supported yet. Its value is **NULL**.
wait_class#	numeric	Not supported yet. Its value is **NULL**.
wait_class	character varying(64)	Class name of a wait event
con_id	numeric	Not supported. The value is **0**.

7.3.13.232 V$VERSION

V$VERSION displays the version number of the database. This view exists in the PG_CATALOG and SYS schemas and all users can access this view. For details about the columns, see **Table 7-518**.

Table 7-518 V$VERSION columns

Name	Type	Description
banner	character varying(80)	Component name and version number

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
banner_full	character varying(160)	Database version and specific version number
banner_legacy	character varying(80)	Database version
con_id	numeric	Not supported. It is always **0**.

7.3.14 Deprecated System Views/System Catalogs

- The following views in GaussDB have been deprecated in the current version:
  - GS_IMCU_SLOT_STATUS
  - GS_IMCV_STATUS
  - GS_DATABASE_DETAILS
  - MTD_USER
- The following lists in GaussDB have been deprecated in the current version:
  - GS_PDB
  - GS_RESOURCE_PLANS
  - GS_RESOURCE_PLAN_DIRECTIVES

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8 Schemas

GaussDB describes the schemas of **Table 8-1**.

NOTE

Do not create service data of users in schemas that provide functional APIs. In addition, system objects such as functions and data types in these schemas cannot be depended on, including but not limited to tables and functions (such as dbe_* and pkg_*).

Table 8-1 Schemas supported in GaussDB

Schema	Description
blockchain	Stores the user history table that is automatically created when a tamper-proof table is created in the ledger database.
dbe_perf	Diagnoses performance issues and is also the data source of WDR snapshots. After a database is installed, only the initial user and MONADMIN have permission to view views and functions in the **DBE_PERF** schema by default.
snapshot	Manages data related to WDR snapshots. By default, the initial user or MONADMIN can access the data.
sqladvsior	Recommends distribution keys. For details, see **Distribution Key Recommendation Functions**.
sys	Provides the system information view APIs.
pg_catalog	Maintains system catalog information, including system catalogs and all built-in data types, functions, and operators.
public	Public schema, which is used to store public objects. If **search_path** is not specified and a schema with the same name exists, new tables (and other objects) are created in the schema by default. If no schema with the same name exists, new tables (and other objects) are automatically placed in this public schema.
pkg_service	Manages information about the package service.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Schema	Description
pkg_util	Manages information about the package tool.
dbe_raw	Advanced function package dbe_raw, which is used to convert raw data, obtain substrings, and calculate the length.
dbe_session	Advanced function package **dbe_session**, which is used to set the value of a specified attribute and support user query and verification.
dbe_lob	Advanced function package dbe_lob, which is used to read, write, and copy large files (CLOB/BLOB).
dbe_match	Advanced function package **dbe_match**, which is used to compare character string similarity.
dbe_task	Advanced function package **dbe_task**, which is used to schedule job tasks, including committing tasks, canceling tasks, synchronizing task status, and updating task information, so that the database can periodically execute specific tasks.
dbe_sql	Advanced function package **dbe_sql**, which is used to execute dynamic SQL statements and construct query and other commands during application running.
dbe_file	Advanced function package **dbe_file**, which is used to read, copy, write, delete, and rename external database files.
dbe_output	Advanced function package **dbe_output**, which is used to print output information.
dbe_random	Advanced function package **dbe_random**, which is used to generate random seeds and random numbers.
dbe_application_info	Advanced function package **dbe_application_info**, which is used for recording client information.
dbe_utility	Advanced function package **dbe_utility**, which is used to call the debugging tool in a stored procedure, for example, to print error stacks.
dbe_scheduler	Advanced function package **dbe_scheduler**, which is used to create scheduled jobs and enable the database to periodically execute specified tasks through programs and schedules. You can also perform external database tasks by authorizing and providing certificates.
information_schema	Stores information about objects defined in the current database.
dbe_sql_util	SQL O&M function, including the O&M API of SQL patches.
dbe_xmlgen	Advanced function package **dbe_xmlgen**, which is used to convert query results into XML character strings.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Schema	Description
dbe_describe	Advanced function package **dbe_describe**, which is used to query parameter information about stored procedures or functions.
dbe_stats	Advanced function package **dbe_stats**, which is used to manage statistics.

Table 8-2 Schemas disabled in GaussDB

Schema	Description
dbe_pldebugger	Debugs PL/SQL functions and stored procedures. Currently, this view is not supported. An error message "unsupported" is displayed when the API is called in this view.
db4ai	Manages data of different versions in AI training.

8.1 Information Schema

An information schema named INFORMATION_SCHEMA automatically exists in all databases. An information schema consists of a group of views that contain information about objects defined in the current database. The owner of this schema is the initial database user. However, all users have only the permission to use this schema and do not have the permission to create objects such as tables and functions.

Information schemas are compatible with PG, including constraint_table_usage, domain_constraints, domain_udt_usage, domains, enabled_roles, key_column_usage, parameters, referential_constraints, applicable_roles, administrable_role_authorizations, attributes, character_sets, check_constraint_routine_usage, check_constraints, collations, collation_character_set_applicability, column_domain_usage, column_privileges, column_udt_usage, columns, constraint_column_usage, role_column_grants, routine_privileges, role_routine_grants, routines, schemata, sequences, table_constraints, table_privileges, role_table_grants, tables, triggered_update_columns, udt_privileges, role_udt_grants, usage_privileges, role_usage_grants, user_defined_types, view_column_usage, view_routine_usage, view_table_usage, views, data_type_privileges, element_types, column_options, foreign_data_wrapper_options, foreign_data_wrappers, foreign_server_options, foreign_servers, foreign_table_options, foreign_tables, user_mapping_options, user_mappings, sql_features, sql_implementation_info, sql_languages, sql_packages, sql_parts, sql_sizing, and sql_sizing_profiles.

The following sections display only the views that are not listed in the preceding description.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTICE

Only the initial user can rebuild views in the information_schema schema using the CREATE OR REPLACE syntax. Exercise caution when performing this operation because unauthorized operations may occur. For example, if a user-defined function is bound to the definition of a rebuilt view, it may be used for privilege escalation.

8.1.1 INFORMATION_SCHEMA._PG_FOREIGN_DATA_WRAPPERS

Displays information about a foreign data wrapper, as described in **Table 8-3**. Only the SYSADMIN user has the permission to view this view.

Table 8-3 INFORMATION_SCHEMA._PG_FOREIGN_DATA_WRAPPERS columns

Name	Type	Description
oid	oid	OID of the foreign data wrapper.
fdwowner	oid	OID of the owner of the foreign data wrapper.
fdwoptions	text[]	Foreign-data wrapper specific option, expressed in a string in the format of *keyword*=*value*.
foreign_data_wrapper_catalog	information_schema.sql_identifier	Name of the database where the foreign data wrapper is located (always the current database).
foreign_data_wrapper_name	information_schema.sql_identifier	Name of the foreign data wrapper.
authorization_identifier	information_schema.sql_identifier	Role of the owner of the foreign data wrapper.
foreign_data_wrapper_language	information_schema.character_data	Programming language of the foreign data wrapper.

8.1.2 INFORMATION_SCHEMA._PG_FOREIGN_SERVERS

Displays information about a foreign server, as described in **Table 8-4**. Only the SYSADMIN user has the permission to view this view.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-4 INFORMATION_SCHEMA._PG_FOREIGN_SERVERS columns

Name	Type	Description
oid	oid	OID of the foreign server.
srvoptions	text[]	Foreign server specific options, expressed in a string in the format of *keyword*=*value*.
foreign_server_catalog	information_schema.sql_identifier	Name of the database where the foreign server is located (always the current database).
foreign_server_name	information_schema.sql_identifier	Name of the foreign server.
foreign_data_wrapper_catalog	information_schema.sql_identifier	Name of the database where the foreign data wrapper is located (always the current database).
foreign_data_wrapper_name	information_schema.sql_identifier	Name of the foreign data wrapper.
foreign_server_type	information_schema.character_data	Type of the foreign server.
foreign_server_version	information_schema.character_data	Version of the foreign server.
authorization_identifier	information_schema.sql_identifier	Role of the owner of the foreign server.

8.1.3 INFORMATION_SCHEMA._PG_FOREIGN_TABLE_COLUMNS

Displays column information about a foreign table, as described in **Table 8-5**. Only the SYSADMIN user has the permission to view this view.

Table 8-5 INFORMATION_SCHEMA._PG_FOREIGN_TABLE_COLUMNS columns

Name	Type	Description
nspname	name	Schema name.
relname	name	Table name.
attname	name	Column name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
attfdwoptions	text[]	Attribute-level foreign data wrapper options, expressed in a string in the format of *keyword*=*value*.

8.1.4 INFORMATION_SCHEMA._PG_FOREIGN_TABLES

Stores information about all foreign tables defined in the current database, as described in **Table 8-6**, whereas displays information about foreign tables accessible to the current user. Only the SYSADMIN user has the permission to view this view.

Table 8-6 INFORMATION_SCHEMA._PG_FOREIGN_TABLES columns

Name	Type	Description
foreign_table_catalog	information_schema.sql_identifier	Name of the database where the foreign table is located (always the current database).
foreign_table_schema	name	Name of the schema that the foreign table is in.
foreign_table_name	name	Name of the foreign table.
ftoptions	text[]	Foreign table options.
foreign_server_catalog	information_schema.sql_identifier	Name of the database where the foreign server is located (always the current database).
foreign_server_name	information_schema.sql_identifier	Name of the foreign server.
authorization_identifier	information_schema.sql_identifier	Role of the owner.

8.1.5 INFORMATION_SCHEMA._PG_USER_MAPPINGS

Stores mappings from local users to remote users, as described in **Table 8-7**. Only the SYSADMIN user has the permission to view this view.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-7 INFORMATION_SCHEMA._PG_USER_MAPPINGS columns

Name	Type	Description
oid	oid	OID of the mapping from the local user to a remote user.
umoptions	text[]	User mapping specific options, expressed in a string in the format of *keyword*=*value*.
umuser	oid	OID of the local user being mapped (**0** if the user mapping is public).
authorization_identifier	information_schema.sql_identifier	Role of the local user.
foreign_server_catalog	information_schema.sql_identifier	Name of the database where the foreign server is located (always the current database).
foreign_server_name	information_schema.sql_identifier	Name of the foreign server.
srvowner	information_schema.sql_identifier	Owner of the foreign server.

8.1.6 INFORMATION_SCHEMA.INFORMATION_SCHEMA_CATALOG_NAME

Displays the name of the current database, as described in **Table 8-8**.

Table 8-8 INFORMATION_SCHEMA.INFORMATION_SCHEMA_CATALOG_NAME columns

Name	Type	Description
catalog_name	information_schema.sql_identifier	Current database name.

8.1.7 INFORMATION_SCHEMA.TRIGGERS

Displays all triggers defined on the tables and views in the current database. Only the triggers owned by the current user or those on which the current user has permissions other than SELECT are displayed, as shown in **Table 8-9**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-9 INFORMATION_SCHEMA.TRIGGERS columns

Name	Type	Description
trigger_catalog	information_schema.sql_identifier	Name of the database that contains the trigger.
trigger_schema	information_schema.sql_identifier	Name of the schema that contains the trigger.
trigger_name	information_schema.sql_identifier	Trigger name.
event_manipulation	information_schema.character_data	Type of the event that triggers the trigger. The value can be **INSERT**, **UPDATE**, or **DELETE**.
event_object_catalog	information_schema.sql_identifier	Name of the database that contains the table where the trigger is located.
event_object_schema	information_schema.sql_identifier	Name of the schema that contains the table where the trigger is located.
event_object_table	information_schema.sql_identifier	Name of the table where the trigger is located.
action_order	information_schema.cardinal_number	Trigger sequence of triggers with the same **event_manipulation**, **action_timing** and **action_orientation** in the same table.
action_condition	information_schema.character_data	WHEN condition of a trigger. By default, this column is left blank.
action_statement	information_schema.character_data	Statement executed by the trigger.
action_orientation	information_schema.character_data	Specifies whether the trigger fires once for each row being processed or once for each statement.
action_timing	information_schema.character_data	Specifies when the trigger fires. The value can be **BEFORE**, **AFTER**, or **INSTEAD OF**.
action_reference_old_table	information_schema.sql_identifier	This parameter is meaningless and is left blank.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
action_reference_new_table	information_schema.sql_identifier	This parameter is meaningless and is left blank.
action_reference_old_row	information_schema.sql_identifier	This parameter is meaningless and is left blank.
action_reference_new_row	information_schema.sql_identifier	This parameter is meaningless and is left blank.
created	information_schema.time_stamp	This parameter is meaningless and is left blank.

NOTE

INFORMATION_SCHEMA.TRIGGERS displays only triggers whose event types are INSERT, UPDATE, and DELETE. Triggers whose event type is TRUNCATE are not displayed.

8.2 DBE_PERF Schema

In the DBE_PERF schema, views are used to diagnose performance issues and are also the data source of WDR snapshots. After a database is installed, only the initial user and users with the MONADMIN permission have permissions to view views and functions in this schema by default. To ensure compatibility with earlier versions, the permission on the dbe_perf schema has no change before and after an upgrade. In the current version, all users are not allowed to create operators in this schema. Existing operators are not affected. Organization views are divided based on multiple dimensions, such as OS, instance, and memory. These views comply with the following naming rules:

- A view starting with **GLOBAL_** requests data from a CN or DN and returns the data without processing the data.
- A view starting with **SUMMARY_** summarizes data in a cluster. In most cases, data from a CN or DN (sometimes only a CN) is processed, aggregated, and returned.
- A view that does not start with **GLOBAL_** or **SUMMARY_** is a local view and does not request data from any other CN or DN.

8.2.1 OS

8.2.1.1 DBE_PERF.OS_RUNTIME

Displays the running status of the current OS, as described in **Table 8-10**.

Table 8-10 DBE_PERF.OS_RUNTIME columns

Name	Type	Description
id	integer	ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
name	text	Name of the OS running status.
value	numeric	Value of the OS running status.
comments	text	Remarks of the OS running status.
cumulative	Boolean	Specifies whether the value of the OS running status is cumulative.

8.2.1.2 DBE_PERF.GLOBAL_OS_RUNTIME

Provides OS running status information about all normal nodes in the entire cluster, as described in **Table 8-11**.

Table 8-11 DBE_PERF.GLOBAL_OS_RUNTIME columns

Name	Type	Description
node_name	name	Node name.
id	integer	ID.
name	text	Name of the OS running status.
value	numeric	Value of the OS running status.
comments	text	Remarks of the OS running status.
cumulative	Boolean	Specifies whether the value of the OS running status is cumulative.

8.2.1.3 DBE_PERF.OS_THREADS

Provides status information about all threads on the current node, as described in **Table 8-12**.

Table 8-12 DBE_PERF.OS_THREADS columns

Name	Type	Description
node_name	text	Current node name.
pid	bigint	ID of the thread running within the current node process.
lwpid	integer	Lightweight thread ID corresponding to the PID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
thread_name	text	Thread name corresponding to the PID.
creation_time	timestamp with time zone	Thread creation time corresponding to the PID.
dbid	oid	ID of the database to which the thread belongs.

8.2.1.4 DBE_PERF.GLOBAL_OS_THREADS

Provides thread status for all normal nodes in the entire cluster, as outlined in **Table 8-13**.

Table 8-13 DBE_PERF.GLOBAL_OS_THREADS columns

Name	Type	Description
node_name	text	Node name.
pid	bigint	ID of the thread running within the current node process.
lwpid	integer	Lightweight thread ID corresponding to the PID.
thread_name	text	Thread name corresponding to the PID.
creation_time	timestamp with time zone	Thread creation time corresponding to the PID.
dbid	oid	ID of the database to which the thread belongs.

8.2.1.5 DBE_PERF.PERF_QUERY

Provides the name, tree structure, and percentage of the stack information collected on the current node, as described in **Table 8-14**. The MONADMIN permission is required.

Before using this view to query stack information, you need to execute the gs_perf_start function to collect flame graph data.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-14 DBE_PERF.PERF_QUERY columns

Name	Type	Description
backtrace	text	Text of the stack information tree structure.
overhead	double precision	Percentage of the time occupied by the current stack information in the entire stack collection process.

8.2.1.6 DBE_PERF.SHAKING_COLLECT_STATUS

If this view is queried on a CN, the performance jitter monitoring status of all nodes is returned. If this view is queried on a DN, the performance jitter status of the current instance is returned. The MONADMIN permission is required. **Table 8-15** displays the monitoring status. When some shards are upgraded, this view is unavailable.

Table 8-15 Return values of DBE_PERF.SHAKING_COLLECT_STATUS

Name	Type	Description
nodename	text	Node name.
status	text	Specifies whether performance jitter monitoring has been enabled on the host where the node is located. • **NOT IN SHAKE MODE**: The monitoring is not enabled. • **IN SHAKE MODE**: The monitoring is enabled. • **NO PERF SERVICE**: The collection process does not respond. • Unexpected Result: get_shaking_connect_status error. **Detail**: Error details, indicating the query fails.

8.2.2 Instance

8.2.2.1 DBE_PERF.INSTANCE_TIME

Provides various time consumption information under the current cluster node, as described in **Table 8-16**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-16 DBE_PERF.INSTANCE_TIME columns

Name	Type	Description
stat_id	integer	Statistics ID.
stat_name	text	Type name. • DB_TIME: effective time spent by jobs in multi-core scenarios. • CPU_TIME: CPU time cost. • EXECUTION_TIME: time spent in the executor. • PARSE_TIME: time spent on parsing SQL statements. • PLAN_TIME: time spent on generating plans. • REWRITE_TIME: time spent on rewriting SQL statements. • PL_EXECUTION_TIME: execution time of the PL/SQL stored procedure. • PL_COMPILATION_TIME: compilation time of the PL/SQL stored procedure. • NET_SEND_TIME: time spent on the network. • DATA_IO_TIME: time spent on I/Os.
value	bigint	Time value (unit: μs).

8.2.2.2 DBE_PERF.GLOBAL_INSTANCE_TIME

Provides various time consumption information under all normal nodes in the entire cluster, as described in **Table 8-17**.

Table 8-17 DBE_PERF.GLOBAL_INSTANCE_TIME columns

Name	Type	Description
node_name	name	Node name.
stat_id	integer	Statistics ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
stat_name	text	Type name. • DB_TIME: effective time spent by jobs in multi-core scenarios. • CPU_TIME: CPU time cost. • EXECUTION_TIME: time spent in the executor. • PARSE_TIME: time spent on parsing SQL statements. • PLAN_TIME: time spent on generating plans. • REWRITE_TIME: time spent on rewriting SQL statements. • PL_EXECUTION_TIME: execution time of the PL/SQL stored procedure. • PL_COMPILATION_TIME: compilation time of the PL/SQL stored procedure. • NET_SEND_TIME: time spent on the network. • DATA_IO_TIME: time spent on I/Os.
value	bigint	Time value (unit: μs).

8.2.3 Memory

8.2.3.1 DBE_PERF.MEMORY_NODE_DETAIL

Displays memory usage of a node in the database, as shown in **Table 8-18**.

Table 8-18 DBE_PERF.MEMORY_NODE_DETAIL columns

Name	Type	Description
nodename	text	Node name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
memorytype	text	Memory name. • **max_process_memory**: maximum available memory of the database node. • **process_used_memory**: memory occupied by a process. • **max_dynamic_memory**: maximum dynamic memory. • **dynamic_used_memory**: used dynamic memory. • **dynamic_peak_memory**: dynamic peak value of the memory. • **dynamic_used_shrctx**: context of the used dynamic shared memory. • **dynamic_peak_shrctx**: dynamic peak value of the shared memory context. • **max_shared_memory**: maximum shared memory. • **shared_used_memory**: used shared memory. • **high_backend_thread_memory**: memory used by high-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **high_backend_thread_max_memory**: memory reserved for high-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **medium_backend_thread_memory**: memory used by medium-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **medium_backend_thread_max_memory**: memory reserved for medium-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **low_backend_thread_memory**: memory used by low-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **low_backend_thread_max_memory**: memory reserved for low-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **max_sctpcomm_memory**: maximum memory allowed for TCP proxy communication. • **sctpcomm_used_memory**: used memory for TCP proxy communication. • **sctpcomm_peak_memory**: memory peak of TCP proxy communication.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
		• **other_used_memory**: other used memory. • **gpu_max_dynamic_memory**: maximum dynamic GPU memory. • **gpu_dynamic_used_memory**: used dynamic GPU memory. • **gpu_dynamic_peak_memory**: dynamic peak GPU memory. • **pooler_conn_memory**: memory allocated to the connection pool. • **pooler_freeconn_memory**: memory occupied by idle connections in the connection pool. • **storage_compress_memory**: memory used by the storage module for compression. • **udf_reserved_memory**: reserved memory for the UDF.
memorymbytes	integer	Size of the used memory in the unit of MB.

8.2.3.2 DBE_PERF.GLOBAL_MEMORY_NODE_DETAIL

Displays the memory usage on all normal nodes in the cluster, as described in **Table 8-19**.

Table 8-19 DBE_PERF.GLOBAL_MEMORY_NODE_DETAIL columns

Name	Type	Description
nodename	text	Node name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
memorytype	text	Memory name. • **max_process_memory**: maximum available memory of the database node. • **process_used_memory**: memory occupied by a process. • **max_dynamic_memory**: maximum dynamic memory. • **dynamic_used_memory**: used dynamic memory. • **dynamic_peak_memory**: dynamic peak value of the memory. • **dynamic_used_shrctx**: context of the used dynamic shared memory. • **dynamic_peak_shrctx**: dynamic peak value of the shared memory context. • **max_shared_memory**: maximum shared memory. • **shared_used_memory**: used shared memory. • **high_backend_thread_memory**: memory used by high-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **high_backend_thread_max_memory**: memory reserved for high-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **medium_backend_thread_memory**: memory used by medium-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **medium_backend_thread_max_memory**: memory reserved for medium-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **low_backend_thread_memory**: memory used by low-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **low_backend_thread_max_memory**: memory reserved for low-priority threads. This is valid only when **enable_threads_memory_reserve** is set to **on**. • **max_sctpcomm_memory**: maximum memory allowed for TCP proxy communication. • **sctpcomm_used_memory**: used memory for TCP proxy communication. • **sctpcomm_peak_memory**: memory peak of TCP proxy communication.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
		• **other_used_memory**: other used memory. • **gpu_max_dynamic_memory**: maximum dynamic GPU memory. • **gpu_dynamic_used_memory**: used dynamic GPU memory. • **gpu_dynamic_peak_memory**: dynamic peak GPU memory. • **pooler_conn_memory**: memory allocated to the connection pool. • **pooler_freeconn_memory**: memory occupied by idle connections in the connection pool. • **storage_compress_memory**: memory used by the storage module for compression. • **udf_reserved_memory**: reserved memory for the UDF.
memorymbytes	integer	Size of the used memory in the unit of MB.

8.2.3.3 DBE_PERF.MEMORY_NODE_NG_DETAIL

Table 8-20 describes the memory usage of a node group.

Table 8-20 DBE_PERF.MEMORY_NODE_NG_DETAIL columns

Name	Type	Description
ngname	text	Node group name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
memorytype	text	Memory name. • **ng_total_memory**: total memory configured in the node group. • **ng_used_memory**: used memory. • **ng_estimate_memory**: memory used by the optimizer for evaluation. • **ng_foreignrp_memsize**: memory configured in the foreign resource pool. • **ng_foreignrp_usedsize**: memory used by the foreign resource pool. • **ng_foreignrp_peaksize**: peak memory used by the foreign resource pool. • **ng_foreignrp_mempct**: percentage of the system memory configured in attributes of the foreign resource pool. • **ng_foreignrp_estmsize**: memory used by the optimizer for job evaluation in the foreign resource pool.
memorymbytes	integer	Size of the used memory in the unit of MB.

8.2.3.4 DBE_PERF.SHARED_MEMORY_DETAIL

Displays the usage information about shared memory contexts on the current node, as shown in **Table 8-21**.

Table 8-21 DBE_PERF.SHARED_MEMORY_DETAIL columns

Name	Type	Description
contextname	text	Name of the memory context.
level	smallint	Level of the memory context.
parent	text	Name of the parent memory context.
totalsize	bigint	Total size of the shared memory (unit: byte).
freesize	bigint	Remaining size of the shared memory (unit: byte).
usedsize	bigint	Used size of the shared memory (unit: byte).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.3.5 DBE_PERF.GLOBAL_SHARED_MEMORY_DETAIL

Displays the usage information about shared memory contexts on all normal nodes in the entire cluster, as described in **Table 8-22**.

Table 8-22 DBE_PERF.GLOBAL_SHARED_MEMORY_DETAIL columns

Name	Type	Description
node_name	name	Node name.
contextname	text	Name of the memory context.
level	smallint	Level of the memory context.
parent	text	Name of the parent memory context.
totalsize	bigint	Total size of the shared memory (unit: byte).
freesize	bigint	Remaining size of the shared memory (unit: byte).
usedsize	bigint	Used size of the shared memory (unit: byte).

8.2.3.6 DBE_PERF.TRACK_MEMORY_CONTEXT_DETAIL

Queries the detailed memory allocation information about the memory context set by DBE_PERF.TRACK_MEMORY_CONTEXT, as described in **Table 8-23**. Only the initial user or a user with the MONADMIN permission can query this view.

Table 8-23 DBE_PERF.TRACK_MEMORY_CONTEXT_DETAIL columns

Name	Type	Description
context_name	text	Name of the memory context.
file	text	File to which the memory allocation location belongs.
line	integer	Line number of the memory allocation location.
size	bigint	Total size of memory that is applied for (unit: byte).

8.2.4 File

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.4.1 DBE_PERF.FILE_IOSTAT

Records statistics about data file I/Os to indicate I/O performance and detect performance problems such as abnormal I/O operations, as described in **Table 8-24**.

Table 8-24 DBE_PERF.FILE_IOSTAT columns

Name	Type	Description
filenum	oid	File ID.
dbid	oid	Database ID.
spcid	oid	Tablespace ID.
phyrds	bigint	Number of times of reading physical files.
phywrts	bigint	Number of times of writing into physical files.
phyblkrd	bigint	Number of times of reading physical file blocks.
phyblkwrt	bigint	Number of times of writing into physical file blocks.
readtim	bigint	Total duration of reading (unit: μs).
writetim	bigint	Total duration of writing (unit: μs).
avgiotim	bigint	Average duration of reading and writing (unit: μs).
lstiotim	bigint	Duration of the last file reading (unit: μs).
miniotim	bigint	Minimum duration of reading and writing (unit: μs).
maxiowtm	bigint	Maximum duration of reading and writing (unit: μs).

8.2.4.2 DBE_PERF.SUMMARY_FILE_IOSTAT

Records the summary of statistics about data file I/Os of each node in the cluster to reflect performance issues such as exceptions in I/O operations, as described in **Table 8-25**.

The values of **phyrds**, **phywrts**, **phyblkrd**, **phyblkwrt**, **readtim** and **writetim** are summed up based on the data of each node. **avgiotim** is the average value (total duration/total times) of each node. **lstiotim** and **maxiowtm** are the maximum values of each node. **miniotim** is the minimum value of each node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-25 DBE_PERF.SUMMARY_FILE_IOSTAT columns

Name	Type	Description
filenum	oid	File ID.
dbid	oid	Database ID.
spcid	oid	Tablespace ID.
phyrds	numeric	Number of times of reading physical files.
phywrts	numeric	Number of times of writing into physical files.
phyblkrd	numeric	Number of times of reading physical file blocks.
phyblkwrt	numeric	Number of times of writing into physical file blocks.
readtim	numeric	Total duration of reading (unit: μs).
writetim	numeric	Total duration of writing (unit: μs).
avgiotim	bigint	Average duration of reading and writing (unit: μs).
lstiotim	bigint	Duration of the last file reading (unit: μs).
miniotim	bigint	Minimum duration of reading and writing (unit: μs).
maxiowtm	bigint	Maximum duration of reading and writing (unit: μs).

8.2.4.3 DBE_PERF.GLOBAL_FILE_IOSTAT

Displays statistics about data file I/Os on all nodes in the cluster, as described in **Table 8-26**.

Table 8-26 DBE_PERF.GLOBAL_FILE_IOSTAT columns

Name	Type	Description
node_name	name	Node name.
filenum	oid	File ID.
dbid	oid	Database ID.
spcid	oid	Tablespace ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
phyrds	bigint	Number of times of reading physical files.
phywrts	bigint	Number of times of writing into physical files.
phyblkrd	bigint	Number of times of reading physical file blocks.
phyblkwrt	bigint	Number of times of writing into physical file blocks.
readtim	bigint	Total duration of reading (unit: μs).
writetim	bigint	Total duration of writing (unit: μs).
avgiotim	bigint	Average duration of reading and writing (unit: μs).
lstiotim	bigint	Duration of the last file reading (unit: μs).
miniotim	bigint	Minimum duration of reading and writing (unit: μs).
maxiowtm	bigint	Maximum duration of reading and writing (unit: μs).

8.2.4.4 DBE_PERF.FILE_REDO_IOSTAT

Records statistics about redo logs (WALs) on the current node, as described in **Table 8-27**.

Table 8-27 DBE_PERF.FILE_REDO_IOSTAT columns

Name	Type	Description
phywrts	bigint	Number of times writing into the WAL buffer.
phyblkwrt	bigint	Number of blocks written into the WAL buffer.
writetim	bigint	Duration of writing into Xlog files (unit: μs).
avgiotim	bigint	Average duration of writing into Xlog files (unit: μs). The value is **writetim** divided by **phywrts**.
lstiotim	bigint	Duration of the last writing into Xlog files (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
miniotim	bigint	Minimum duration of writing into Xlog files (unit: μs).
maxiowtm	bigint	Maximum duration of writing into Xlog files (unit: μs).

8.2.4.5 DBE_PERF.SUMMARY_FILE_REDO_IOSTAT

Records redo log (WAL) statistics of all nodes in the cluster, as described in **Table 8-28**. The values of the **phywrts**, **phyblkwrt**, and **writetim** columns are accumulated based on the data of each node. The value of **avgiotim** is the average value of each node (summed up **writetim** or **phywrts**). The values of **lstiotim** and **maxiowtm** are the maximum values of each node, and the value of **miniotim** is the minimum value of each node.

Table 8-28 DBE_PERF.SUMMARY_FILE_REDO_IOSTAT columns

Name	Type	Description
phywrts	numeric	Number of times writing into the WAL buffer.
phyblkwrt	numeric	Number of blocks written into the WAL buffer.
writetim	numeric	Duration of writing into Xlog files (unit: μs).
avgiotim	bigint	Average duration of writing into Xlog files (unit: μs). The value is **writetim** divided by **phywrts**.
lstiotim	bigint	Duration of the last writing into Xlog files (unit: μs).
miniotim	bigint	Minimum duration of writing into Xlog files (unit: μs).
maxiowtm	bigint	Maximum duration of writing into Xlog files (unit: μs).

8.2.4.6 DBE_PERF.GLOBAL_FILE_REDO_IOSTAT

Displays statistics about redo logs (WALs) on nodes in the cluster, as described in **Table 8-29**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-29 DBE_PERF.GLOBAL_FILE_REDO_IOSTAT columns

Name	Type	Description
node_name	name	Node name.
phywrts	bigint	Number of times writing into the WAL buffer.
phyblkwrt	bigint	Number of blocks written into the WAL buffer.
writetim	bigint	Duration of writing into Xlog files (unit: μs).
avgiotim	bigint	Average duration of writing into Xlog files (unit: μs). The value is **writetim** divided by **phywrts**.
lstiotim	bigint	Duration of the last writing into Xlog files (unit: μs)
miniotim	bigint	Minimum duration of writing into Xlog files (unit: μs).
maxiowtm	bigint	Maximum duration of writing into Xlog files (unit: μs).

8.2.4.7 DBE_PERF.LOCAL_REL_IOSTAT

Displays the accumulated I/O status of all data files on the current node, as described in **Table 8-30**.

Table 8-30 DBE_PERF.LOCAL_REL_IOSTAT columns

Name	Type	Description
phyrds	bigint	Number of times of reading physical files.
phywrts	bigint	Number of times of writing into physical files.
phyblkrd	bigint	Number of times of reading physical file blocks.
phyblkwrt	bigint	Number of times of writing into physical file blocks.
dbid	oid	In the current version, the value is always **0**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.4.8 DBE_PERF.SUMMARY_REL_IOSTAT

Obtains statistics about data file I/Os on all nodes, as described in **Table 8-31**.

Table 8-31 DBE_PERF.SUMMARY_REL_IOSTAT columns

Name	Type	Description
phyrds	numeric	Number of times of reading physical files.
phywrts	numeric	Number of times of writing into physical files.
phyblkrd	numeric	Number of times of reading physical file blocks.
phyblkwrt	numeric	Number of times of writing into physical file blocks.
dbid	oid	In the current version, the value is always **0**.

8.2.4.9 DBE_PERF.GLOBAL_REL_IOSTAT

Obtains statistics about data file I/Os on all nodes in the cluster, as described in **Table 8-32**.

Table 8-32 DBE_PERF.GLOBAL_REL_IOSTAT columns

Name	Type	Description
node_name	name	Node name.
phyrds	bigint	Number of times of reading physical files.
phywrts	bigint	Number of times of writing into physical files.
phyblkrd	bigint	Number of times of reading physical file blocks.
phyblkwrt	bigint	Number of times of writing into physical file blocks.
dbid	oid	In the current version, the value is always **0**.

8.2.5 Object

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.5.1 DBE_PERF.STAT_USER_TABLES

Displays the status information about user-defined ordinary tables in all schemas on the current node, as described in **Table 8-33**.

Table 8-33 DBE_PERF.STAT_USER_TABLES columns

Name	Type	Description
relid	oid	OID of a table.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).
n_live_tup	bigint	Estimated number of live rows.
n_dead_tup	bigint	Estimated number of inactive rows. It indicates the number of inactive row pointers in Ustore tables.
last_vacuum	timestamp with time zone	Last time at which the table was manually vacuumed (excluding VACUUM FULL).
last_autovacuum	timestamp with time zone	Last time when the table was vacuumed by the autovacuum daemon thread.
last_analyze	timestamp with time zone	Last time at which the table was manually analyzed.
last_autoanalyze	timestamp with time zone	Last time when the table was analyzed by the autovacuum daemon thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
vacuum_count	bigint	Number of times this table is manually vacuumed (excluding VACUUM FULL).
autovacuum_count	bigint	Number of times that the table is vacuumed by the autovacuum daemon thread.
analyze_count	bigint	Number of times that the table has been manually analyzed.
autoanalyze_count	bigint	Number of times that the table has been analyzed by the autovacuum daemon thread.
last_updated	timestamp with time zone	Time when the monitoring data of the object in the view is updated for the last time.

8.2.5.2 DBE_PERF.SUMMARY_STAT_USER_TABLES

Displays the sum of the status information about user-defined ordinary tables in all schemas on each node of a cluster. It is used on the CN. The status information about user-defined ordinary tables on each DN is summed up. The timestamp column is not summed up and only the latest value of this column on all nodes is used, as described in **Table 8-34**.

Table 8-34 DBE_PERF.SUMMARY_STAT_USER_TABLES

Name	Type	Description
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	numeric	Number of sequential scans initiated on the table.
seq_tup_read	numeric	Number of live rows fetched by sequential scans.
idx_scan	numeric	Number of index scans initiated on the table.
idx_tup_fetch	numeric	Number of live rows fetched by index scans.
n_tup_ins	numeric	Number of rows inserted.
n_tup_upd	numeric	Number of rows updated.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
n_tup_del	numeric	Number of rows deleted.
n_tup_hot_upd	numeric	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).
n_live_tup	numeric	Estimated number of live rows.
n_dead_tup	numeric	Estimated number of inactive rows. It indicates the number of inactive row pointers in Ustore tables.
last_vacuum	timestamp with time zone	Last time at which the table was manually vacuumed (excluding VACUUM FULL).
last_autovacuum	timestamp with time zone	Last time when the table was vacuumed by the autovacuum daemon thread.
last_analyze	timestamp with time zone	Last time at which the table was manually analyzed.
last_autoanalyze	timestamp with time zone	Last time when the table was analyzed by the autovacuum daemon thread.
vacuum_count	numeric	Number of times this table is manually vacuumed (excluding VACUUM FULL).
autovacuum_count	numeric	Number of times that the table is vacuumed by the autovacuum daemon thread.
analyze_count	numeric	Number of times that the table has been manually analyzed.
autoanalyze_count	numeric	Number of times that the table has been analyzed by the autovacuum daemon thread.

8.2.5.3 DBE_PERF.GLOBAL_STAT_USER_TABLES

Displays the status of user-defined ordinary tables in all schemas on each node in cluster. It is used on CNs. The status information about user-defined ordinary tables on each DN is displayed but not summed up, as described in **Table 8-35**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-35 DBE_PERF.GLOBAL_STAT_USER_TABLES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of a table.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).
n_live_tup	bigint	Estimated number of live rows.
n_dead_tup	bigint	Estimated number of inactive rows. It indicates the number of inactive row pointers in Ustore tables.
last_vacuum	timestamp with time zone	Last time at which the table was manually vacuumed (excluding VACUUM FULL).
last_autovacuum	timestamp with time zone	Last time when the table was vacuumed by the autovacuum daemon thread.
last_analyze	timestamp with time zone	Last time at which the table was manually analyzed.
last_autoanalyze	timestamp with time zone	Last time when the table was analyzed by the autovacuum daemon thread.
vacuum_count	bigint	Number of times this table is manually vacuumed (excluding VACUUM FULL).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
autovacuum_count	bigint	Number of times that the table is vacuumed by the autovacuum daemon thread.
analyze_count	bigint	Number of times that the table has been manually analyzed.
autoanalyze_count	bigint	Number of times that the table has been analyzed by the autovacuum daemon thread.

8.2.5.4 DBE_PERF.STAT_USER_INDEXES

Displays the index status information about the user-defined ordinary tables on the current node, as described in **Table 8-36**.

Table 8-36 DBE_PERF.STAT_USER_INDEXES columns

Name	Type	Description
relid	oid	OID of the table for this index.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema where the index is located.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_scan	bigint	Number of index scans performed on the index.
idx_tup_read	bigint	Number of index entries returned by scans on the index.
idx_tup_fetch	bigint	Number of live rows fetched in the original table by a simple index scan that uses the index.
last_updated	timestamp with time zone	Time when the monitoring data of the object in the view is updated for the last time.

8.2.5.5 DBE_PERF.SUMMARY_STAT_USER_INDEXES

Displays the sum of the index status of user-defined ordinary tables in all schemas on each node of a cluster. It applies to CNs only and the index status information about user-defined ordinary tables on each CN is summed up, as described in **Table 8-37**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-37 DBE_PERF.SUMMARY_STAT_USER_INDEXES columns

Name	Type	Description
schemaname	name	Name of the schema where the index is located.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_scan	numeric	Number of index scans performed on the index.
idx_tup_read	numeric	Number of index entries returned by scans on the index.
idx_tup_fetch	numeric	Number of live rows fetched in the original table by a simple index scan that uses the index.

8.2.5.6 DBE_PERF.GLOBAL_STAT_USER_INDEXES

Displays the index status information about user-defined ordinary tables in all schemas on each node of the cluster. The information is about CNs and DNs. It is used on CNs and information is not summed up, as described in **Table 8-38**.

Table 8-38 DBE_PERF.GLOBAL_STAT_USER_INDEXES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of the table for the index.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema where the index is located.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_scan	bigint	Number of index scans performed on the index.
idx_tup_read	bigint	Number of index entries returned by scans on the index.
idx_tup_fetch	bigint	Number of live rows fetched in the original table by a simple index scan that uses the index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.5.7 DBE_PERF.STAT_SYS_TABLES

Displays status information about all the system catalogs in the pg_catalog, information_schema, and pg_toast schemas on the current node, as described in Table 8-39.

Table 8-39 DBE_PERF.STAT_SYS_TABLES columns

Name	Type	Description
relid	oid	OID of a table.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).
n_live_tup	bigint	Estimated number of live rows.
n_dead_tup	bigint	Estimated number of inactive rows. It indicates the number of inactive row pointers in Ustore tables.
last_vacuum	timestamp with time zone	Last time at which the table was manually vacuumed (excluding VACUUM FULL).
last_autovacuum	timestamp with time zone	Last time when the table was vacuumed by the autovacuum daemon thread.
last_analyze	timestamp with time zone	Last time at which the table was manually analyzed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
last_autoanalyze	timestamp with time zone	Last time when the table was analyzed by the autovacuum daemon thread.
vacuum_count	bigint	Number of times the table has been manually vacuumed (not counting VACUUM FULL).
autovacuum_count	bigint	Number of times that the table is vacuumed by the autovacuum daemon thread.
analyze_count	bigint	Number of times that the table has been manually analyzed.
autoanalyze_count	bigint	Number of times that the table has been analyzed by the autovacuum daemon thread.
last_updated	timestamp with time zone	Time when the monitoring data of the object in the view is updated for the last time.

8.2.5.8 DBE_PERF.SUMMARY_STAT_SYS_TABLES

Displays the sum of status of all system catalogs in the pg_catalog, information_schema, and pg_toast schemas on each node in the cluster. The statistics are used on CNs and include data on CNs and DNs. The status information in the system catalog on each node is summed up. The timestamp column is not summed up and only the latest value of this column on all nodes is used, as described in **Table 8-40**.

Table 8-40 DBE_PERF.SUMMARY_STAT_SYS_TABLES columns

Name	Type	Description
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	numeric	Number of sequential scans initiated on the table.
seq_tup_read	numeric	Number of live rows fetched by sequential scans.
idx_scan	numeric	Number of index scans initiated on the table.
idx_tup_fetch	numeric	Number of live rows fetched by index scans.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
n_tup_ins	numeric	Number of rows inserted.
n_tup_upd	numeric	Number of rows updated.
n_tup_del	numeric	Number of rows deleted.
n_tup_hot_upd	numeric	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).
n_live_tup	numeric	Estimated number of live rows.
n_dead_tup	numeric	Estimated number of inactive rows. It indicates the number of inactive row pointers in Ustore tables.
last_vacuum	timestamp with time zone	Last time at which the table was manually vacuumed (excluding VACUUM FULL).
last_autovacuum	timestamp with time zone	Last time when the table was vacuumed by the autovacuum daemon thread.
last_analyze	timestamp with time zone	Last time at which the table was manually analyzed.
last_autoanalyze	timestamp with time zone	Last time when the table was analyzed by the autovacuum daemon thread.
vacuum_count	numeric	Number of times this table is manually vacuumed (excluding VACUUM FULL).
autovacuum_count	numeric	Number of times that the table is vacuumed by the autovacuum daemon thread.
analyze_count	numeric	Number of times that the table has been manually analyzed.
autoanalyze_count	numeric	Number of times that the table has been analyzed by the autovacuum daemon thread.

8.2.5.9 DBE_PERF.GLOBAL_STAT_SYS_TABLES

Displays the status information of all system catalogs in pg_catalog, information_schema, and pg_toast modes on each node in the cluster. The status information includes that of CNs and DNs. It is used on CNs and is not summed up, as described in **Table 8-41**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-41 DBE_PERF.GLOBAL_STAT_SYS_TABLES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of a table.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).
n_live_tup	bigint	Estimated number of live rows.
n_dead_tup	bigint	Estimated number of inactive rows. It indicates the number of inactive row pointers in Ustore tables.
last_vacuum	timestamp with time zone	Last time at which the table was manually vacuumed (excluding VACUUM FULL).
last_autovacuum	timestamp with time zone	Last time when the table was vacuumed by the autovacuum daemon thread.
last_analyze	timestamp with time zone	Last time at which the table was manually analyzed.
last_autoanalyze	timestamp with time zone	Last time when the table was analyzed by the autovacuum daemon thread.
vacuum_count	bigint	Number of times this table is manually vacuumed (excluding VACUUM FULL).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
autovacuum_count	bigint	Number of times that the table is vacuumed by the autovacuum daemon thread.
analyze_count	bigint	Number of times that the table has been manually analyzed.
autoanalyze_count	bigint	Number of times that the table has been analyzed by the autovacuum daemon thread.

8.2.5.10 DBE_PERF.STAT_SYS_INDEXES

Displays index status information about all the system catalogs in the pg_catalog, information_schema, and pg_toast schemas, as described in **Table 8-42**.

Table 8-42 DBE_PERF.STAT_SYS_INDEXES columns

Name	Type	Description
relid	oid	OID of the table for the index.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema where the index is located.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_scan	bigint	Number of index scans performed on the index.
idx_tup_read	bigint	Number of index entries returned by scans on the index.
idx_tup_fetch	bigint	Number of live rows fetched in the original table by a simple index scan that uses the index.
last_updated	timestamp with time zone	Time when the monitoring data of the object in the view is updated for the last time.

8.2.5.11 DBE_PERF.SUMMARY_STAT_SYS_INDEXES

Displays the sum of index status statistics of all system catalogs in the pg_catalog, information_schema, and pg_toast schemas on each node in the cluster. It is used on CNs and the index status information of system catalogs on each node is summed up, as described in **Table 8-43**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-43 DBE_PERF.SUMMARY_STAT_SYS_INDEXES columns

Name	Type	Description
schemaname	name	Name of the schema where the index is located.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_scan	numeric	Number of index scans performed on the index.
idx_tup_read	numeric	Number of index entries returned by scans on the index.
idx_tup_fetch	numeric	Number of live rows fetched in the original table by a simple index scan that uses the index.

8.2.5.12 DBE_PERF.GLOBAL_STAT_SYS_INDEXES

Displays the index status information of all system catalogs in pg_catalog, information_schema, and pg_toast schemas on each node in the cluster. The index status information is about CNs and DNs. It is used on CNs and information is not summed up, as described in **Table 8-44**.

Table 8-44 DBE_PERF.GLOBAL_STAT_SYS_INDEXES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of the table for the index.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema where the index is located.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_scan	bigint	Number of index scans performed on the index.
idx_tup_read	bigint	Number of index entries returned by scans on the index.
idx_tup_fetch	bigint	Number of live rows fetched in the original table by a simple index scan that uses the index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.5.13 DBE_PERF.STAT_ALL_TABLES

Displays the status of each table (including the TOAST table) of the current node in the database, as described in **Table 8-45**.

Table 8-45 DBE_PERF.STAT_ALL_TABLES columns

Name	Type	Description
relid	oid	OID of a table.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).
n_live_tup	bigint	Estimated number of live rows.
n_dead_tup	bigint	Estimated number of inactive rows. It indicates the number of inactive row pointers in Ustore tables.
last_vacuum	timestamp with time zone	Last time at which the table was manually vacuumed (excluding VACUUM FULL).
last_autovacuum	timestamp with time zone	Last time when the table was vacuumed by the autovacuum daemon thread.
last_analyze	timestamp with time zone	Last time at which the table was manually analyzed.
last_autoanalyze	timestamp with time zone	Last time when the table was analyzed by the autovacuum daemon thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
vacuum_count	bigint	Number of times this table is manually vacuumed (excluding VACUUM FULL).
autovacuum_count	bigint	Number of times that the table is vacuumed by the autovacuum daemon thread.
analyze_count	bigint	Number of times that the table has been manually analyzed.
autoanalyze_count	bigint	Number of times that the table has been analyzed by the autovacuum daemon thread.
last_updated	timestamp with time zone	Time when the monitoring data of the object in the view is updated for the last time.

8.2.5.14 DBE_PERF.SUMMARY_STAT_ALL_TABLES

Displays the sum of the status information of each table (including the TOAST table) on each node in the cluster. It is used on the CN. The status information in the table on each node is summed up. The timestamp column is not summed up and only the latest value of this column on all nodes is used, as described in **Table 8-46**.

Table 8-46 DBE_PERF.SUMMARY_STAT_ALL_TABLES columns

Name	Type	Description
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	numeric	Number of sequential scans initiated on the table.
seq_tup_read	numeric	Number of live rows fetched by sequential scans.
idx_scan	numeric	Number of index scans initiated on the table.
idx_tup_fetch	numeric	Number of live rows fetched by index scans.
n_tup_ins	numeric	Number of rows inserted.
n_tup_upd	numeric	Number of rows updated.
n_tup_del	numeric	Number of rows deleted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
n_tup_hot_upd	numeric	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).
n_live_tup	numeric	Estimated number of live rows.
n_dead_tup	numeric	Estimated number of inactive rows. It indicates the number of inactive row pointers in Ustore tables.
last_vacuum	timestamp with time zone	Last time at which the table was manually vacuumed (excluding VACUUM FULL).
last_autovacuum	timestamp with time zone	Last time when the table was vacuumed by the autovacuum daemon thread.
last_analyze	timestamp with time zone	Last time at which the table was manually analyzed.
last_autoanalyze	timestamp with time zone	Last time when the table was analyzed by the autovacuum daemon thread.
vacuum_count	numeric	Number of times this table is manually vacuumed (excluding VACUUM FULL).
autovacuum_count	numeric	Number of times this table has been vacuumed by the autovacuum daemon.
analyze_count	numeric	Number of times that the table has been manually analyzed.
autoanalyze_count	numeric	Number of times that the table has been analyzed by the autovacuum daemon thread.
last_updated	timestamp with time zone	Time when the monitoring data of the table in the view is updated for the last time.

8.2.5.15 DBE_PERF.GLOBAL_STAT_ALL_TABLES

Displays the status of each table (including the TOAST table) on each node in the cluster. It is used on CNs and the status of each table on each node is not summed up, as described in **Table 8-47**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-47 DBE_PERF.GLOBAL_STAT_ALL_TABLES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of a table.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).
n_live_tup	bigint	Estimated number of live rows.
n_dead_tup	bigint	Estimated number of inactive rows. It indicates the number of inactive row pointers in Ustore tables.
last_vacuum	timestamp with time zone	Last time at which the table was manually vacuumed (excluding VACUUM FULL).
last_autovacuum	timestamp with time zone	Last time when the table was vacuumed by the autovacuum daemon thread.
last_analyze	timestamp with time zone	Last time at which the table was manually analyzed.
last_autoanalyze	timestamp with time zone	Last time when the table was analyzed by the autovacuum daemon thread.
vacuum_count	bigint	Number of times this table is manually vacuumed (excluding VACUUM FULL).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
autovacuum_count	bigint	Number of times that the table is vacuumed by the autovacuum daemon thread.
analyze_count	bigint	Number of times that the table has been manually analyzed.
autoanalyze_count	bigint	Number of times that the table has been analyzed by the autovacuum daemon thread.
last_updated	timestamp with time zone	Time when the monitoring data of the table in the view is updated for the last time.

8.2.5.16 DBE_PERF.STAT_ALL_INDEXES

Displays the access information about each index of the current node in the database, as described in **Table 8-48**.

Table 8-48 DBE_PERF.STAT_ALL_INDEXES columns

Name	Type	Description
relid	oid	OID of the table for the index.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema where the index is located.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_scan	bigint	Number of index scans performed on the index.
idx_tup_read	bigint	Number of index entries returned by scans on the index.
idx_tup_fetch	bigint	Number of live rows fetched in the original table by a simple index scan that uses the index.
last_updated	timestamp with time zone	Time when the monitoring data of the object in the view is updated for the last time.

8.2.5.17 DBE_PERF.SUMMARY_STAT_ALL_INDEXES

Displays the sum of access statistics of each index on each database node in a cluster. It is used on CNs. The access statistics of each node index are summed up.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

The timestamp column is not summed up, and only the latest value of the column on all nodes is used, as described in **Table 8-49**.

Table 8-49 DBE_PERF.SUMMARY_STAT_ALL_INDEXES columns

Name	Type	Description
schemaname	name	Name of the schema where the index is located.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_scan	numeric	Number of index scans performed on the index.
idx_tup_read	numeric	Number of index entries returned by scans on the index.
idx_tup_fetch	numeric	Number of live rows fetched in the original table by a simple index scan that uses the index.
last_updated	timestamp with time zone	Time when the monitoring data of the index in the view is updated for the last time.

8.2.5.18 DBE_PERF.GLOBAL_STAT_ALL_INDEXES

Displays the access information about each index on each node of the database in the cluster. It is used on the CN. The status information of each index on each node is not summed up, as described in **Table 8-50**.

Table 8-50 DBE_PERF.GLOBAL_STAT_ALL_INDEXES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of the table for the index.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema where the index is located.
relname	name	Name of the table for the index.
indexrelname	name	Index name.
idx_scan	bigint	Number of index scans performed on the index.
idx_tup_read	bigint	Number of index entries returned by scans on the index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
idx_tup_fetch	bigint	Number of live rows fetched in the original table by a simple index scan that uses the index.
last_updated	timestamp with time zone	Time when the monitoring data of the index in the view is updated for the last time.

8.2.5.19 DBE_PERF.STAT_DATABASE

Displays the statistics of the current node in the database, as described in **Table 8-51**.

Table 8-51 DBE_PERF.STAT_DATABASE columns

Name	Type	Description
datid	oid	OID of the database.
datname	name	Database name.
numbackends	integer	Number of backends currently connected to this database.
xact_commit	bigint	Number of transactions in this database that have been committed.
xact_rollback	bigint	Number of transactions in this database that have been rolled back.
blks_read	bigint	Number of disk blocks read in this database.
blks_hit	bigint	Number of disk blocks that have been hit in the cache. In this case, data does not need to be read from disks. (The cache includes only the buffer cache and does not include the file system cache of the OS.)
tup_returned	bigint	Number of live rows fetched by sequential scans and number of index rows returned by index scans in the database.
tup_fetched	bigint	Number of rows returned by the current database through indexes.
tup_inserted	bigint	Number of rows inserted.
tup_updated	bigint	Number of rows updated.
tup_deleted	bigint	Number of rows deleted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
conflicts	bigint	Number of queries canceled due to conflicts with database replay (conflicts occur only on the standby node). For details, see **DBE_PERF.STAT_DATABASE_CONFLICTS**.
temp_files	bigint	Number of temporary files created by queries in this database. All temporary files are counted, which is not affected by the value of the GUC parameter **log_temp_files**.
temp_bytes	bigint	Total amount of data written to temporary files by queries in this database. All temporary files are counted, which is not affected by the value of the GUC parameter **log_temp_files**.
deadlocks	bigint	Number of deadlocks detected in this database.
blk_read_time	double precision	Time spent reading data file blocks by backends in this database (unit: ms).
blk_write_time	double precision	Time spent in writing data file blocks by backends in this database (unit: ms).
stats_reset	timestamp with time zone	Time at which the current statistics were reset.

8.2.5.20 DBE_PERF.SUMMARY_STAT_DATABASE

Sums up node status information in the cluster. (It is used on CNs. The node status information in the database is summed up by database name. However, columns of a timestamp type are not summed up; instead, only their latest values are obtained from all nodes.) **Table 8-52** describe related columns.

Table 8-52 DBE_PERF.SUMMARY_STAT_DATABASE columns

Name	Type	Description
datname	name	Database name.
numbackends	bigint	Number of backends currently connected to this database.
xact_commit	numeric	Number of transactions in this database that have been committed.
xact_rollback	numeric	Number of transactions in this database that have been rolled back.
blks_read	numeric	Number of disk blocks read in this database.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
blks_hit	numeric	Number of disk blocks that have been hit in the cache. In this case, data does not need to be read from disks. (The cache includes only the buffer cache and does not include the file system cache of the OS.)
tup_returned	numeric	Number of live rows fetched by sequential scans and number of index rows returned by index scans in the database.
tup_fetched	numeric	Number of rows returned by the current database through indexes.
tup_inserted	bigint	Number of rows inserted.
tup_updated	bigint	Number of rows updated.
tup_deleted	bigint	Number of rows deleted.
conflicts	bigint	Number of queries canceled due to conflicts with database replay (conflicts occur only on the standby node). For details, see **DBE_PERF.STAT_DATABASE_CONFLICTS**.
temp_files	numeric	Number of temporary files created by queries in this database. All temporary files are counted, which is not affected by the value of the GUC parameter **log_temp_files**.
temp_bytes	numeric	Total amount of data written to temporary files by queries in this database. All temporary files are counted, which is not affected by the value of the GUC parameter **log_temp_files**.
deadlocks	bigint	Number of deadlocks detected in this database.
blk_read_time	double precision	Time spent reading data file blocks by backends in this database (unit: ms).
blk_write_time	double precision	Time spent in writing data file blocks by backends in this database (unit: ms).
stats_reset	timestamp with time zone	Time at which the current statistics were reset.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.5.21 DBE_PERF.GLOBAL_STAT_DATABASE

Displays the statistics of each node in the cluster, as described in **Table 8-53**. (It is used on CNs and the status statistics of each database are not summed up.)

Table 8-53 DBE_PERF.GLOBAL_STAT_DATABASE columns

Name	Type	Description
node_name	name	Node name.
datid	oid	OID of the database.
datname	name	Database name.
numbackends	integer	Number of backends currently connected to this database.
xact_commit	bigint	Number of transactions in this database that have been committed.
xact_rollback	bigint	Number of transactions in this database that have been rolled back.
blks_read	bigint	Number of disk blocks read in this database.
blks_hit	bigint	Number of disk blocks that have been hit in the cache. In this case, data does not need to be read from disks. (The cache includes only the buffer cache and does not include the file system cache of the OS.)
tup_returned	bigint	Number of live rows fetched by sequential scans and number of index rows returned by index scans in the database.
tup_fetched	bigint	Number of rows returned by the current database through indexes.
tup_inserted	bigint	Number of rows inserted.
tup_updated	bigint	Number of rows updated.
tup_deleted	bigint	Number of rows deleted.
conflicts	bigint	Number of queries canceled due to conflicts with database replay (conflicts occur only on the standby node). For details, see **DBE_PERF.STAT_DATABASE_CONFLICTS**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
temp_files	bigint	Number of temporary files created by queries in this database. All temporary files are counted, which is not affected by the value of the GUC parameter **log_temp_files**.
temp_bytes	bigint	Total amount of data written to temporary files by queries in this database. All temporary files are counted, which is not affected by the value of the GUC parameter **log_temp_files**.
deadlocks	bigint	Number of deadlocks detected in this database.
blk_read_time	double precision	Time spent reading data file blocks by backends in this database (unit: ms).
blk_write_time	double precision	Time spent in writing data file blocks by backends in this database (unit: ms).
stats_reset	timestamp with time zone	Time at which the current statistics were reset.

8.2.5.22 DBE_PERF.STAT_DATABASE_CONFLICTS

Displays statistics about the conflict status of the current node in the database, as described in **Table 8-54**.

Table 8-54 DBE_PERF.STAT_DATABASE_CONFLICTS columns

Name	Type	Description
datid	oid	Database ID.
datname	name	Database name.
confl_tablespace	bigint	Number of conflicting tablespaces.
confl_lock	bigint	Number of conflicting locks.
confl_snapshot	bigint	Number of conflicting snapshots.
confl_bufferpin	bigint	Number of conflicting buffers.
confl_deadlock	bigint	Number of conflicting deadlocks.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.5.23 DBE_PERF.SUMMARY_STAT_DATABASE_CONFLICTS

Displays the sum of conflict status statistics of each node in the cluster. It is used on CNs and the conflict status statistics of each node are summed up by database name, as described in **Table 8-55**.

Table 8-55 DBE_PERF.SUMMARY_STAT_DATABASE_CONFLICTS columns

Name	Type	Description
datname	name	Database name.
confl_tablespace	bigint	Number of conflicting tablespaces.
confl_lock	bigint	Number of conflicting locks.
confl_snapshot	bigint	Number of conflicting snapshots.
confl_bufferpin	bigint	Number of conflicting buffers.
confl_deadlock	bigint	Number of conflicting deadlocks.

8.2.5.24 DBE_PERF.GLOBAL_STAT_DATABASE_CONFLICTS

Displays statistics about the conflict status of each node in the cluster. It is used on CNs and the status information of each database node is not summed up, as described in **Table 8-56**.

Table 8-56 DBE_PERF.GLOBAL_STAT_DATABASE_CONFLICTS columns

Name	Type	Description
node_name	name	Node name.
datid	oid	Database ID.
datname	name	Database name.
confl_tablespace	bigint	Number of conflicting tablespaces.
confl_lock	bigint	Number of conflicting locks.
confl_snapshot	bigint	Number of conflicting snapshots.
confl_bufferpin	bigint	Number of conflicting buffers.
confl_deadlock	bigint	Number of conflicting deadlocks.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.5.25 DBE_PERF.STAT_XACT_ALL_TABLES

Displays transaction status information about all ordinary tables and TOAST tables in schemas on the current node, as described in **Table 8-57**.

Table 8-57 DBE_PERF.STAT_XACT_ALL_TABLES columns

Name	Type	Description
relid	oid	OID of a table.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).

8.2.5.26 DBE_PERF.SUMMARY_STAT_XACT_ALL_TABLES

Displays the sum of transaction status of all ordinary tables and TOAST tables in schemas of each node in the cluster. It is used on CNs and the transaction status statistics of tables on each node are summed up by table name, as described in **Table 8-58**.

Table 8-58 DBE_PERF.SUMMARY_STAT_XACT_ALL_TABLES columns

Name	Type	Description
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	numeric	Number of sequential scans initiated on the table.
seq_tup_read	numeric	Number of live rows fetched by sequential scans.
idx_scan	numeric	Number of index scans initiated on the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
idx_tup_fetch	numeric	Number of live rows fetched by index scans.
n_tup_ins	numeric	Number of rows inserted.
n_tup_upd	numeric	Number of rows updated.
n_tup_del	numeric	Number of rows deleted.
n_tup_hot_upd	numeric	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).

8.2.5.27 DBE_PERF.GLOBAL_STAT_XACT_ALL_TABLES

Displays the transaction status information of all ordinary tables and TOAST tables in the schemas of each node in the cluster, as described in **Table 8-59**. (It is used on CNs. The transaction status information of the same table name on different nodes is not summed up.)

Table 8-59 DBE_PERF.GLOBAL_STAT_XACT_ALL_TABLES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of a table.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.5.28 DBE_PERF.STAT_XACT_SYS_TABLES

Displays transaction status information about the system catalogs in the schemas of the current node, as described in **Table 8-60**.

Table 8-60 DBE_PERF.STAT_XACT_SYS_TABLES columns

Name	Type	Description
relid	oid	OID of a table.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).

8.2.5.29 DBE_PERF.SUMMARY_STAT_XACT_SYS_TABLES

Displays the sum of the transaction status of system catalogs in schemas on each node in the cluster. It is used on CNs and the transaction status statistics of tables on each node are summed up by table name, as described in **Table 8-61**.

Table 8-61 DBE_PERF.SUMMARY_STAT_XACT_SYS_TABLES columns

Name	Type	Description
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	numeric	Number of sequential scans initiated on the table.
seq_tup_read	numeric	Number of live rows fetched by sequential scans.
idx_scan	numeric	Number of index scans initiated on the table.
idx_tup_fetch	numeric	Number of live rows fetched by index scans.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
n_tup_ins	numeric	Number of rows inserted.
n_tup_upd	numeric	Number of rows updated.
n_tup_del	numeric	Number of rows deleted.
n_tup_hot_upd	numeric	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).

8.2.5.30 DBE_PERF.GLOBAL_STAT_XACT_SYS_TABLES

Displays the transaction status information of the system catalogs in the schemas of each node in the cluster, as described in **Table 8-62**. (It is used on CNs. The transaction status information of the same table on different nodes is not summed up.)

Table 8-62 DBE_PERF.GLOBAL_STAT_XACT_SYS_TABLES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of a table.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.5.31 DBE_PERF.STAT_XACT_USER_TABLES

Displays transaction status information about the user tables in schemas on the current node, as described in **Table 8-63**.

Table 8-63 DBE_PERF.STAT_XACT_USER_TABLES columns

Name	Type	Description
relid	oid	OID of a table.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).

8.2.5.32 DBE_PERF.SUMMARY_STAT_XACT_USER_TABLES

Displays the sum of the transaction status of user tables in schemas on each node in the cluster. It is used on CNs and the transaction status statistics of tables on each node are summed up by table name, as described in **Table 8-64**.

Table 8-64 DBE_PERF.SUMMARY_STAT_XACT_USER_TABLES columns

Name	Type	Description
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	numeric	Number of sequential scans initiated on the table.
seq_tup_read	numeric	Number of live rows fetched by sequential scans.
idx_scan	numeric	Number of index scans initiated on the table.
idx_tup_fetch	numeric	Number of live rows fetched by index scans.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
n_tup_ins	numeric	Number of rows inserted.
n_tup_upd	numeric	Number of rows updated.
n_tup_del	numeric	Number of rows deleted.
n_tup_hot_upd	numeric	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).

8.2.5.33 DBE_PERF.GLOBAL_STAT_XACT_USER_TABLES

Displays the transaction status information of the user tables in the schemas of each node in the cluster, as described in **Table 8-65**. (It is used on CNs. The transaction status information of the same table on different nodes is not summed up.)

Table 8-65 DBE_PERF.GLOBAL_STAT_XACT_USER_TABLES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of a table.
schemaname	name	Name of the schema where the table is located.
relname	name	Table name.
seq_scan	bigint	Number of sequential scans initiated on the table.
seq_tup_read	bigint	Number of live rows fetched by sequential scans.
idx_scan	bigint	Number of index scans initiated on the table.
idx_tup_fetch	bigint	Number of live rows fetched by index scans.
n_tup_ins	bigint	Number of rows inserted.
n_tup_upd	bigint	Number of rows updated.
n_tup_del	bigint	Number of rows deleted.
n_tup_hot_upd	bigint	Number of HOT updated rows (that is, the number of rows whose index columns are not updated).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.5.34 DBE_PERF.STAT_XACT_USER_FUNCTIONS

Displays statistics about function executions in the current transaction on the current database node, as described in **Table 8-66**.

Table 8-66 DBE_PERF.STAT_XACT_USER_FUNCTIONS columns

Name	Type	Description
funcid	oid	OID of the function.
schemaname	name	Name of the schema where the function is located.
funcname	name	Function name.
calls	bigint	Number of times the function has been called.
total_time	double precision	Total time spent in this function and all other functions called by it.
self_time	double precision	Time spent in this function, excluding other functions called by it.

8.2.5.35 DBE_PERF.SUMMARY_STAT_XACT_USER_FUNCTIONS

Displays the summary of function execution statistics in the current transaction on each node in the cluster, as described in **Table 8-67**. (It is used on CNs.)

Table 8-67 DBE_PERF.SUMMARY_STAT_XACT_USER_FUNCTIONS columns

Name	Type	Description
schemaname	name	Name of the schema where the function is located.
funcname	name	Function name.
calls	numeric	Number of times the function has been called.
total_time	double precision	Total time spent in this function and all other functions called by it.
self_time	double precision	Time spent in this function, excluding other functions called by it.

8.2.5.36 DBE_PERF.GLOBAL_STAT_XACT_USER_FUNCTIONS

Displays statistics about function execution in the current transaction on each node in the cluster, as described in **Table 8-68**. (It is used on CNs.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-68 DBE_PERF.GLOBAL_STAT_XACT_USER_FUNCTIONS columns

Name	Type	Description
node_name	name	Node name.
funcid	oid	OID of the function.
schemaname	name	Name of the schema where the function is located.
funcname	name	Function name.
calls	bigint	Number of times the function has been called.
total_time	double precision	Total time spent in this function and all other functions called by it.
self_time	double precision	Time spent in this function, excluding other functions called by it.

8.2.5.37 DBE_PERF.STAT_BAD_BLOCK

Obtains the information about table and index read failures on the current node, as described in **Table 8-69**.

Table 8-69 DBE_PERF.STAT_BAD_BLOCK columns

Name	Type	Description
nodename	text	Node name.
databaseid	integer	Database OID.
tablespaceid	integer	Tablespace OID.
relfilenode	integer	Relationship filenode.
bucketid	smallint	ID of the bucket for consistent hashing.
forknum	integer	Fork number.
error_count	integer	Number of errors.
first_time	timestamp with time zone	Time when the page damage occurs for the first time.
last_time	timestamp with time zone	Time when the page damage occurs for the last time.

8.2.5.38 DBE_PERF.SUMMARY_STAT_BAD_BLOCK

Obtains the sum of the table and index read failure information on each node in the cluster, as described in **Table 8-70**. It is used on CNs. The statistics of different

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

DNs is summed up. The value of **first_time** is the statistics collection start time, and the value of **last_time** is the statistics collection end time.

Table 8-70 DBE_PERF.SUMMARY_STAT_BAD_BLOCK columns

Name	Type	Description
databaseid	integer	Database OID.
tablespaceid	integer	Tablespace OID.
relfilenode	integer	Relationship filenode.
forknum	bigint	Fork number.
error_count	bigint	Number of errors.
first_time	timestamp with time zone	Time when the page damage occurs for the first time.
last_time	timestamp with time zone	Time when the page damage occurs for the last time.

8.2.5.39 DBE_PERF.GLOBAL_STAT_BAD_BLOCK

Obtains information about table and index read failures on each node in the cluster, as described in **Table 8-71**. It is used on CNs. The read failure information of DNs is displayed only, but not summed up.

Table 8-71 DBE_PERF.GLOBAL_STAT_BAD_BLOCK columns

Name	Type	Description
node_name	text	Node name.
databaseid	integer	Database OID.
tablespaceid	integer	Tablespace OID.
relfilenode	integer	Relationship filenode.
forknum	integer	Fork number.
error_count	integer	Number of errors.
first_time	timestamp with time zone	Time when the page damage occurs for the first time.
last_time	timestamp with time zone	Time when the page damage occurs for the last time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.5.40 DBE_PERF.STAT_USER_FUNCTIONS

Displays statistics about user-defined functions (the function language is not an internal language) in all the schemas of the current node, as described in **Table 8-72**.

Table 8-72 DBE_PERF.STAT_USER_FUNCTIONS columns

Name	Type	Description
funcid	oid	OID of the function.
schemaname	name	Schema name.
funcname	name	User-defined function name.
calls	bigint	Number of times the function has been called.
total_time	double precision	Total time spent in this function, including other functions called by it (unit: ms).
self_time	double precision	Time spent in this function, excluding other functions called by it (unit: ms).

8.2.5.41 DBE_PERF.SUMMARY_STAT_USER_FUNCTIONS

Displays the sum of statistics about user-defined functions on each node in the cluster. It is used on CNs and the status information of user-defined functions on each node is summed up by function name, as described in **Table 8-73**.

Table 8-73 DBE_PERF.SUMMARY_STAT_USER_FUNCTIONS columns

Name	Type	Description
schemaname	name	Schema name.
funcname	name	User-defined function name.
calls	numeric	Number of times the function has been called.
total_time	double precision	Total time spent in this function, including other functions called by it (unit: ms).
self_time	double precision	Time spent in this function, excluding other functions called by it (unit: ms).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.5.42 DBE_PERF.GLOBAL_STAT_USER_FUNCTIONS

Displays statistics about user-defined functions on each node in the cluster, as described in **Table 8-74**. (It is used on CNs. Status information about functions with the same name on different nodes is not summed up.)

Table 8-74 DBE_PERF.GLOBAL_STAT_USER_FUNCTIONS columns

Name	Type	Description
node_name	name	Node name.
funcid	oid	OID of the function.
schemaname	name	Name of the schema where the function is located.
funcname	name	User-defined function name.
calls	bigint	Number of times the function has been called.
total_time	double precision	Total time spent in this function, including other functions called by it (unit: ms).
self_time	double precision	Time spent in this function, excluding other functions called by it (unit: ms).

8.2.6 Workload

8.2.6.1 DBE_PERF.WORKLOAD_SQL_COUNT

Displays the distribution of SQL statements in workloads on the current node, as described in **Table 8-75**.

Table 8-75 DBE_PERF.WORKLOAD_SQL_COUNT columns

Name	Type	Description
workload	name	Workload name.
select_count	bigint	Number of SELECT statements.
update_count	bigint	Number of UPDATE statements.
insert_count	bigint	Number of INSERT statements.
delete_count	bigint	Number of DELETE statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
ddl_count	bigint	Number of DDL statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including TRUNCATE, CLUSTER, VACUUM, ANALYZE, CREATE (excluding CREATE ROLE and CREATE USER), DROP (excluding DROP ROLE and DROP USER), ALTER (excluding ALTER ROLE, ALTER USER, ALTER DEFAULT PRIVILEGES, and ALTER SESSION), REINDEX, COMMENT, BARRIER, SECURITY LABEL ON, PURGE, TIMECAPSULE, REFRESH, IMPDP, and EXPDP statements.
dml_count	bigint	Number of DML statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including SELECT, UPDATE, INSERT, DELETE, MERGE, PREPARE, EXPLAIN, LOCK, COPY, ANONYMOUS, DELTA, EXECUTE, MOVE, FETCH, CLOSE, DEALLOCATE, DECLARE, CURSOR, and LOAD statements.
dcl_count	bigint	Number of DCL statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including SHOW, CREATE ROLE, CREATE USER, DROP ROLE, DROP USER, ALTER ROLE, ALTER USER, ALTER DEFAULT PRIVILEGES, GRANT, REVOKE, REASSIGN, SET, RESET, and ALTER SESSION statements.
dbid	oid	ID of the database to which the statistics belong.
user_dml_count	bigint	Number of DML statements executed by users. The value of **user_dml_count** is equal to the value of **dml_count** minus the value of **bg_dml_count**.
bg_dml_count	bigint	Number of DML statements executed by background threads (excluding WORKER and THREADPOOL_WORKER) in the database in all scenarios. **NOTE** The DML statements executed by the current stored procedure cannot be counted.
fqs_count	bigint	Number of Fast Query Shipping (FQS) statements. The statistical result contains the FQS statements executed by users and the FQS statements executed by the database background thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
lightproxy_count	bigint	Number of lightweight CN statements. The statistical result contains the lightweight CN statements executed by users and the lightweight CN statements executed by the database background thread.
stream_count	bigint	Number of stream statements. The statistical result contains the stream statements executed by users and the stream statements executed by the database background thread.
pgxc_count	bigint	Number of PGXC statements. The statistical result contains the PGXC statements executed by users and the stream statements executed by the database background thread.

8.2.6.2 DBE_PERF.SUMMARY_WORKLOAD_SQL_COUNT

Displays the distribution of SQL statements in workloads on each CN in the cluster, as described in **Table 8-76**.

Table 8-76 DBE_PERF.SUMMARY_WORKLOAD_SQL_COUNT columns

Name	Type	Description
node_name	name	Node name.
workload	name	Workload name.
select_count	bigint	Number of SELECT statements.
update_count	bigint	Number of UPDATE statements.
insert_count	bigint	Number of INSERT statements.
delete_count	bigint	Number of DELETE statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
ddl_count	bigint	Number of DDL statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including TRUNCATE, CLUSTER, VACUUM, ANALYZE, CREATE (excluding CREATE ROLE and CREATE USER), DROP (excluding DROP ROLE and DROP USER), ALTER (excluding ALTER ROLE, ALTER USER, ALTER DEFAULT PRIVILEGES, and ALTER SESSION), REINDEX, COMMENT, BARRIER, SECURITY LABEL ON, PURGE, TIMECAPSULE, REFRESH, IMPDP, and EXPDP statements.
dml_count	bigint	Number of DML statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including SELECT, UPDATE, INSERT, DELETE, MERGE, PREPARE, EXPLAIN, LOCK, COPY, ANONYMOUS, DELTA, EXECUTE, MOVE, FETCH, CLOSE, DEALLOCATE, DECLARE, CURSOR, and LOAD statements.
dcl_count	bigint	Number of DCL statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including SHOW, CREATE ROLE, CREATE USER, DROP ROLE, DROP USER, ALTER ROLE, ALTER USER, ALTER DEFAULT PRIVILEGES, GRANT, REVOKE, REASSIGN, SET, RESET, and ALTER SESSION statements.
dbid	oid	ID of the database to which the statistics belong.
user_dml_count	bigint	Number of DML statements executed by users. The value of **user_dml_count** is equal to the value of **dml_count** minus the value of **bg_dml_count**.
bg_dml_count	bigint	Number of DML statements executed by background threads (excluding WORKER and THREADPOOL_WORKER) in the database in all scenarios. **NOTE** The DML statements executed by the current stored procedure cannot be counted.
fqs_count	bigint	Number of Fast Query Shipping (FQS) statements. The statistical result contains the FQS statements executed by users and the FQS statements executed by the database background thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
lightproxy_count	bigint	Number of lightweight CN statements. The statistical result contains the lightweight CN statements executed by users and the lightweight CN statements executed by the database background thread.
stream_count	bigint	Number of stream statements. The statistical result contains the stream statements executed by users and the stream statements executed by the database background thread.
pgxc_count	bigint	Number of PGXC statements. The statistical result contains the PGXC statements executed by users and the PGXC statements executed by the database background thread.

8.2.6.3 DBE_PERF.WORKLOAD_TRANSACTION

Displays information about transactions loaded on the current node, as described in **Table 8-77**.

Table 8-77 DBE_PERF.WORKLOAD_TRANSACTION columns

Name	Type	Description
workload	name	Workload name.
commit_counter	bigint	Number of user transactions committed.
rollback_counter	bigint	Number of user transactions rolled back.
resp_min	bigint	Minimum response time of user transactions (unit: μs).
resp_max	bigint	Maximum response time of user transactions (unit: μs).
resp_avg	bigint	Average response time of user transactions (unit: μs).
resp_total	bigint	Total response time of user transactions (unit: μs).
bg_commit_counter	bigint	Number of background transactions committed.
bg_rollback_counter	bigint	Number of background transactions rolled back.
bg_resp_min	bigint	Minimum response time of background transactions (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
bg_resp_max	bigint	Maximum response time of background transactions (unit: μs).
bg_resp_avg	bigint	Average response time of background transactions (unit: μs).
bg_resp_total	bigint	Total response time of background transactions (unit: μs).
dist_commit_counter	bigint	Number of distributed transactions that are committed.
dist_rollback_counter	bigint	Number of distributed transactions that are rolled back.
dist_resp_min	bigint	Minimum response time of distributed transactions (unit: μs).
dist_resp_max	bigint	Maximum response time of distributed transactions (unit: μs).
dist_resp_avg	bigint	Average response time of distributed transactions (unit: μs).
dist_resp_total	bigint	Total response time of distributed transactions (unit: μs).
dbid	oid	ID of the database to which the statistics belong.

NOTE

Only the information about the main transaction is collected. The information about the sub-transaction is not collected.

8.2.6.4 DBE_PERF.SUMMARY_WORKLOAD_TRANSACTION

Displays the information about transactions in the cluster, as described in **Table 8-78**.

Table 8-78 DBE_PERF.SUMMARY_WORKLOAD_TRANSACTION columns

Name	Type	Description
workload	name	Workload name.
commit_counter	numeric	Number of user transactions committed.
rollback_counter	numeric	Number of user transactions rolled back.
resp_min	bigint	Minimum response time of user transactions (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
resp_max	bigint	Maximum response time of user transactions (unit: μs).
resp_avg	bigint	Average response time of user transactions (unit: μs).
resp_total	numeric	Total response time of user transactions (unit: μs).
bg_commit_counter	numeric	Number of background transactions committed.
bg_rollback_counter	numeric	Number of background transactions rolled back.
bg_resp_min	bigint	Minimum response time of background transactions (unit: μs).
bg_resp_max	bigint	Maximum response time of background transactions (unit: μs).
bg_resp_avg	bigint	Average response time of background transactions (unit: μs).
bg_resp_total	numeric	Total response time of background transactions (unit: μs).
dist_commit_counter	bigint	Number of distributed transactions that are committed.
dist_rollback_counter	bigint	Number of distributed transactions that are rolled back.
dist_resp_min	bigint	Minimum response time of distributed transactions (unit: μs).
dist_resp_max	bigint	Maximum response time of distributed transactions (unit: μs).
dist_resp_avg	bigint	Average response time of distributed transactions (unit: μs).
dist_resp_total	bigint	Total response time of distributed transactions (unit: μs).
dbid	oid	ID of the database to which the statistics belong.

NOTE

Only the information about the main transaction is collected. The information about the sub-transaction is not collected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.6.5 DBE_PERF.GLOBAL_WORKLOAD_TRANSACTION

Displays workload information on each node in the cluster, as described in **Table 8-79**.

Table 8-79 DBE_PERF.GLOBAL_WORKLOAD_TRANSACTION columns

Name	Type	Description
node_name	name	Node name.
workload	name	Workload name.
commit_counter	bigint	Number of user transactions committed.
rollback_counter	bigint	Number of user transactions rolled back.
resp_min	bigint	Minimum response time of user transactions (unit: μs).
resp_max	bigint	Maximum response time of user transactions (unit: μs).
resp_avg	bigint	Average response time of user transactions (unit: μs).
resp_total	bigint	Total response time of user transactions (unit: μs).
bg_commit_counter	bigint	Number of background transactions committed.
bg_rollback_counter	bigint	Number of background transactions rolled back.
bg_resp_min	bigint	Minimum response time of background transactions (unit: μs).
bg_resp_max	bigint	Maximum response time of background transactions (unit: μs).
bg_resp_avg	bigint	Average response time of background transactions (unit: μs).
bg_resp_total	bigint	Total response time of background transactions (unit: μs).
dist_commit_counter	bigint	Number of distributed transactions that are committed.
dist_rollback_counter	bigint	Number of distributed transactions that are rolled back.
dist_resp_min	bigint	Minimum response time of distributed transactions (unit: μs).
dist_resp_max	bigint	Maximum response time of distributed transactions (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
dist_resp_avg	bigint	Average response time of distributed transactions (unit: μs).
dist_resp_total	bigint	Total response time of distributed transactions (unit: μs).
dbid	oid	ID of the database to which the statistics belong.

NOTE

Only the information about the main transaction is collected. The information about the sub-transaction is not collected.

8.2.6.6 DBE_PERF.WORKLOAD_SQL_ELAPSE_TIME

Collects statistics about the select, update, insert, and delete (SUID) operations on (service) workloads, as described in **Table 8-80**.

Table 8-80 DBE_PERF.WORKLOAD_SQL_ELAPSE_TIME columns

Name	Type	Description
workload	name	Service workload name.
total_select_elapse	bigint	Total time spent on SELECT statements (unit: μs)
max_select_elapse	bigint	Maximum time spent on SELECT statements (unit: μs).
min_select_elapse	bigint	Minimum time spent on SELECT statements (unit: μs).
avg_select_elapse	bigint	Average time spent on SELECT statements (unit: μs).
total_update_elapse	bigint	Total time spent on UPDATE statements (unit: μs).
max_update_elapse	bigint	Maximum time spent on UPDATE statements (unit: μs).
min_update_elapse	bigint	Minimum time spent on UPDATE statements (unit: μs).
avg_update_elapse	bigint	Average time spent on UPDATE statements (unit: μs).
total_insert_elapse	bigint	Total time spent on INSERT statements (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
max_insert_elapse	bigint	Maximum time spent on INSERT statements (unit: μs).
min_insert_elapse	bigint	Minimum time spent on INSERT statements (unit: μs).
avg_insert_elapse	bigint	Average time spent on INSERT statements (unit: μs).
total_delete_elapse	bigint	Total time spent on DELETE statements (unit: μs).
max_delete_elapse	bigint	Maximum time spent on DELETE statements (unit: μs).
min_delete_elapse	bigint	Minimum time spent on DELETE statements (unit: μs).
avg_delete_elapse	bigint	Average time spent on DELETE statements (unit: μs).
dbid	oid	ID of the database to which the statistics belong.

8.2.6.7 DBE_PERF.SUMMARY_WORKLOAD_SQL_ELAPSE_TIME

Collects statistics about the select, update, insert, and delete (SUID) operations on the (service) workloads of all CNs, as described in **Table 8-81**.

Table 8-81 DBE_PERF.SUMMARY_WORKLOAD_SQL_ELAPSE_TIME columns

Name	Type	Description
node_name	name	Node name.
workload	name	Service workload name.
total_select_elapse	bigint	Total time spent on SELECT statements (unit: μs)
max_select_elapse	bigint	Maximum time spent on SELECT statements (unit: μs).
min_select_elapse	bigint	Minimum time spent on SELECT statements (unit: μs).
avg_select_elapse	bigint	Average time spent on SELECT statements (unit: μs).
total_update_elapse	bigint	Total time spent on UPDATE statements (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
max_update_elapse	bigint	Maximum time spent on UPDATE statements (unit: μs).
min_update_elapse	bigint	Minimum time spent on UPDATE statements (unit: μs).
avg_update_elapse	bigint	Average time spent on UPDATE statements (unit: μs).
total_insert_elapse	bigint	Total time spent on INSERT statements (unit: μs).
max_insert_elapse	bigint	Maximum time spent on INSERT statements (unit: μs).
min_insert_elapse	bigint	Minimum time spent on INSERT statements (unit: μs).
avg_insert_elapse	bigint	Average time spent on INSERT statements (unit: μs).
total_delete_elapse	bigint	Total time spent on DELETE statements (unit: μs).
max_delete_elapse	bigint	Maximum time spent on DELETE statements (unit: μs).
min_delete_elapse	bigint	Minimum time spent on DELETE statements (unit: μs).
avg_delete_elapse	bigint	Average time spent on DELETE statements (unit: μs).
dbid	oid	ID of the database to which the statistics belong.

8.2.6.8 DBE_PERF.USER_TRANSACTION

Collects statistics about transactions executed by users, as described in **Table 8-82**. MONADMIN can view information about transactions executed by all users.

Table 8-82 DBE_PERF.USER_TRANSACTION columns

Name	Type	Description
username	name	Username.
commit_counter	bigint	Number of user transactions committed.
rollback_counter	bigint	Number of user transactions rolled back.
resp_min	bigint	Minimum response time of transactions committed by users (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
resp_max	bigint	Maximum response time of transactions committed by users (unit: μs).
resp_avg	bigint	Average response time of transactions committed and rolled back by users (unit: μs).
resp_total	bigint	Total response time of transactions committed by users (unit: μs).
bg_commit_counter	bigint	Number of background transactions committed.
bg_rollback_counter	bigint	Number of background transactions rolled back.
bg_resp_min	bigint	Minimum response time of background transactions (unit: μs).
bg_resp_max	bigint	Maximum response time of background transactions (unit: μs).
bg_resp_avg	bigint	Average response time of background transactions (unit: μs).
bg_resp_total	bigint	Total response time of background transactions (unit: μs).
dist_commit_counter	bigint	Number of distributed transactions that are committed.
dist_rollback_counter	bigint	Number of distributed transactions that are rolled back.
dist_resp_min	bigint	Minimum response time of distributed transactions that are committed (unit: μs).
dist_resp_max	bigint	Maximum response time of distributed transactions that are committed (unit: μs).
dist_resp_avg	bigint	Average response time of distributed transactions that are committed and rolled back (unit: μs).
dist_resp_total	bigint	Total response time of distributed transactions that are committed (unit: μs).
dbid	oid	ID of the database to which the statistics belong.

NOTE

Only the information about the main transaction is collected. The information about the sub-transaction is not collected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.6.9 DBE_PERF.GLOBAL_USER_TRANSACTION

Collects statistics about transactions executed by all users, as described in **Table 8-83**.

Table 8-83 DBE_PERF.GLOBAL_USER_TRANSACTION columns

Name	Type	Description
node_name	name	Node name.
username	name	Username.
commit_counter	bigint	Number of user transactions committed.
rollback_counter	bigint	Number of user transactions rolled back.
resp_min	bigint	Minimum response time of transactions committed by users (unit: μs).
resp_max	bigint	Maximum response time of transactions committed by users (unit: μs).
resp_avg	bigint	Average response time of transactions committed and rolled back by users (unit: μs).
resp_total	bigint	Total response time of transactions committed by users (unit: μs).
bg_commit_counter	bigint	Number of background transactions committed.
bg_rollback_counter	bigint	Number of background transactions rolled back.
bg_resp_min	bigint	Minimum response time of background transactions (unit: μs).
bg_resp_max	bigint	Maximum response time of background transactions (unit: μs).
bg_resp_avg	bigint	Average response time of background transactions (unit: μs).
bg_resp_total	bigint	Total response time of background transactions (unit: μs).
dist_commit_counter	bigint	Number of distributed transactions that are committed.
dist_rollback_counter	bigint	Number of distributed transactions that are rolled back.
dist_resp_min	bigint	Minimum response time of distributed transactions that are committed (unit: μs).
dist_resp_max	bigint	Maximum response time of distributed transactions that are committed (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
dist_resp_avg	bigint	Average response time of distributed transactions that are committed and rolled back (unit: μs).
dist_resp_total	bigint	Total response time of distributed transactions that are committed (unit: μs).
dbid	oid	ID of the database to which the statistics belong.

NOTE

Only the information about the main transaction is collected. The information about the sub-transaction is not collected.

8.2.7 Session and Thread

8.2.7.1 DBE_PERF.SESSION_STAT

Collects statistics about session status on the current node based on session threads or the AutoVacuum thread, as described in **Table 8-84**.

Table 8-84 DBE_PERF.SESSION_STAT columns

Name	Type	Description
sessid	text	Thread start time and ID.
statid	integer	Statistics ID.
statname	text	Name of the statistics session.
statunit	text	Unit of the statistics session.
value	bigint	Session value.

8.2.7.2 DBE_PERF.GLOBAL_SESSION_STAT

Collects statistics about session status on each node based on session threads or the AutoVacuum thread, as described in **Table 8-85**.

Table 8-85 DBE_PERF.GLOBAL_SESSION_STAT columns

Name	Type	Description
node_name	name	Node name.
sessid	text	Thread start time and ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
statid	integer	Statistics ID.
statname	text	Name of the statistics session.
statunit	text	Unit of the statistics session.
value	bigint	Session value.

8.2.7.3 DBE_PERF.SESSION_TIME

Collects statistics about the running time of session threads and time consumed in each execution phase on the current node, as described in **Table 8-86**.

Table 8-86 DBE_PERF.SESSION_TIME columns

Name	Type	Description
sessid	text	Thread start time and ID.
stat_id	integer	Statistics ID.
stat_name	text	Session type.
value	bigint	Session value.

8.2.7.4 DBE_PERF.GLOBAL_SESSION_TIME

Collects statistics about the running time of session threads and time consumed in each execution phase on each node, as described in **Table 8-87**.

Table 8-87 DBE_PERF.GLOBAL_SESSION_TIME columns

Name	Type	Description
node_name	name	Node name.
sessid	text	Thread start time and ID.
stat_id	integer	Statistics ID.
stat_name	text	Session type.
value	bigint	Session value.

8.2.7.5 DBE_PERF.SESSION_MEMORY

Collects statistics about memory usage at the session level in the unit of MB, including all the memory allocated to GaussDB and stream threads on DNs for jobs currently executed by users, as described in **Table 8-88**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-88 DBE_PERF.SESSION_MEMORY columns

Name	Type	Description
sessid	text	Thread start time and ID.
init_mem	integer	Memory allocated to the currently executed job before the job enters the executor.
used_mem	integer	Memory allocated to the currently executed job.
peak_mem	integer	Peak memory allocated to the currently executed job.

8.2.7.6 DBE_PERF.GLOBAL_SESSION_MEMORY

Collects statistics about memory usage at the session level on each node in the unit of MB, including all the memory allocated to GaussDB and stream threads on DNs for jobs currently executed by users, as described in **Table 8-89**.

Table 8-89 DBE_PERF.GLOBAL_SESSION_MEMORY columns

Name	Type	Description
node_name	name	Node name.
sessid	text	Thread start time and ID.
init_mem	integer	Memory allocated to the currently executed job before the job enters the executor.
used_mem	integer	Memory allocated to the currently executed job.
peak_mem	integer	Peak memory allocated to the currently executed job.

8.2.7.7 DBE_PERF.SESSION_MEMORY_DETAIL

Collects statistics on the memory usage of threads on the current node, in the unit specified by **MemoryContext**, as described in **Table 8-90**.

Table 8-90 DBE_PERF.SESSION_MEMORY_DETAIL columns

Name	Type	Description
sessid	text	Thread start time and ID.
sesstype	text	Thread name.
contextname	text	Name of the memory context.
level	smallint	Level of memory context importance.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
parent	text	Name of the parent memory context.
totalsize	bigint	Size of the allocated memory (unit: byte).
freesize	bigint	Size of the idle memory (unit: byte).
usedsize	bigint	Size of the used memory (unit: byte).

8.2.7.8 DBE_PERF.GLOBAL_SESSION_MEMORY_DETAIL

Collects statistics about thread memory usage on each node by **MemoryContext**, as described in **Table 8-91**.

Table 8-91 DBE_PERF.GLOBAL_SESSION_MEMORY_DETAIL columns

Name	Type	Description
node_name	name	Node name.
sessid	text	Thread start time and ID.
sesstype	text	Thread name.
contextname	text	Name of the memory context.
level	smallint	Level of memory context importance.
parent	text	Name of the parent memory context.
totalsize	bigint	Size of the allocated memory (unit: byte).
freesize	bigint	Size of the idle memory (unit: byte).
usedsize	bigint	Size of the used memory (unit: byte).

8.2.7.9 DBE_PERF.SESSION_STAT_ACTIVITY

Displays information about threads that are running on the current node, as described in **Table 8-92**.

Table 8-92 DBE_PERF.SESSION_STAT_ACTIVITY columns

Name	Type	Description
datid	oid	OID of the database that the user session connects to in the backend.
datname	name	Name of the database that the user session connects to in the backend.
pid	bigint	Background thread ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
usesysid	oid	OID of the user logged in to the backend.
usename	name	Name of the user logged in to the backend.
application_name	text	Name of the application connected to the backend.
client_addr	inet	IP address of the client connected to the backend. If this column is **NULL**, it indicates either the client is connected via a UDS on the server or this is an internal thread. For an autovacuum worker thread, this column will display **autovacuum**.
client_hostname	text	Host name of the connected client, as reported by a reverse DNS lookup of **client_addr**. This column will be non-null only for IP connections and only when **log_hostname** is enabled.
client_port	integer	TCP port number that the client uses for communication with the backend (**–1** if a Unix socket is used).
backend_start	timestamp with time zone	Time when this process was started, that is, when the client connected to the server.
xact_start	timestamp with time zone	Time when current transaction was started (null if no transaction is active). If the current query is the first of its transaction, the value of this column is the same as that of the **query_start** column.
query_start	timestamp with time zone	Time when the currently active query was started, or time when the last query was started if **state** is not **active**. For a stored procedure, function, or package, the first query time is queried and does not change with the running of statements in the stored procedure.
state_change	timestamp with time zone	Time when **state** was last modified.
waiting	Boolean	Specifies whether the backend is currently waiting for a lock. If the backend is currently waiting for a lock, the value is **true**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
state	text	Overall status of this backend. The value can be: • **active**: The backend is executing a query. • **idle**: The backend is waiting for a new client command. • **idle in transaction**: The backend is in a transaction, but is not currently executing a query. • **idle in transaction** (aborted): The backend is in a transaction, but there are statements failed in the transaction. • **fastpath function call**: The backend is executing a fast-path function. • **disabled**: This state is reported if **track_activities** is disabled in this backend. **NOTE** Common users can view their own session status only. The state information of other accounts is empty. For example, after the **judy** user is connected to the database, the state information of the **joe** user and the initial user **omm** in pg_stat_activity is empty. gaussdb=# SELECT datname, usename, usesysid,state,pid FROM pg_stat_activity; datname | usename | usesysid | state | pid ----------+---------+----------+-------- +-----------------testdb | omm | 10 | | 139968752121616 testdb | omm | 10 | | 139968903116560 db_tpcds | judy | 16398 | active | 139968391403280 testdb | omm | 10 | | 139968643069712 testdb | omm | 10 | | 139968680818448 testdb | joe | 16390 | | 139968563377936 (6 rows)
resource_pool	name	Resource pool used by the user.
query_id	bigint	ID of a query.
query	text	Latest query at the backend. If **state** is **active**, this column shows the ongoing query. In all other states, it shows the last query that was executed.
unique_sql_id	bigint	Unique SQL statement ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
trace_id	text	Trace ID specified by the driver, which is used to identify a request of an application.

8.2.7.10 DBE_PERF.GLOBAL_SESSION_STAT_ACTIVITY

Displays information about threads that are running on each node in the cluster, as described in **Table 8-93**.

Table 8-93 DBE_PERF.GLOBAL_SESSION_STAT_ACTIVITY columns

Name	Type	Description
coorname	text	CN name.
datid	oid	OID of the database that the user session connects to in the backend.
datname	text	Name of the database that the user session connects to in the backend.
pid	bigint	Background thread ID.
usesysid	oid	OID of the user logged in to the backend.
usename	text	Name of the user logged in to the backend.
application_name	text	Name of the application connected to the backend.
client_addr	inet	IP address of the client connected to the backend. If this column is **NULL**, it indicates either the client is connected via a UDS on the server or this is an internal thread. For an autovacuum worker thread, this column will display **autovacuum**.
client_hostname	text	Host name of the connected client, as reported by a reverse DNS lookup of **client_addr**. This column will be non-null only for IP connections and only when **log_hostname** is enabled.
client_port	integer	TCP port number that the client uses for communication with the backend (**–1** if a Unix socket is used).
backend_start	timestamp with time zone	Time when this process was started, that is, when the client connected to the server.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
xact_start	timestamp with time zone	Time when current transaction was started (null if no transaction is active). If the current query is the first of its transaction, the value of this column is the same as that of the **query_start** column.
query_start	timestamp with time zone	Time when the currently active query was started, or time when the last query was started if **state** is not **active**. For a stored procedure, function, or package, the first query time is queried and does not change with the running of statements in the stored procedure.
state_change	timestamp with time zone	Time when **state** was last modified.
waiting	Boolean	If the background is waiting for a lock, the value is **true**. Otherwise, the value is **false**. If a lock is being waited for on a DN, determine the DN based on the **wait_status** column in the pg_thread_wait_status view and query the **waiting** column in pg_stat_activity on the DN to check whether a lock wait occurs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
state	text	Overall status of this backend. The value can be: • **active**: The backend is executing a query. • **idle**: The backend is waiting for a new client command. • **idle in transaction**: The backend is in a transaction, but is not currently executing a query. • **idle in transaction** (aborted): The backend is in a transaction, but there are statements failed in the transaction. • **fastpath function call**: The backend is executing a fast-path function. • **disabled**: This state is reported if **track_activities** is disabled in this backend. **NOTE** Common users can view their own session status only. The state information of other accounts is empty. For example, after the **judy** user is connected to the database, the state information of the **joe** user and the initial user **omm** in pg_stat_activity is empty. gaussdb=# SELECT datname, username, usesysid,state,pid FROM pg_stat_activity; datname | username | usesysid | state | pid ----------+----------+----------+-------- +-----------------testdb | omm | 10 | | 139968752121616 testdb | omm | 10 | | 139968903116560 db_tpcds | judy | 16398 | active | 139968391403280 testdb | omm | 10 | | 139968643069712 testdb | omm | 10 | | 139968680818448 testdb | joe | 16390 | | 139968563377936 (6 rows)
resource_pool	name	Resource pool used by the user.
query_id	bigint	ID of a query.
query	text	Latest query at the backend. If **state** is **active**, this column shows the ongoing query. In all other states, it shows the last query that was executed.
unique_sql_id	bigint	Unique SQL statement ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
trace_id	text	Trace ID specified by the driver, which is used to identify a request of an application.

8.2.7.11 DBE_PERF.THREAD_WAIT_STATUS

In this view, you can check the blocking and waiting status of the backend threads and auxiliary threads on the current node, as described in **Table 8-94**. For details about events, see **Table 8-192**.

Table 8-94 DBE_PERF.THREAD_WAIT_STATUS columns

Name	Type	Description
node_name	text	Node name.
db_name	text	Database name.
thread_name	text	Thread name.
query_id	bigint	Query ID. The value of this column is the same as that of **debug_query_id**.
tid	bigint	Thread ID of the current thread.
sessionid	bigint	Session ID.
lwtid	integer	Lightweight thread ID of the current thread.
psessionid	bigint	Parent thread of the streaming thread.
tlevel	integer	Level of the streaming thread.
smpid	integer	Parallel thread ID.
wait_status	text	Waiting status of the current thread. For details about the waiting status, see **Table 8-192**.
wait_event	text	If **wait_status** is **acquire lock**, **acquire lwlock**, or **wait io**, this column describes the lock, lightweight lock, or I/O information. Otherwise, this column is empty.
locktag	text	Information about the lock that the current thread is waiting for.
lockmode	text	Lock mode that the current thread is waiting to obtain. The values include modes in the table-level lock, row-level lock, and page-level lock.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
block_sessionid	bigint	ID of the session that blocks the current thread from obtaining the lock.
global_sessionid	text	Global session ID.

8.2.7.12 DBE_PERF.GLOBAL_THREAD_WAIT_STATUS

In this view, you can check the blocking and waiting status of the backend threads and auxiliary threads on all nodes, as described in **Table 8-95**. For details about the events, see **Table 8-192**.

In GLOBAL_THREAD_WAIT_STATUS, you can see all the call hierarchy relationships between threads of the SQL statements on all nodes in the cluster, and the blocking status for each thread. With this view, you can easily locate the causes of process hang and similar issues.

The definitions of GLOBAL_THREAD_WAIT_STATUS and THREAD_WAIT_STATUS are the same because the essence of the GLOBAL_THREAD_WAIT_STATUS view is the query summary of the THREAD_WAIT_STATUS view on each node in the cluster.

Table 8-95 DBE_PERF.GLOBAL_THREAD_WAIT_STATUS columns

Name	Type	Description
node_name	text	Node name.
db_name	text	Database name.
thread_name	text	Thread name.
query_id	bigint	Query ID. The value of this column is the same as that of **debug_query_id**.
tid	bigint	Thread ID of the current thread.
sessionid	bigint	Session ID.
lwtid	integer	Lightweight thread ID of the current thread.
psessionid	bigint	Parent thread of the streaming thread.
tlevel	integer	Level of the streaming thread.
smpid	integer	Parallel thread ID.
wait_status	text	Waiting status of the current thread. For details about the waiting status, see **Table 8-192**.
wait_event	text	If **wait_status** is **acquire lock**, **acquire lwlock**, or **wait io**, this column describes the lock, lightweight lock, or I/O information. Otherwise, this column is empty.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
locktag	text	Information about the lock that the current thread is waiting for.
lockmode	text	Lock mode that the current thread is waiting to obtain. The values include modes in the table-level lock, row-level lock, and page-level lock.
block_sessionid	bigint	ID of the session that blocks the current thread from obtaining the lock.
global_sessionid	text	Global session ID.

8.2.7.13 DBE_PERF.LOCAL_THREADPOOL_STATUS

Displays the status of worker threads and sessions in a thread pool, as described in **Table 8-96**. This view is valid only when **enable_thread_pool** is set to **on**.

Table 8-96 DBE_PERF.LOCAL_THREADPOOL_STATUS columns

Name	Type	Description
node_name	text	Node name.
group_id	integer	ID of the thread pool group.
bind_numa_id	integer	NUMA ID to which the thread pool group is bound.
bind_cpu_number	integer	Information about the CPU to which the thread pool group is bound. If no CPUs are bound, the value is **NULL**.
listener	integer	Number of listener threads in the thread pool group.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
worker_info	text	Information about threads in the thread pool, including: • **default**: Number of initial threads in the thread pool group. • **new**: Number of new threads in the thread pool group. • **expect**: Expected number of threads in the thread pool group. • **actual**: Actual number of threads in the thread pool group. • **idle**: Number of idle threads in the thread pool group. • **pending**: Number of pending threads in the thread pool group.
session_info	text	Information about sessions in the thread pool, including: • **total**: Total number of sessions in the thread pool group. • **waiting**: Number of sessions pending scheduling in the thread pool group. • **running**: Number of running sessions in the thread pool group. • **idle**: Number of idle sessions in the thread pool group.
stream_info	text	Information about streams in the thread pool, including: • **total**: total number of stream threads in the thread pool group. • **running**: number of running stream threads in the thread pool group. • **idle**: number of idle stream threads in the thread pool group.

NOTE

The thread pool has a dynamic scaling mechanism. When a new thread is idle after scaling, the thread pool scales in about 3 minutes. After dynamic scale-in, the number of threads in the thread pool is not less than the initial number of threads.

8.2.7.14 DBE_PERF.GLOBAL_THREADPOOL_STATUS

Displays the status of worker threads and sessions in thread pools on all nodes.

For details about columns, see **Table 8-96**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.7.15 DBE_PERF.SESSION_CPU_RUNTIME

Displays information about CPU usage of ongoing complex jobs executed by the current user, as described in **Table 8-97**.

Table 8-97 DBE_PERF.SESSION_CPU_RUNTIME columns

Name	Type	Description
datid	oid	OID of the database that the backend is connected to.
usename	name	Name of the user logged in to the backend.
pid	bigint	Backend thread ID.
start_time	timestamp with time zone	Time when the statement execution starts. For a stored procedure, function, or package, the first query time is queried and does not change with the running of statements in the stored procedure.
min_cpu_time	bigint	Minimum CPU time of the statement across all DNs (unit: ms).
max_cpu_time	bigint	Maximum CPU time of the statement across all DNs (unit: ms).
total_cpu_time	bigint	Total CPU time of the statement across all DNs (unit: ms).
query	text	Statement being executed.
node_group	text	Node group of the user to which the statement belongs.
top_cpu_dn	text	Top *N* CPU usage.

8.2.7.16 DBE_PERF.SESSION_MEMORY_RUNTIME

Displays information about memory usage of ongoing complex jobs executed by the current user, as described in **Table 8-98**.

Table 8-98 DBE_PERF.SESSION_MEMORY_RUNTIME columns

Name	Type	Description
datid	oid	OID of the database that the backend is connected to.
usename	name	Name of the user logged in to the backend.
pid	bigint	Backend thread ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
start_time	timestamp with time zone	Time when the statement execution starts. For a stored procedure, function, or package, the first query time is queried and does not change with the running of statements in the stored procedure.
min_peak_memory	integer	Minimum peak memory of the statement among all DNs (unit: MB).
max_peak_memory	integer	Maximum peak memory of the statement among all DNs (unit: MB).
spill_info	text	Statement spill information on all DNs. • **None**: No data has been spilled to disks on all DNs. • **All**: Data has been spilled to disks on all DNs. • **[*a*:*b*]**: Data has been spilled to disks on *a* of *b* DNs.
query	text	Statement being executed.
node_group	text	Node group of the user to which the statement belongs.
top_mem_dn	text	Top *N* memory usage.

8.2.7.17 DBE_PERF.STATEMENT_IOSTAT_COMPLEX_RUNTIME

Displays I/O workload management information about ongoing jobs executed by the current user, as described in **Table 8-99**. IOPS is counted by 10 thousands per second.

Table 8-99 DBE_PERF.STATEMENT_IOSTAT_COMPLEX_RUNTIME columns

Name	Type	Description
query_id	bigint	Job ID.
mincurriops	integer	Minimum I/O of the current job across DNs.
maxcurriops	integer	Maximum I/O of the current job across DNs.
minpeakiops	integer	Minimum peak I/O of the current job across DNs.
maxpeakiops	integer	Maximum peak I/O of the current job across DNs.
io_limits	integer	**io_limits** set for the job.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
io_priority	text	**io_priority** set for the job.
query	text	Job.
node_group	text	Node group of the user to which a job belongs.
curr_io_limits	integer	Real-time value of **iolimits** when **io_priority** is used to control I/Os.

8.2.7.18 DBE_PERF.LOCAL_ACTIVE_SESSION

Displays samples in the ACTIVE SESSION PROFILE memory on the current node, as described in **Table 8-100**.

Table 8-100 DBE_PERF.LOCAL_ACTIVE_SESSION columns

Name	Type	Description
sampleid	bigint	Sample ID.
sample_time	timestamp with time zone	Sampling time.
need_flush_sample	Boolean	Specifies whether the sample needs to be flushed to disks.
databaseid	oid	Database ID.
thread_id	bigint	Thread ID.
sessionid	bigint	Session ID.
start_time	timestamp with time zone	Start time of a session.
event	text	Event name.
lwtid	integer	Lightweight thread ID of the current thread.
psessionid	bigint	Parent thread of the streaming thread.
tlevel	integer	Level of the streaming thread. The value corresponds to the level (ID) of the execution plan.
smpid	integer	Concurrent thread ID in SMP execution mode.
userid	oid	ID of a session user.
application_name	text	Name of an application.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
client_addr	inet	IP address of a client.
client_hostname	text	Name of a client.
client_port	integer	TCP port number used by a client to communicate with the backend.
query_id	bigint	ID of the unique SQL statement.
unique_query_id	bigint	ID of the normalized SQL statement.
user_id	oid	User ID in the key of the unique query
cn_id	integer	A CN ID on a DN indicates that the unique SQL statement comes from a CN ID in a key of the unique query on a CN.
unique_query	text	Standardized text string of the unique SQL statement.
locktag	text	Information of a lock that the session waits for, which can be parsed using **locktag_decode**.
lockmode	text	Mode of a lock that the session waits for.
block_sessionid	bigint	Blocks a session from obtaining the session ID of a lock if the session is waiting for the lock.
final_block_sessionid	bigint	ID of the blocked session at the source end.
wait_status	text	Provides more details about an event column.
global_sessionid	text	Global session ID.
xact_start_time	timestamp with time zone	Start time of the transaction.
query_start_time	timestamp with time zone	Time when the statement starts to be executed.
state	text	Current statement state The value can be **active**, **idle in transaction**, **fastpath function call**, **idle in transaction (aborted)**, **disabled**, or **retrying**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
event_start_time	timestamp with time zone	Start time of a wait event.
current_xid	xid	Current transaction ID.
top_xid	xid	Top-level transaction ID.
parent_unique_sql_id	bigint	SQL_Id of the parent statement that calls the statement.
control_status	text	The value is **under_cpu_throttle** when CPU resource for abnormal SQL statements is controlled. Otherwise, the value is **normal**.
top_unique_query_id	bigint	If the sampled statement belongs to a stored procedure, this column displays the **unique_query_id** of the top-level SQL statement of the stored procedure.
top_unique_query	text	If the sampled statement belongs to a stored procedure, this column displays the **unique_query** of the top-level SQL statement of the stored procedure.
sql_hash	bigint	Hash value of the SQL statement, which is **0**.
plan_hash	bigint	Hash value of the execution plan of the SQL statement, which is **0**.

8.2.7.19 DBE_PERF.GLOBAL_ACTIVE_SESSION

Displays a summary of samples in the ACTIVE SESSION PROFILE memory on CN and main DN nodes, as described in **Table 8-101**. The **event_start_time**, **current_xid**, and **top_xid** columns are added to GaussDB Kernel 505.0. After GaussDB Kernel is upgraded to GaussDB Kernel 505.0 or later, the columns cannot be queried during the upgrade observation period.

Table 8-101 DBE_PERF.GLOBAL_ACTIVE_SESSION columns

Name	Type	Description
node_name	text	Node name.
sampleid	bigint	Sample ID.
sample_time	timestamp without time zone	Sampling time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
need_flush_sample	Boolean	Specifies whether the sample needs to be refreshed.
databaseid	oid	Database ID.
thread_id	bigint	Thread ID.
sessionid	bigint	Session ID.
start_time	timestamp without time zone	Start time of a session.
event	text	Specified event name.
lwtid	integer	Lightweight thread ID of the current thread.
psessionid	bigint	Parent thread of the streaming thread.
tlevel	integer	Level of the streaming thread. The value corresponds to the level (ID) of the execution plan.
smpid	integer	Concurrent thread ID in SMP execution mode.
userid	oid	ID of a session user.
application_name	text	Name of an application.
client_addr	inet	IP address of a client.
client_hostname	text	Name of a client.
client_port	integer	TCP port number used by a client to communicate with the backend.
query_id	bigint	ID of the unique SQL statement.
unique_query_id	bigint	ID of the normalized SQL statement.
user_id	oid	User ID in the key of the unique query.
cn_id	integer	A CN ID on a DN indicates that the unique SQL statement comes from a CN ID in a key of the unique query on a CN.
unique_query	text	Standardized UniqueSQL text string.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
locktag	text	Information of a lock that the session waits for. It can be parsed using **locktag_decode**.
lockmode	text	Mode of a lock that the session waits for.
block_sessionid	bigint	Blocks a session from obtaining the session ID of a lock if the session is waiting for the lock.
final_block_sessionid	bigint	ID of the blocked session at the source end.
wait_status	text	Provides more details about the event column.
global_sessionid	text	Global session ID.
xact_start_time	timestamp with time zone	Start time of the transaction.
query_start_time	timestamp with time zone	Time when the statement starts to be executed.
state	text	Current statement state. The value can be **active**, **idle in transaction**, **fastpath function call**, **idle in transaction (aborted)**, **disabled**, or **retrying**.
event_start_time	timestamp with time zone	Start time of a wait event.
current_xid	xid	Current transaction ID.
top_xid	xid	Top-level transaction ID.
parent_unique_sql_id	bigint	SQL_Id of the parent statement that calls the statement.
sql_hash	bigint	Hash value of the SQL statement, which is **0**.
plan_hash	bigint	Hash value of the execution plan of the SQL statement, which is **0**.

8.2.8 Transaction

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.8.1 DBE_PERF.TRANSACTIONS_RUNNING_XACTS

Displays information about running transactions on the current node, as described in **Table 8-102**.

Table 8-102 DBE_PERF.TRANSACTIONS_RUNNING_XACTS columns

Name	Type	Description
handle	integer	Handle corresponding to the transaction in GTM.
gxid	xid	Transaction ID.
state	tinyint	Transaction status (**3**: prepared; **0**: starting).
node	text	Node name.
xmin	xid	Minimum transaction ID on the node.
vacuum	Boolean	Specifies whether the current transaction is a lazy vacuum (VACUUM only if necessary) transaction. • **true**: yes. • **false**: no.
timeline	bigint	Number of database restarts.
prepare_xid	xid	ID of the transaction in the **prepared** state (the value is **0** if the state is not **prepared**).
pid	bigint	Thread ID corresponding to the transaction.
next_xid	xid	Transaction ID sent from a CN to a DN.
dbid	oid	ID of the database corresponding to the transaction.

8.2.8.2 DBE_PERF.SUMMARY_TRANSACTIONS_RUNNING_XACTS

Displays information about running transactions on each CN in the cluster, as described in **Table 8-103**.

Table 8-103 DBE_PERF.SUMMARY_TRANSACTIONS_RUNNING_XACTS columns

Name	Type	Description
handle	integer	Handle corresponding to the transaction in GTM.
gxid	xid	Transaction ID.
state	tinyint	Transaction status (**3**: prepared; **0**: starting).
node	text	Node name.
xmin	xid	Minimum transaction ID on the node.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
vacuum	Boolean	Specifies whether the current transaction is a lazy vacuum (VACUUM only if necessary) transaction. • **true**: yes. • **false**: no.
timeline	bigint	Number of database restarts.
prepare_xid	xid	ID of the transaction in the **prepared** state (the value is **0** if the state is not **prepared**).
pid	bigint	Thread ID corresponding to the transaction.
next_xid	xid	Transaction ID sent from a CN to a DN.
dbid	oid	ID of the database corresponding to the transaction.

8.2.8.3 DBE_PERF.GLOBAL_TRANSACTIONS_RUNNING_XACTS

Displays information about the running transactions on each CN and DN in the cluster, as described in **Table 8-104**.

Table 8-104 DBE_PERF.GLOBAL_TRANSACTIONS_RUNNING_XACTS columns

Name	Type	Description
handle	integer	Handle corresponding to the transaction in GTM.
gxid	xid	Transaction ID.
state	tinyint	Transaction status (**3**: prepared; **0**: starting).
node	text	Node name.
xmin	xid	Minimum transaction ID on the node.
vacuum	Boolean	Specifies whether the current transaction is a lazy vacuum (VACUUM only if necessary) transaction. • **true**: yes • **false**: no
timeline	bigint	Number of database restarts.
prepare_xid	xid	ID of the transaction in the **prepared** state (the value is **0** if the state is not **prepared**).
pid	bigint	Thread ID corresponding to the transaction.
next_xid	xid	Transaction ID sent from a CN to a DN.
dbid	oid	ID of the database corresponding to the transaction.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.8.4 DBE_PERF.TRANSACTIONS_PREPARED_XACTS

Displays information about transactions that are prepared for two-phase commit, as described in **Table 8-105**.

Table 8-105 DBE_PERF.TRANSACTIONS_PREPARED_XACTS columns

Name	Type	Description
transaction	xid	Numeric transaction identifier of the prepared transaction.
gid	text	Global transaction identifier that was assigned to the transaction.
prepared	timestamp with time zone	Time at which the transaction is prepared for commit.
owner	name	Name of the user who executes the transaction.
database	name	Name of the database in which the transaction is executed.

8.2.8.5 DBE_PERF.SUMMARY_TRANSACTIONS_PREPARED_XACTS

Summarizes the information about the transactions that are ready for two-phase commit on each CN in the cluster, as described in **Table 8-106**.

Table 8-106 DBE_PERF.SUMMARY_TRANSACTIONS_PREPARED_XACTS columns

Name	Type	Description
transaction	xid	Numeric transaction identifier of the prepared transaction.
gid	text	Global transaction identifier that was assigned to the transaction.
prepared	timestamp with time zone	Time at which the transaction is prepared for commit.
owner	name	Name of the user who executes the transaction.
database	name	Name of the database in which the transaction is executed.

8.2.8.6 DBE_PERF.GLOBAL_TRANSACTIONS_PREPARED_XACTS

Summarizes the information about the transactions that are ready for two-phase commit on each CN and DN in the cluster, as described in **Table 8-107**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-107 DBE_PERF.GLOBAL_TRANSACTIONS_PREPARED_XACTS columns

Name	Type	Description
transaction	xid	Numeric transaction identifier of the prepared transaction.
gid	text	Global transaction identifier that was assigned to the transaction.
prepared	timestamp with time zone	Time at which the transaction is prepared for commit.
owner	name	Name of the user who executes the transaction.
database	name	Name of the database in which the transaction is executed.

8.2.9 Query

8.2.9.1 DBE_PERF.STATEMENT

Obtains information about executed statements (normalized SQL statements) on the current node, as described in **Table 8-108**. You can view all statistics about normalized SQL statements received by the CN, whereas you can view only the statistics about normalized SQL statements executed on the current DN.

NOTE

The **unique_sql_id** generated by different **savepoint_name** values is different. When a large number of **savepoint_name** is used, the number of **unique_sql_id** values generated in the system increases rapidly. If the number of **unique_sql_id** values is greater than the number of **instr_unique_sql_count** values, the newly generated **unique_sql_id** information is not counted.

Table 8-108 DBE_PERF.STATEMENT columns

Name	Type	Description
node_name	name	Node name.
node_id	integer	Node ID (**node_id** in **pgxc_node**).
user_name	name	Username. If the user who runs the query statement does not exist, ***REMOVED_USER*** is displayed.
user_id	oid	OID of the user.
unique_sql_id	bigint	ID of the normalized SQL statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
query	text	Normalized SQL statement. Note: The length is controlled by **instr_unique_sql_size**.
n_calls	bigint	Number of actual calls. The number of times that the FETCH statement in a stored procedure is triggered is the same as the **n_calls** increment of the statement executed by the cursor.
min_elapse_time	bigint	Minimum execution time of the SQL statement in the kernel (unit: μs).
max_elapse_time	bigint	Maximum execution time of the SQL statement in the kernel (unit: μs).
total_elapse_time	bigint	Total execution time of the SQL statement in the kernel (unit: μs).
n_returned_rows	bigint	Number of rows in the result set returned by the SELECT statement.
n_tuples_fetched	bigint	Number of rows randomly scanned.
n_tuples_returned	bigint	Number of rows sequentially scanned.
n_tuples_inserted	bigint	Number of rows inserted.
n_tuples_updated	bigint	Number of rows updated.
n_tuples_deleted	bigint	Number of rows deleted.
n_blocks_fetched	bigint	Number of logical reads.
n_blocks_hit	bigint	Number of memory hits.
n_soft_parse	bigint	Number of soft parsing times. The value of **n_soft_parse** plus the value of **n_hard_parse** may be greater than the value of **n_calls** because the number of subqueries is not counted in the value of **n_calls**. (This is meaningless for non-DML statements.)
n_hard_parse	bigint	Number of hard parsing times. The value of **n_soft_parse** plus the value of **n_hard_parse** may be greater than the value of **n_calls** because the number of subqueries is not counted in the value of **n_calls**. The parsing of complex data types, for example, the smalldatetime type, also increases the number of parsing times. (This is meaningless for non-DML statements.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
db_time	bigint	Valid internal database time, which is accumulated if multiple threads are involved (unit: μs).
cpu_time	bigint	CPU time (unit: μs).
execution_time	bigint	Execution time in the executor (unit: μs).
parse_time	bigint	SQL parsing time (unit: μs). (If an exception occurs during parsing, **parse_time** is not recorded.)
plan_time	bigint	SQL plan generation time (unit: μs). (If an exception occurs during plan generation, **plan_time** is not recorded.)
rewrite_time	bigint	SQL rewriting time (unit: μs). (If an exception occurs during rewriting, **rewrite_time** is not recorded.)
pl_execution_time	bigint	Execution time of PL/pgSQL (unit: μs).
pl_compilation_time	bigint	Compilation time of PL/pgSQL (unit: μs).
data_io_time	bigint	Time spent on I/O (unit: μs).
net_send_info	text	Network status of messages sent through a physical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, CNs communicate with each other, CNs communicate with clients, and CNs communicate with DNs through physical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: {"time":xxx, "n_calls":xxx, "size":xxx}.
net_recv_info	text	Network status of messages received through a physical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, CNs communicate with each other, CNs communicate with clients, and CNs communicate with DNs through physical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: {"time":xxx, "n_calls":xxx, "size":xxx}.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
net_stream_send_info	text	Network status of messages sent through a logical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, DNs of different shards communicate with each other through logical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: {"time":xxx, "n_calls":xxx, "size":xxx}.
net_stream_recv_info	text	Network status of messages received through a logical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, DNs of different shards communicate with each other through logical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: {"time":xxx, "n_calls":xxx, "size":xxx}.
last_updated	timestamp with time zone	Last time when the statement was updated.
sort_count	bigint	Sorting count.
sort_time	bigint	Sorting execution time (unit: μs), including the time of the sort operator and its sub-operators.
sort_mem_used	bigint	Size of work memory used during sorting (unit: KB).
sort_spill_count	bigint	Count of file writing when data is flushed to disks during sorting.
sort_spill_size	bigint	File size used when data is flushed to disks during sorting (unit: KB).
hash_count	bigint	Hashing count.
hash_time	bigint	Hashing duration (unit: μs).
hash_mem_used	bigint	Size of work memory used during hashing (unit: KB).
hash_spill_count	bigint	Count of file writing when data is flushed to disks during hashing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
hash_spill_size	bigint	File size used when data is flushed to disks during hashing (unit: KB).
parent_unique_sql_id	bigint	Unique ID of the parent SQL statement. The value is **0** for a non-stored procedure substatement.
lock_wait_time	bigint	SQL lock waiting time (unit: μs).
total_used_memory	bigint	Total memory used by normalized statements, in MB.
max_used_memory	integer	Maximum memory usage of a single statement, in MB.
min_used_memory	integer	Minimum memory usage of a single statement, in MB.
db_name	name	Name of the database connected to the backend. If the database where the query statement is executed does not exist, ***REMOVED_DB*** is displayed.
db_id	oid	OID of the database that the backend is connected to.
eliminated_count	bigint	Number of plan cache evictions generated by the prepare statement.
plan_count	text	Total number of SQL execution plans, including LP, FQS, STREAM, and PGXC.
aplan_count	bigint	Number of Aplan execution times.
aplan_parse_time	bigint	Aplan parsing time.
aplan_execution_time	bigint	Aplan execution time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
hard_parse_reason	text	Hard parsing reason. The causes are as follows: • **first_execute**: number of hard parsing times for the first execution. • **cplan**: number of hard parsing times for generating a Cplan. • **gplan**: number of hard parsing times for generating a Gplan. • **aplan**: number of hard parsing times for generating an Aplan. • **revalid**: number of hard parsing times for rebuilding a plan after the plan becomes invalid. • **memory_control**: number of hard parsing times for rebuilding a plan after the plan is evicted by memory control. The preceding reasons are logically overlapped. For example, if the plan is executed for the first time and the Cplan is generated, the value of **hard_parse_reason** is **{"first_execute":1, "cplan":1, "gplan":0, "aplan":0, "revalid":0, "memory_control":0}**. If the value of **revalid** is not **0**, the cause and number of times that the plan expires are displayed, for example, "revalid":1{invalid reason: guc changed:1}.
sql_hash	bigint	Hash value of the current SQL statement. In the current schema, it is recorded as **0**.
plan_hash	bigint	Hash value of the current SQL plan. In the current schema, it is recorded as **0**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- SQL plan management does not support the distributed mode. Therefore, **sql_hash** and **plan_hash** do not take effect in the distributed mode.
- Statistics about statements in a stored procedure on a DN are aggregated only to the outermost statement. The statements called at the middle layer do not contain statistics about substatements.
- The following types of statements can be recorded in a stored procedure: **DML statements**, **dynamic statements**, **OPEN/FETCH cursor statements**, **nested subprograms**, **RETURN QUERY statements**, and PERFORM statements (statements that call a stored procedure using the PERFORM keyword or statements that call a stored procedure without **OUT** output parameters).
- When a dynamic statement is used as a loop variable, cursor, or RETURN QUERY, no record is generated for the dynamic statement.
- After a stored procedure is executed, if you clear the records in the **DBE_PERF.STATEMENT** table and execute the same stored procedure again, no new execution record is generated due to plan reuse.
- After user A executes a stored procedure, if another user executes the same stored procedure, no new execution record is generated due to plan reuse.
- The **CPU_TIME** value of the statement in the stored procedure is **0**.
- If an error is reported when a statement in a stored procedure is executed in the statistics reporting phase, the statistical columns recorded by the statement in DBE_PERF.STATEMENT may all be 0.

8.2.9.2 DBE_PERF.SUMMARY_STATEMENT

Obtains full information (including DNs) about the executed statements (normalized SQL statements) on each CN, as shown in **Table 8-109**.

Table 8-109 DBE_PERF.SUMMARY_STATEMENT columns

Name	Type	Description
node_name	name	Node name.
node_id	integer	Node ID (**node_id** in **pgxc_node**).
user_name	name	Username. If the user who runs the query statement does not exist, ***REMOVED_USER*** is displayed.
user_id	oid	OID of the user.
unique_sql_id	bigint	ID of the normalized SQL statement.
query	text	Normalized SQL statement. Note: The length is controlled by **instr_unique_sql_size**.
n_calls	bigint	Number of actual calls. The number of times that the FETCH statement in a stored procedure is triggered is the same as the **n_calls** increment of the statement executed by the cursor.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
min_elapse_time	bigint	Minimum execution time of the SQL statement in the kernel (unit: μs).
max_elapse_time	bigint	Maximum execution time of the SQL statement in the kernel (unit: μs).
total_elapse_time	bigint	Total execution time of the SQL statement in the kernel (unit: μs).
n_returned_rows	bigint	Number of rows in the result set returned by the SELECT statement.
n_tuples_fetched	bigint	Number of rows randomly scanned.
n_tuples_returned	bigint	Number of rows sequentially scanned.
n_tuples_inserted	bigint	Number of rows inserted.
n_tuples_updated	bigint	Number of rows updated.
n_tuples_deleted	bigint	Number of rows deleted.
n_blocks_fetched	bigint	Number of logical reads.
n_blocks_hit	bigint	Number of memory hits.
n_soft_parse	bigint	Number of soft parsing times. The value of **n_soft_parse** plus the value of **n_hard_parse** may be greater than the value of **n_calls** because the number of subqueries is not counted in the value of **n_calls**. (This is meaningless for non-DML statements.)
n_hard_parse	bigint	Number of hard parsing times. The value of **n_soft_parse** plus the value of **n_hard_parse** may be greater than the value of **n_calls** because the number of subqueries is not counted in the value of **n_calls**. The parsing of complex data types, for example, the smalldatetime type, also increases the number of parsing times. (This is meaningless for non-DML statements.)
db_time	bigint	Valid internal database time, which is accumulated if multiple threads are involved (unit: μs).
cpu_time	bigint	CPU time (unit: μs).
execution_time	bigint	Execution time in the executor (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
parse_time	bigint	SQL parsing time (unit: μs). (If an exception occurs during parsing, **parse_time** is not recorded.)
plan_time	bigint	SQL plan generation time (unit: μs). (If an exception occurs during plan generation, **plan_time** is not recorded.)
rewrite_time	bigint	SQL rewriting time (unit: μs). (If an exception occurs during rewriting, **rewrite_time** is not recorded.)
pl_execution_time	bigint	Execution time of PL/pgSQL (unit: μs).
pl_compilation_time	bigint	Compilation time of PL/pgSQL (unit: μs).
data_io_time	bigint	Time spent on I/O (unit: μs).
net_send_info	text	Network status of messages sent through a physical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, CNs communicate with each other, CNs communicate with clients, and CNs communicate with DNs through physical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: {"time":xxx, "n_calls":xxx, "size":xxx}.
net_recv_info	text	Network status of messages received through a physical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, CNs communicate with each other, CNs communicate with clients, and CNs communicate with DNs through physical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: {"time":xxx, "n_calls":xxx, "size":xxx}.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
net_stream_send_info	text	Network status of messages sent through a logical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, DNs of different shards communicate with each other through logical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: {"time":xxx, "n_calls":xxx, "size":xxx}.
net_stream_recv_info	text	Network status of messages received through a logical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, DNs of different shards communicate with each other through logical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: {"time":xxx, "n_calls":xxx, "size":xxx}.
last_updated	timestamp with time zone	Last time when the statement was updated.
sort_count	bigint	Sorting count.
sort_time	bigint	Sorting execution time (unit: μs), including the time of the sort operator and its sub-operators.
sort_mem_used	bigint	Size of work memory used during sorting (unit: KB).
sort_spill_count	bigint	Count of file writing when data is flushed to disks during sorting.
sort_spill_size	bigint	File size used when data is flushed to disks during sorting (unit: KB).
hash_count	bigint	Hashing count.
hash_time	bigint	Hashing duration (unit: μs).
hash_mem_used	bigint	Size of work memory used during hashing (unit: KB).
hash_spill_count	bigint	Count of file writing when data is flushed to disks during hashing.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
hash_spill_size	bigint	File size used when data is flushed to disks during hashing (unit: KB).
parent_unique_sql_id	bigint	Unique ID of the parent SQL statement. The value is **0** for a non-stored procedure substatement.
lock_wait_time	bigint	SQL lock waiting time (unit: μs).
total_used_memory	bigint	Total memory used by normalized statements, in MB.
max_used_memory	integer	Maximum memory usage of a single statement, in MB.
min_used_memory	integer	Minimum memory usage of a single statement, in MB.
db_name	name	Name of the database connected to the backend. If the database where the query statement is executed does not exist, ***REMOVED_DB*** is displayed.
db_id	oid	OID of the database that the backend is connected to.
eliminated_count	bigint	Number of plan cache evictions generated by the prepared statement.
plan_count	text	Total number of SQL execution plans, including LP, FQS, STREAM, and PGXC.
aplan_count	bigint	Number of Aplan execution times.
aplan_parse_time	bigint	Aplan parsing time.
aplan_execution_time	bigint	Aplan execution time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
hard_parse_reason	text	Hard parsing reason. The causes are as follows: • **first_execute**: number of hard parsing times for the first execution. • **cplan**: number of hard parsing times for generating a Cplan. • **gplan**: number of hard parsing times for generating a Gplan. • **aplan**: number of hard parsing times for generating an Aplan. • **revalid**: number of hard parsing times for rebuilding a plan after the plan becomes invalid. • **memory_control**: number of hard parsing times for rebuilding a plan after the plan is evicted by memory control. The preceding reasons are logically overlapped. For example, if the plan is executed for the first time and the Cplan is generated, the value of **hard_parse_reason** is **{"first_execute":1, "cplan":1, "gplan":0, "aplan":0, "revalid":0, "memory_control":0}**. If the value of **revalid** is not 0, the reason and number of times that the plan becomes invalid are displayed, for example, **"revalid":1{invalid reason: guc changed:1}**.
sql_hash	bigint	Hash value of the current SQL statement. In the current schema, it is recorded as **0**.
plan_hash	bigint	Hash value of the current SQL plan. In the current schema, it is recorded as **0**.

8.2.9.3 DBE_PERF.STATEMENT_COUNT

Displays statistics about five types of running statements (SELECT, INSERT, UPDATE, DELETE, and MERGE INTO) as well as DDL, DML, and DCL statements on the current node of the database, as described in **Table 8-110**.

NOTE

An administrator can query the STATEMENT_COUNT view to view the statistics of all users on the current node. When the cluster or node is restarted, the statistics are cleared and the counting restarts. The system counts when a node receives a query, including a query inside the cluster. For example, if the CN receives a query that contains multiple subqueries, the subqueries are counted separately when they are delivered to a single DN or multiple DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-110 DBE_PERF.STATEMENT_COUNT columns

Name	Type	Description
node_name	text	Node name.
user_name	text	Username.
select_count	bigint	Statistical result of SELECT statements.
update_count	bigint	Statistical result of UPDATE statements.
insert_count	bigint	Statistical result of INSERT statements.
delete_count	bigint	Statistical result of DELETE statements.
mergeinto_count	bigint	Statistical result of MERGE INTO statements.
ddl_count	bigint	Number of DDL statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including TRUNCATE, CLUSTER, VACUUM, ANALYZE, CREATE (excluding CREATE ROLE and CREATE USER), DROP (excluding DROP ROLE and DROP USER), ALTER (excluding ALTER ROLE, ALTER USER, ALTER DEFAULT PRIVILEGES, and ALTER SESSION), REINDEX, COMMENT, BARRIER, SECURITY LABEL ON, PURGE, TIMECAPSULE, REFRESH, IMPDP, and EXPDP statements.
dml_count	bigint	Number of DML statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including SELECT, UPDATE, INSERT, DELETE, MERGE, PREPARE, EXPLAIN, LOCK, COPY, ANONYMOUS, DELTA, EXECUTE, MOVE, FETCH, CLOSE, DEALLOCATE, DECLARE, CURSOR, and LOAD statements.
dcl_count	bigint	Number of DCL statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including SHOW, CREATE ROLE, CREATE USER, DROP ROLE, DROP USER, ALTER ROLE, ALTER USER, ALTER DEFAULT PRIVILEGES, GRANT, REVOKE, REASSIGN, SET, RESET, and ALTER SESSION statements.
total_select_elapse	bigint	Total time spent on SELECT statements (unit: μs)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
avg_select_elapse	bigint	Average time spent on SELECT statements (unit: μs).
max_select_elapse	bigint	Maximum time spent on SELECT statements (unit: μs).
min_select_elapse	bigint	Minimum time spent on SELECT statements (unit: μs).
total_update_elapse	bigint	Total time spent on UPDATE statements (unit: μs).
avg_update_elapse	bigint	Average time spent on UPDATE statements (unit: μs).
max_update_elapse	bigint	Maximum time spent on UPDATE statements (unit: μs).
min_update_elapse	bigint	Minimum time spent on UPDATE statements (unit: μs).
total_insert_elapse	bigint	Total time spent on INSERT statements (unit: μs).
avg_insert_elapse	bigint	Average time spent on INSERT statements (unit: μs).
max_insert_elapse	bigint	Maximum time spent on INSERT statements (unit: μs).
min_insert_elapse	bigint	Minimum time spent on INSERT statements (unit: μs).
total_delete_elapse	bigint	Total time spent on DELETE statements (unit: μs).
avg_delete_elapse	bigint	Average time spent on DELETE statements (unit: μs).
max_delete_elapse	bigint	Maximum time spent on DELETE statements (unit: μs).
min_delete_elapse	bigint	Minimum time spent on DELETE statements (unit: μs).
dbid	oid	ID of the database to which the five types of statements (SELECT, INSERT, UPDATE, DELETE, and MERGE INTO) belong.
user_dml_count	bigint	Number of DML statements executed by users. The value of **user_dml_count** is equal to the value of **dml_count** minus the value of **bg_dml_count**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
bg_dml_count	bigint	Number of DML statements executed by background threads (excluding WORKER and THREADPOOL_WORKER) in the database in all scenarios. **NOTE** The DML statements executed by the current stored procedure cannot be counted.

8.2.9.4 DBE_PERF.GLOBAL_STATEMENT_COUNT

Displays statistics about five types of running statements (SELECT, INSERT, UPDATE, DELETE, and MERGE INTO) as well as DDL, DML, and DCL statements on each node of the database, as described in **Table 8-111**.

Table 8-111 DBE_PERF.GLOBAL_STATEMENT_COUNT columns

Name	Type	Description
node_name	text	Node name.
user_name	text	Username.
select_count	bigint	Statistical result of SELECT statements.
update_count	bigint	Statistical result of UPDATE statements.
insert_count	bigint	Statistical result of INSERT statements.
delete_count	bigint	Statistical result of DELETE statements.
mergeinto_count	bigint	Statistical result of MERGE INTO statements.
ddl_count	bigint	Number of DDL statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including TRUNCATE, CLUSTER, VACUUM, ANALYZE, CREATE (excluding CREATE ROLE and CREATE USER), DROP (excluding DROP ROLE and DROP USER), ALTER (excluding ALTER ROLE, ALTER USER, ALTER DEFAULT PRIVILEGES, and ALTER SESSION), REINDEX, COMMENT, BARRIER, SECURITY LABEL ON, PURGE, TIMECAPSULE, REFRESH, IMPDP, and EXPDP statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
dml_count	bigint	Number of DML statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including SELECT, UPDATE, INSERT, DELETE, MERGE, PREPARE, EXPLAIN, LOCK, COPY, ANONYMOUS, DELTA, EXECUTE, MOVE, FETCH, CLOSE, DEALLOCATE, DECLARE, CURSOR, and LOAD statements.
dcl_count	bigint	Number of DCL statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including SHOW, CREATE ROLE, CREATE USER, DROP ROLE, DROP USER, ALTER ROLE, ALTER USER, ALTER DEFAULT PRIVILEGES, GRANT, REVOKE, REASSIGN, SET, RESET, and ALTER SESSION statements.
total_select_elapse	bigint	Total time spent on SELECT statements (unit: μs)
avg_select_elapse	bigint	Average time spent on SELECT statements (unit: μs).
max_select_elapse	bigint	Maximum time spent on SELECT statements (unit: μs).
min_select_elapse	bigint	Minimum time spent on SELECT statements (unit: μs).
total_update_elapse	bigint	Total time spent on UPDATE statements (unit: μs).
avg_update_elapse	bigint	Average time spent on UPDATE statements (unit: μs).
max_update_elapse	bigint	Maximum time spent on UPDATE statements (unit: μs).
min_update_elapse	bigint	Minimum time spent on UPDATE statements (unit: μs).
total_insert_elapse	bigint	Total time spent on INSERT statements (unit: μs).
avg_insert_elapse	bigint	Average time spent on INSERT statements (unit: μs).
max_insert_elapse	bigint	Maximum time spent on INSERT statements (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
min_insert_elapse	bigint	Minimum time spent on INSERT statements (unit: μs).
total_delete_elapse	bigint	Total time spent on DELETE statements (unit: μs).
avg_delete_elapse	bigint	Average time spent on DELETE statements (unit: μs).
max_delete_elapse	bigint	Maximum time spent on DELETE statements (unit: μs).
min_delete_elapse	bigint	Minimum time spent on DELETE statements (unit: μs).
dbid	oid	ID of the database to which the five types of statements (SELECT, INSERT, UPDATE, DELETE, and MERGE INTO) belong.
user_dml_count	bigint	Number of DML statements executed by users. The value of **user_dml_count** is equal to the value of **dml_count** minus the value of **bg_dml_count**.
bg_dml_count	bigint	Number of DML statements executed by background threads (excluding WORKER and THREADPOOL_WORKER) in the database in all scenarios. **NOTE** The DML statements executed by the current stored procedure cannot be counted.

8.2.9.5 DBE_PERF.SUMMARY_STATEMENT_COUNT

Displays the summary statistics of five types of statements (SELECT, INSERT, UPDATE, DELETE, and MERGE INTO) and DDL, DML, and DCL statements executed on each database node, as shown in **Table 8-112**.

Table 8-112 DBE_PERF.SUMMARY_STATEMENT_COUNT columns

Name	Type	Description
user_name	text	Username.
select_count	numeric	Statistical result of SELECT statements.
update_count	numeric	Statistical result of UPDATE statements.
insert_count	numeric	Statistical result of INSERT statements.
delete_count	numeric	Statistical result of DELETE statements.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
mergeinto_count	numeric	Statistical result of MERGE INTO statements.
ddl_count	numeric	Number of DDL statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including TRUNCATE, CLUSTER, VACUUM, ANALYZE, CREATE (excluding CREATE ROLE and CREATE USER), DROP (excluding DROP ROLE and DROP USER), ALTER (excluding ALTER ROLE, ALTER USER, ALTER DEFAULT PRIVILEGES, and ALTER SESSION), REINDEX, COMMENT, BARRIER, SECURITY LABEL ON, PURGE, TIMECAPSULE, REFRESH, IMPDP, and EXPDP statements.
dml_count	numeric	Number of DML statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including SELECT, UPDATE, INSERT, DELETE, MERGE, PREPARE, EXPLAIN, LOCK, COPY, ANONYMOUS, DELTA, EXECUTE, MOVE, FETCH, CLOSE, DEALLOCATE, DECLARE, CURSOR, and LOAD statements.
dcl_count	numeric	Number of DCL statements. The statistics include the SQL statements executed by users and the SQL statements executed by database background threads, including SHOW, CREATE ROLE, CREATE USER, DROP ROLE, DROP USER, ALTER ROLE, ALTER USER, ALTER DEFAULT PRIVILEGES, GRANT, REVOKE, REASSIGN, SET, RESET, and ALTER SESSION statements.
total_select_elapse	numeric	Total time spent on SELECT statements (unit: μs)
avg_select_elapse	bigint	Average time spent on SELECT statements (unit: μs).
max_select_elapse	bigint	Maximum time spent on SELECT statements (unit: μs).
min_select_elapse	bigint	Minimum time spent on SELECT statements (unit: μs).
total_update_elapse	numeric	Total time spent on UPDATE statements (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
avg_update_elapse	bigint	Average time spent on UPDATE statements (unit: μs).
max_update_elapse	bigint	Maximum time spent on UPDATE statements (unit: μs).
min_update_elapse	bigint	Minimum time spent on UPDATE statements (unit: μs).
total_insert_elapse	numeric	Total time spent on INSERT statements (unit: μs).
avg_insert_elapse	bigint	Average time spent on INSERT statements (unit: μs).
max_insert_elapse	bigint	Maximum time spent on INSERT statements (unit: μs).
min_insert_elapse	bigint	Minimum time spent on INSERT statements (unit: μs).
total_delete_elapse	numeric	Total time spent on DELETE statements (unit: μs).
avg_delete_elapse	bigint	Average time spent on DELETE statements (unit: μs).
max_delete_elapse	bigint	Maximum time spent on DELETE statements (unit: μs).
min_delete_elapse	bigint	Minimum time spent on DELETE statements (unit: μs).
dbid	oid	ID of the database to which the five types of statements (SELECT, INSERT, UPDATE, DELETE, and MERGE INTO) belong.
user_dml_count	bigint	Number of DML statements executed by users. The value of **user_dml_count** is equal to the value of **dml_count** minus the value of **bg_dml_count**.
bg_dml_count	bigint	Number of DML statements executed by background threads (excluding WORKER and THREADPOOL_WORKER) in the database in all scenarios. **NOTE** The DML statements executed by the current stored procedure cannot be counted.

8.2.9.6 DBE_PERF.STATEMENT_RESPONSETIME_PERCENTILE

Obtains the response times of 80% and 95% SQL statements in the cluster, as described in **Table 8-113**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-113 DBE_PERF.STATEMENT_RESPONSETIME_PERCENTILE columns

Name	Type	Description
p80	bigint	Response time of 80% SQL statements in the cluster (unit: μs)
p95	bigint	Response time of 95% SQL statements in the cluster (unit: μs)

8.2.9.7 DBE_PERF.STATEMENT_HISTORY

Displays information about statements executed on the current node. The result can be queried only in the system database but cannot be queried in the user database, as described in **Table 8-114**.

Table 8-114 DBE_PERF.STATEMENT_HISTORY columns

Name	Type	Description
db_name	name	Database name.
schema_name	name	Schema name.
origin_node	integer	Node name.
user_name	name	Username.
application_name	text	Name of the application that sends a request.
client_addr	text	IP address of the client that sends a request.
client_port	integer	Port number of the client that sends a request.
unique_query_id	bigint	ID of the normalized SQL statement.
debug_query_id	bigint	ID of the unique SQL statement. Some statements are not unique. For example, the value of **debug_query_id** in the Parse packet, DCL statements, and TCL statements is **0**. Statements in a stored procedure use the same ID.
query	text	Normalized SQL (available only on CNs).
start_time	timestamp with time zone	Time when a statement starts.
finish_time	timestamp with time zone	Time when a statement ends.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
slow_sql_threshold	bigint	Standard for slow SQL statement execution.
transaction_id	bigint	Transaction ID.
thread_id	bigint	ID of an execution thread.
session_id	bigint	Session ID of a user.
n_soft_parse	bigint	Number of soft parsing times. The value of **n_soft_parse** plus the value of **n_hard_parse** may be greater than the value of **n_calls** because the number of subqueries is not counted in the value of **n_calls**. (This is meaningless for non-DML statements.)
n_hard_parse	bigint	Number of hard parsing times. The value of **n_soft_parse** plus the value of **n_hard_parse** may be greater than the value of **n_calls** because the number of subqueries is not counted in the value of **n_calls**. The parsing of complex data types, for example, smalldatetime, also increases the number of parsing times. (This is meaningless for non-DML statements.)
query_plan	text	Statement execution plan.
n_returned_rows	bigint	Number of rows in the result set returned by the SELECT statement.
n_tuples_fetched	bigint	Number of rows randomly scanned, that is, number of times that TABLE ACCESS BY INDEX ROWID is performed. The process of TABLE ACCESS BY INDEX ROWID is as follows: When you use indexes to query data, if the columns to be queried are not in the indexes, the database engine needs to access the main table again through the primary key or unique identifier in the indexes to obtain the required complete data rows.
n_tuples_returned	bigint	Number of rows sequentially scanned, that is, the sum of the number of live rows obtained by sequential scans and the number of index entries returned by index scans.
n_tuples_inserted	bigint	Number of rows inserted.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
n_tuples_updated	bigint	Number of rows updated.
n_tuples_deleted	bigint	Number of rows deleted.
n_blocks_fetched	bigint	Number of buffer block accesses, indicating the total number of blocks read from the buffer and disk.
n_blocks_hit	bigint	Number of buffer block hits, indicating the number of blocks read from the buffer.
db_time	bigint	Valid DB time, which is accumulated if multiple threads are involved (unit: μs).
cpu_time	bigint	Time consumed on the CPU (unit: μs).
execution_time	bigint	Execution time in the executor (unit: μs).
parse_time	bigint	SQL parsing time (unit: μs). (If an exception occurs during parsing, **parse_time** is not recorded.)
plan_time	bigint	SQL plan generation time (unit: μs). (If an exception occurs during plan generation, **plan_time** is not recorded.)
rewrite_time	bigint	SQL rewriting time (unit: μs). (If an exception occurs during rewriting, **rewrite_time** is not recorded.)
pl_execution_time	bigint	Execution time of PL/pgSQL (unit: μs).
pl_compilation_time	bigint	Compilation time of PL/pgSQL (unit: μs).
data_io_time	bigint	Time spent on I/O (unit: μs).
net_send_info	text	Network status of messages sent through a physical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, CNs communicate with each other, CNs communicate with clients, and CNs communicate with DNs through physical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: **{"time":***xxx***, "n_calls":***xxx***, "size":***xxx***}**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
net_recv_info	text	Network status of messages received through a physical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, CNs communicate with each other, CNs communicate with clients, and CNs communicate with DNs through physical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: **{"time":***xxx***, "n_calls":***xxx***, "size":***xxx***}**.
net_stream_send_info	text	Network status of messages sent through a logical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, DNs of different shards communicate with each other through logical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: **{"time":***xxx***, "n_calls":***xxx***, "size":***xxx***}**.
net_stream_recv_info	text	Network status of messages received through a logical connection, including the time (unit: μs), number of calls, and throughput (unit: byte). In a distributed database, DNs of different shards communicate with each other through logical connections. This column can be used to analyze the network overhead of SQL statements in a distributed system. Example: **{"time":***xxx***, "n_calls":***xxx***, "size":***xxx***}**.
lock_count	bigint	Number of locks.
lock_time	bigint	Time required for locking.
lock_wait_count	bigint	Number of lock waits.
lock_wait_time	bigint	Time required for lock waiting.
lock_max_count	bigint	Maximum number of locks.
lwlock_count	bigint	Number of lightweight locks (reserved).
lwlock_wait_count	bigint	Number of lightweight lock waits.
lwlock_time	bigint	Time required for lightweight locking (reserved).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
lwlock_wait_time	bigint	Time required for lightweight lock wait.
details	bytea	List of wait events, statement lock events, and GSTrace function running time. When the value of the record level is greater than or equal to L0, the list of wait events starts to be recorded. It displays statistics about wait events on the current node. For key events in the kernel, see **Table 8-192**. Alternatively, view the list of all events in the system in the wait_event_info view. For details about the impact of each transaction lock on services, see **LOCK**. In the GStrace full-link tracing function, the **function** field of GStrace indicates the function name, and the parent-child relationship of the function is expressed in a tree structure. The **duration** field of GStrace indicates the function running time, in microseconds. When the record level is **L2**, the list of statement lock events starts to be recorded in chronological order. The number of recorded lock events is determined by the **track_stmt_details_size** parameter. This column is in binary format and needs to be read by using the parsing function pg_catalog.statement_detail_decode. For details, see **Table 1-204**. Events include: • LOCK_START: Start locking. • LOCK_END: Complete locking. • LOCK_WAIT_START: Start lock waiting. • LOCK_WAIT_END: Complete lock waiting. • LOCK_RELEASE: Release the lock. • LWLOCK_WAIT_START: Start lightweight lock waiting. • LWLOCK_WAIT_END: Complete lightweight lock waiting.
is_slow_sql	boolean	Specifies whether the SQL statement is a slow SQL statement.
trace_id	text	Driver-specific trace ID, which is associated with an application request.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
advise	text	Risk information that may cause the SQL statement to be a slow SQL statement. **NOTE** If the statement is executed in SQLBypass mode (you can run the explain command to view the plan. For details, see **enable_opfusion**), there is no slow SQL risk and the information in this field may be missing.
parent_unique_sql_id	bigint	Normalized SQL ID of the outer SQL statement. For statements executed in a stored procedure, the value is the normalized SQL ID of the statement that calls the stored procedure. For statements outside the stored procedure, the value is **0**.
finish_status	text	Statement completion status. The default value is **normal**. **cancelled** indicates cancellation or termination. **error** indicates that the operation ends due to other exceptions except **cancelled**.
used_memory	integer	Memory usage of a single statement (unit: MB).
lock_max_local_count	bigint	Maximum number of locks that can be held by the locallock mechanism.
lock_max_fastpath_count	bigint	Maximum number of locks that can be held by the fastpath mechanism.
lock_max_global_count	bigint	Maximum number of locks that can be held by the main lock table.
driver_start_time	timestamp with time zone	Time when the JDBC driver starts the execution.
driver_wait_response	timestamp with time zone	Time when the JDBC driver starts to wait for the data returned by the database after sending a packet.
driver_finish_time	timestamp with time zone	Time when the JDBC driver receives the data returned by the database.
driver_info	json	Time when the JDBC driver sends and receives packets. After the **track_stmt_stat_level** parameter is modified, the modification takes effect only for new sessions.
kernel_info	json	Time when database packets are sent and received.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
adaptive_join_states	text	Specifies the chosen connection mode of the adaptive connection operator. The format is **[plan_node_id, use_hashjoin]**. The first digit indicates the corresponding **plan_node_id**, and the second digit specifies whether to use HashJoin. The value **1** indicates yes, and the value **0** indicates no.
aplan_count	bigint	Number of Aplan execution times.
aplan_parse_time	bigint	Aplan parsing time.
aplan_execution_time	bigint	Aplan execution time.
sql_hash	bigint	Hash value of the current SQL statement.
plan_hash	bigint	Hash value of the current SQL plan.
plan_hash_prev	bigint	For a generic plan, the value is the hash value of the previous plan.
relids	text	pg_class.oid involving tables in the statement. Separate multiple tables with commas (,).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

- If the values in **track_stmt_stat_level** for controlling full SQL statements and slow SQL statements are not **OFF**, the **is_slow_sql** field is recorded as **true** for an SQL statement that exceeds the **log_min_duration_statement** threshold. Other fields are recorded based on the levels in **track_stmt_stat_level**.
- Adaptive join does not support distributed mode. Therefore, **adaptive_join_states** does not take effect in distributed mode.
- SQL plan management does not support the distributed mode. Therefore, **sql_hash**, **plan_hash**, and **plan_hash_prev** do not take effect in the distributed mode.
- Lock-related metrics of sub-statements in a distributed stored procedure are not aggregated to the top-level statements.
- If a query plan fails to be recorded, the cause of the failure is displayed in the **query_plan** column. The following are some examples:
  - In the LightProxy scenario, statements are pushed down, and the plan is recorded on DNs instead of CNs. Message: "The current plan is lightproxy. Please check the query_plan on the DN node."
  - In the Stream scenario, the plan is pushed down and recorded on CNs instead of DNs. Message: "The current plan is streaming. Please check the query_plan on the CN node."
  - For COMMIT statements that are automatically committed by DML statements, related plans are not recorded. Message: "The current record is a transaction commit of a DML statement. Please check the plan of this DML statement."
  - Plans are not recorded for non-DML/DQL statements. Message: "Do not record plans for non-DML/DQL statements." For DDL statements, in a cluster with multiple CNs, query_plan records information about interactions between nodes.
  - Plans are not recorded for substatements that are not adapted or have no plan in a stored procedure, such as simple expressions, OPEN CURSOR, CLOSE, and BYPASS path statements. Plans for substatements executed in a stored procedure are not recorded on DNs. Message: "Statement has no plan or is unsupported for plan recording."
  - For SQL statements that have entered the PARSE phase, no plan has been generated and therefore the statements are not recorded. Message: "This statement is a PARSE statement and has not yet generated a query plan."
  - When the system checks whether **query_plan** needs to be reported, the SQL statement execution is not slow, but the overall execution of the SQL statements is slow due to subsequent tasks such as cleanup. Message: "Check the reported query_plan time is 2026-01-01 08:05:01.735619+08, at this time the SQL is not slow."
  - If the statement execution status is abnormal, the plan may fail to be recorded. In this case, check the logs to analyze the cause. Message: "An error occurred during statement execution. Please check the log for analysis."
  - For other scenarios where the recording fails, check the logs to analyze the cause. Message: "The query plan failed to be recorded. Please check the log analysis."

8.2.9.8 DBE_PERF.GLOBAL_STATEMENT_COMPLEX_HISTORY

Displays workload management information about completed jobs executed on each node, as described in **Table 8-115**.

Table 8-115 DBE_PERF.GLOBAL_STATEMENT_COMPLEX_HISTORY columns

Name	Type	Description
datid	oid	OID of the database that the backend is connected to.
dbname	text	Name of the database that the backend is connected to.
schemaname	text	Schema name.
nodename	text	Name of the CN where the statement is executed.
username	text	Username used for connecting to the backend.
application_name	text	Name of the application connected to the backend.
client_addr	inet	IP address of the client connected to the backend. If this column is **NULL**, it indicates either the client is connected via a Unix socket on the server or this is an internal thread, such as **AUTOVACUUM**.
client_hostname	text	Host name of the connected client, as reported by a reverse DNS lookup of **client_addr**. This column will be non-null only for IP connections and only when **log_hostname** is enabled.
client_port	integer	TCP port number that the client uses for communication with this backend (**–1** if a Unix socket is used).
query_band	text	Job type, which is specified by the GUC parameter **query_band**. The default value is a null string.
block_time	bigint	Duration that the statement is blocked before being executed, including the statement parsing and optimization duration (unit: ms).
start_time	timestamp with time zone	Time when the statement execution starts.
finish_time	timestamp with time zone	Time when the statement execution ends.
duration	bigint	Execution time of the statement (unit: ms).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
estimate_total_time	bigint	Estimated execution time of the statement (unit: ms).
status	text	Final statement execution status. The value can be **finished** (normal) or **aborted** (abnormal).
abort_info	text	Exception information displayed if the final statement execution status is **aborted**.
resource_pool	text	Resource pool used by the user.
control_group	text	Cgroup used by the statement.
estimate_memory	integer	Estimated memory used by the statement.
min_peak_memory	integer	Minimum peak memory of the statement among all DNs (unit: MB).
max_peak_memory	integer	Maximum peak memory of the statement among all DNs (unit: MB).
average_peak_memory	integer	Average memory usage during statement execution (unit: MB).
memory_skew_percent	integer	Memory usage skew of the statement among each DN.
spill_info	text	Statement spill information on all DNs. • **None**: No data is spilled to disks on all DNs. • **All**: Data is spilled to disks on all DNs. • **[*a*:*b*]**: The statement has been spilled to disks on *a* of *b* DNs.
min_spill_size	integer	Minimum spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
max_spill_size	integer	Maximum spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
average_spill_size	integer	Average spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
spill_skew_percent	integer	DN spill skew when a spill occurs.
min_dn_time	bigint	Minimum execution time of the statement across all DNs (unit: ms).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
max_dn_time	bigint	Maximum execution time of the statement across all DNs (unit: ms).
average_dn_time	bigint	Average execution time of the statement across all DNs (unit: ms).
dntime_skew_percent	integer	Execution time skew of the statement among each DN.
min_cpu_time	bigint	Minimum CPU time of the statement across all DNs (unit: ms).
max_cpu_time	bigint	Maximum CPU time of the statement across all DNs (unit: ms).
total_cpu_time	bigint	Total CPU time of the statement across all DNs (unit: ms).
cpu_skew_percent	integer	CPU time skew of the statement among DNs.
min_peak_iops	integer	Minimum peak IOPS of the statement across all DNs (unit: 10 thousands per second).
max_peak_iops	integer	Maximum peak IOPS of the statement across all DNs (unit: 10 thousands per second).
average_peak_iops	integer	Average peak IOPS of the statement across all DNs (unit: 10 thousands per second).
iops_skew_percent	integer	I/O skew across DNs.
warning	text	Warning. The following warnings are displayed: ● Spill file size larger than 256MB ● Broadcast size larger than 100MB ● Early spill ● Spill times is greater than 3 ● Spill on memory adaptive ● Hash table conflict
queryid	bigint	Internal query ID used for statement execution.
query	text	Statement executed.
query_plan	text	Execution plan of the statement.
cpu_top1_node_name	text	Name of the node with the highest CPU usage.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
cpu_top2_node_name	text	Name of the node with the second highest CPU usage.
cpu_top3_node_name	text	Name of the node with the third highest CPU usage.
cpu_top4_node_name	text	Name of the node with the fourth highest CPU usage.
cpu_top5_node_name	text	Name of the node with the fifth highest CPU usage.
mem_top1_node_name	text	Name of the node with the highest memory usage.
mem_top2_node_name	text	Name of the node with the second highest memory usage.
mem_top3_node_name	text	Name of the node with the third highest memory usage.
mem_top4_node_name	text	Name of the node with the fourth highest memory usage.
mem_top5_node_name	text	Name of the node with the fifth highest memory usage.
cpu_top1_value	bigint	1st CPU usage.
cpu_top2_value	bigint	2nd CPU usage.
cpu_top3_value	bigint	3rd CPU usage.
cpu_top4_value	bigint	4th CPU usage.
cpu_top5_value	bigint	5th CPU usage.
mem_top1_value	bigint	1st memory usage.
mem_top2_value	bigint	2nd memory usage.
mem_top3_value	bigint	3rd memory usage.
mem_top4_value	bigint	4th memory usage.
mem_top5_value	bigint	5th memory usage.
top_mem_dn	text	Top N memory usage.
top_cpu_dn	text	Top N CPU usage.
node_group	text	Node group of the user to which the statement belongs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.9.9 DBE_PERF.GLOBAL_STATEMENT_COMPLEX_HISTORY_TABLE

Displays workload management information about completed jobs executed on each node. Data is dumped from the kernel to this system catalog. If the GUC parameter **enable_resource_record** is set to **on**, the system imports records from **DBE_PERF.GLOBAL_STATEMENT_COMPLEX_HISTORY** to this system catalog every 3 minutes. You are advised not to enable this function because it occupies storage space and affects performance.

For details about the columns, see **DBE_PERF.GLOBAL_STATEMENT_COMPLEX_HISTORY**.

8.2.9.10 DBE_PERF.STATEMENT_WLMSTAT_COMPLEX_RUNTIME

Displays workload management information about ongoing jobs executed by the current user, as described in **Table 8-116**.

Table 8-116 DBE_PERF.STATEMENT_WLMSTAT_COMPLEX_RUNTIME columns

Name	Type	Description
datid	oid	OID of the database that this backend is connected to.
datname	name	Name of the database that the backend is connected to.
threadid	bigint	Backend thread ID.
processid	integer	PID of the backend thread.
usesysid	oid	OID of the user logged in to the backend.
appname	text	Name of the application connected to the backend.
usename	name	Username for logging in to the backend.
priority	bigint	Priority of Cgroup where the statement is located.
attribute	text	Statement attribute: • **Ordinary**: default attribute of a statement before it is parsed by the database. • **Simple**: simple statement. • **Complicated**: complicated statement. • **Internal**: internal statement of the database.
block_time	bigint	Pending duration of the statement by now, in seconds.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
elapsed_time	bigint	Actual execution duration of the statement by now, in seconds.
total_cpu_time	bigint	Total CPU usage duration of the statement on the DN in the last period, in seconds.
cpu_skew_percent	integer	CPU usage inclination ratio of the statement on the DN in the last period.
statement_mem	integer	Memory used for executing the statement (reserved column).
active_points	integer	Number of concurrently active points occupied by the statement in the resource pool.
dop_value	integer	DOP value obtained by the statement from the resource pool.
control_group	text	Cgroup currently used by the statement.
status	text	Statement status, which can be: • **pending**: waiting to be executed. • **running**: being executed. • **finished**: finished normally. (If **enqueue** is set to **StoredProc** or **Transaction**, this state indicates that only a part of jobs in the statement has been executed. This state persists until this statement is complete.) • **aborted**: terminated unexpectedly. • **active**: a normal state except for those above. • **unknown**: unknown.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
enqueue	text	Queuing status of the statement, including: • **Global**: queuing in the global queue. • **Respool**: queuing in the resource pool queue. • **CentralQueue**: queuing on the CCN. • **Transaction**: queuing in a transaction block. • **StoredProc**: queuing in a stored procedure. • **None**: not in a queue. • **Forced None**: (transaction block statement or stored procedure statement) being forcibly executed because the statement waiting time exceeds the specified value.
resource_pool	name	Resource pool where the statement is located.
query	text	Latest query at the backend. If **state** is **active**, this column shows the executing query. In all other states, it shows the last query that was executed.

8.2.9.11 DBE_PERF.SESSION_SQL_MEMORY

Displays the memory usage of the statements being executed by the current user on the current node, as described in **Table 8-117**.

Table 8-117 DBE_PERF.SESSION_SQL_MEMORY columns

Name	Type	Description
pid	bigint	Thread ID.
sessionid	bigint	Session ID.
unique_sql_id	bigint	Unique SQL statement ID.
query_id	bigint	Debug query ID of a statement.
query	text	Current running statement of the session.
peak_used_memory	integer	Peak memory usage of the statement, in MB.
current_used_memory	integer	Current memory usage of the statement, in MB.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.9.12 DBE_PERF.GLOBAL_SESSION_SQL_MEMORY

Displays the memory usage of the statements being executed by the current user on each node, as shown in **Table 8-118**.

Table 8-118 DBE_PERF.GLOBAL_SESSION_SQL_MEMORY columns

Name	Type	Description
node_name	text	Name of the node in the current cluster.
pid	bigint	Thread ID.
sessionid	bigint	Session ID.
unique_sql_id	bigint	Unique SQL statement ID.
query_id	bigint	Debug query ID of a statement.
query	text	Current running statement of the session.
peak_used_memory	integer	Peak memory usage of the statement, in MB.
current_used_memory	integer	Current memory usage of the statement, in MB.

8.2.9.13 DBE_PERF.GS_PLANCACHE_STAT

Displays the parameterized statements of all sessions in the current process, as shown in **Table 8-119**. Only users with the MONADMIN permission can query this view.

NOTE

The same statement may generate different plans under the influence of GUC parameters, resulting in multiple occurrences of the statement in the view.

Table 8-119 DBE_PERF.GS_PLANCACHE_STAT columns

Name	Type	Description
pid	bigint	Background thread ID.
sessionid	bigint	Current session ID.
global_sessionid	text	Global session ID.
database_id	oid	Database ID.
parameterized_sql	text	Template SQL statement after parameterization.
saved_time	timestamp with time zone	Saves the timestamp when the template SQL statement is cached.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
last_used_time	timestamp with time zone	Saves the timestamp when the template SQL statement is last accessed.
parameter_types	regtype[]	Expected parameter type of the template SQL statement.
search_path	text	Search Path.
used_count	bigint	Number of times that the template SQL statement is used.
is_deleted	bool	Specifies whether the plan cache of the SQL statement has been evicted by memory control.
size	bigint	Memory size occupied by the SQL statement cache plan, in bytes.
valid	bool	Specifies whether the SQL statement cache plan is valid.
is_parameterized	bool	Specifies whether the SQL statement is parameterized.
opcache_reuse_cnt	integer	Number of times that the execution cache is reused.
opcache_mem_size	bigint	Memory size occupied by the execution cache.

8.2.9.14 DBE_PERF.GS_DIST_PLANCACHE_STAT

Displays the parameterized statement information of all processes (including the standby node), as shown in **Table 8-120**. Only users with the MONADMIN permission can query this view.

NOTE

Different plans may be generated for the same statement affected by GUC parameters. The statement is displayed multiple times in the view.

Table 8-120 DBE_PERF.GS_DIST_PLANCACHE_STAT columns

Name	Type	Description
node_name	name	Node name.
pid	bigint	Background thread ID.
sessionid	bigint	Current session ID.
global_sessionid	text	Global session ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
database_id	oid	Database ID.
parameterized_sql	text	Template SQL statement after parameterization.
saved_time	timestamp with time zone	Saves the timestamp when the template SQL statement is cached.
last_used_time	timestamp with time zone	Saves the timestamp when the template SQL statement is last accessed.
parameter_types	regtype[]	Expected parameter type of the template SQL statement.
search_path	text	Search Path.
used_count	bigint	Number of times that the template SQL statement is used.
is_deleted	bool	Specifies whether the plan cache of the SQL statement has been evicted by memory control.
size	bigint	Memory size occupied by the SQL statement cache plan, in bytes.
valid	bool	Specifies whether the SQL statement cache plan is valid.

8.2.10 Cache and I/O

8.2.10.1 DBE_PERF.STATIO_USER_TABLES

Displays I/O status information about all user relationship tables in the namespace, as described in **Table 8-121**.

Table 8-121 DBE_PERF.STATIO_USER_TABLES columns

Name	Type	Description
relid	oid	Table OID.
schemaname	name	Name of the schema that contains the table.
relname	name	Table name.
heap_blks_read	bigint	Number of disk blocks read from the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
heap_blks_hit	bigint	Number of cache hits in the table.
idx_blks_read	bigint	Number of disk blocks read from indexes in the table.
idx_blks_hit	bigint	Number of cache hits in indexes in the table.
toast_blks_read	bigint	Number of disk blocks read from TOAST tables (if any) in the table.
toast_blks_hit	bigint	Number of buffer hits in TOAST tables (if any) in the table.
tidx_blks_read	bigint	Number of disk blocks read from the TOAST tables index (if any) in the table.
tidx_blks_hit	bigint	Number of buffer-hits in the TOAST table index (if any) in the table.
last_updated	timestamp with time zone	Time when the monitoring data of the object in the view is updated for the last time.

8.2.10.2 DBE_PERF.SUMMARY_STATIO_USER_TABLES

Displays a summary of I/O status information about user relationship tables of each node in the cluster, as described in **Table 8-122**.

Table 8-122 DBE_PERF.SUMMARY_STATIO_USER_TABLES columns

Name	Type	Description
schemaname	name	Name of the schema that the table is in.
relname	name	Table name.
heap_blks_read	numeric	Number of disk blocks read from the table.
heap_blks_hit	numeric	Number of cache hits in the table.
idx_blks_read	numeric	Number of disk blocks read from indexes in the table.
idx_blks_hit	numeric	Number of cache hits in indexes in the table.
toast_blks_read	numeric	Number of disk blocks read from TOAST tables (if any) in the table.
toast_blks_hit	numeric	Number of buffer hits in TOAST tables (if any) in the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
tidx_blks_read	numeric	Number of disk blocks read from the TOAST tables index (if any) in the table.
tidx_blks_hit	numeric	Number of buffer hits in the TOAST table index (if any) in the table.

8.2.10.3 DBE_PERF.GLOBAL_STATIO_USER_TABLES

Displays I/O status information about all user relationship tables in namespaces on each node, as described in **Table 8-123**.

Table 8-123 DBE_PERF.GLOBAL_STATIO_USER_TABLES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	Table OID.
schemaname	name	Name of the schema that contains the table.
relname	name	Table name.
heap_blks_read	bigint	Number of disk blocks read from the table.
heap_blks_hit	bigint	Number of cache hits in the table.
idx_blks_read	bigint	Number of disk blocks read from indexes in the table.
idx_blks_hit	bigint	Number of cache hits in indexes in the table.
toast_blks_read	bigint	Number of disk blocks read from TOAST tables (if any) in the table.
toast_blks_hit	bigint	Number of buffer hits in TOAST tables (if any) in the table.
tidx_blks_read	bigint	Number of disk blocks read from the TOAST tables index (if any) in the table.
tidx_blks_hit	bigint	Number of buffer hits in the TOAST table index (if any) in the table.

8.2.10.4 DBE_PERF.STATIO_USER_INDEXES

Displays I/O status information about all user relationship table indexes in namespaces on the current node, as described in **Table 8-124**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-124 DBE_PERF.STATIO_USER_INDEXES columns

Name	Type	Description
relid	oid	OID of the table that the index is created for.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema that contains the index.
relname	name	Name of the table that the index is created for.
indexrelname	name	Index name.
idx_blks_read	bigint	Number of disk blocks read from the index.
idx_blks_hit	bigint	Number of cache hits in the index.
last_updated	timestamp with time zone	Time when the monitoring data of the object in the view is updated for the last time.

8.2.10.5 DBE_PERF.SUMMARY_STATIO_USER_INDEXES

Displays a summary of I/O status information about all user relationship table indexes in namespaces in the cluster, as described in **Table 8-125**.

Table 8-125 DBE_PERF.SUMMARY_STATIO_USER_INDEXES columns

Name	Type	Description
schemaname	name	Name of the schema that contains the index.
relname	name	Name of the table that the index is created for.
indexrelname	name	Index name.
idx_blks_read	numeric	Number of disk blocks read from the index.
idx_blks_hit	numeric	Number of cache hits in the index.

8.2.10.6 DBE_PERF.GLOBAL_STATIO_USER_INDEXES

Displays I/O status information about all user relationship table indexes in namespaces on each node, as described in **Table 8-126**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-126 DBE_PERF.GLOBAL_STATIO_USER_INDEXES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of the table that the index is created for.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema that contains the index.
relname	name	Name of the table that the index is created for.
indexrelname	name	Index name.
idx_blks_read	numeric	Number of disk blocks read from the index.
idx_blks_hit	numeric	Number of cache hits in the index.

8.2.10.7 DBE_PERF.STATIO_USER_SEQUENCES

Displays I/O status information about all user relationship table sequences in namespaces on the current node, as described in **Table 8-127**.

Table 8-127 DBE_PERF.STATIO_USER_SEQUENCE columns

Name	Type	Description
relid	oid	OID of the sequence.
schemaname	name	Name of the schema that the sequence is in.
relname	name	Sequence name.
blks_read	bigint	Number of disk blocks read from the sequence.
blks_hit	bigint	Number of cache hits in the sequence.

8.2.10.8 DBE_PERF.SUMMARY_STATIO_USER_SEQUENCES

Displays I/O status information about all user relationship table sequences in namespaces in the cluster, as described in **Table 8-128**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-128 DBE_PERF.SUMMARY_STATIO_USER_SEQUENCES columns

Name	Type	Description
schemaname	name	Name of the schema that the sequence is in.
relname	name	Sequence name.
blks_read	numeric	Number of disk blocks read from the sequence.
blks_hit	numeric	Number of cache hits in the sequence.

8.2.10.9 DBE_PERF.GLOBAL_STATIO_USER_SEQUENCES

Displays I/O status information about all user relationship table sequences in namespaces on each node, as described in **Table 8-129**.

Table 8-129 DBE_PERF.GLOBAL_STATIO_USER_SEQUENCES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of the sequence.
schemaname	name	Name of the schema that the sequence is in.
relname	name	Sequence name.
blks_read	bigint	Number of disk blocks read from the sequence.
blks_hit	bigint	Number of cache hits in the sequence.

8.2.10.10 DBE_PERF.STATIO_SYS_TABLES

Displays the I/O status information about all system catalogs in namespaces on the current node, as described in **Table 8-130**.

Table 8-130 DBE_PERF.STATIO_SYS_TABLES columns

Name	Type	Description
relid	oid	Table OID.
schemaname	name	Name of the schema that contains the table.
relname	name	Table name.
heap_blks_read	bigint	Number of disk blocks read from the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
heap_blks_hit	bigint	Number of cache hits in the table.
idx_blks_read	bigint	Number of disk blocks read from indexes in the table.
idx_blks_hit	bigint	Number of cache hits in indexes in the table.
toast_blks_read	bigint	Number of disk blocks read from TOAST tables (if any) in the table.
toast_blks_hit	bigint	Number of buffer hits in TOAST tables (if any) in the table.
tidx_blks_read	bigint	Number of disk blocks read from the TOAST tables index (if any) in the table.
tidx_blks_hit	bigint	Number of buffer-hits in the TOAST table index (if any) in the table.
last_updated	timestamp with time zone	Time when the monitoring data of the object in the view is updated for the last time.

8.2.10.11 DBE_PERF.SUMMARY_STATIO_SYS_TABLES

Displays a summary of I/O status information about system catalogs of each node in the cluster, as described in **Table 8-131**.

Table 8-131 DBE_PERF.SUMMARY_STATIO_SYS_TABLES columns

Name	Type	Description
schemaname	name	Name of the schema that contains the table.
relname	name	Table name.
heap_blks_read	numeric	Number of disk blocks read from the table.
heap_blks_hit	numeric	Number of cache hits in the table.
idx_blks_read	numeric	Number of disk blocks read from indexes in the table.
idx_blks_hit	numeric	Number of cache hits in indexes in the table.
toast_blks_read	numeric	Number of disk blocks read from TOAST tables (if any) in the table.
toast_blks_hit	numeric	Number of buffer hits in TOAST tables (if any) in the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
tidx_blks_read	numeric	Number of disk blocks read from the TOAST tables index (if any) in the table.
tidx_blks_hit	numeric	Number of buffer-hits in the TOAST table index (if any) in the table.

8.2.10.12 DBE_PERF.GLOBAL_STATIO_SYS_TABLES

Displays I/O status information about all system catalogs in namespaces on each node, as described in **Table 8-132**.

Table 8-132 DBE_PERF.GLOBAL_STATIO_SYS_TABLES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	Table OID.
schemaname	name	Name of the schema that contains the table.
relname	name	Table name.
heap_blks_read	bigint	Number of disk blocks read from the table.
heap_blks_hit	bigint	Number of cache hits in the table.
idx_blks_read	bigint	Number of disk blocks read from indexes in the table.
idx_blks_hit	bigint	Number of cache hits in indexes in the table.
toast_blks_read	bigint	Number of disk blocks read from TOAST tables (if any) in the table.
toast_blks_hit	bigint	Number of buffer hits in TOAST tables (if any) in the table.
tidx_blks_read	bigint	Number of disk blocks read from the TOAST tables index (if any) in the table.
tidx_blks_hit	bigint	Number of buffer-hits in the TOAST table index (if any) in the table.

8.2.10.13 DBE_PERF.STATIO_SYS_INDEXES

Displays the I/O status information about all system catalog indexes in the current namespace, as described in **Table 8-133**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-133 DBE_PERF.STATIO_SYS_INDEXES columns

Name	Type	Description
relid	oid	OID of the table that the index is created for.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema that contains the index.
relname	name	Name of the table that the index is created for.
indexrelname	name	Index name.
idx_blks_read	bigint	Number of disk blocks read from the index.
idx_blks_hit	bigint	Number of cache hits in the index.
last_updated	timestamp with time zone	Time when the monitoring data of the object in the view is updated for the last time.

8.2.10.14 DBE_PERF.SUMMARY_STATIO_SYS_INDEXES

Displays a summary of I/O status information about all system catalog indexes in namespaces in the cluster, as described in **Table 8-134**.

Table 8-134 DBE_PERF.SUMMARY_STATIO_SYS_INDEXES columns

Name	Type	Description
schemaname	name	Name of the schema that contains the index.
relname	name	Name of the table that the index is created for.
indexrelname	name	Index name.
idx_blks_read	numeric	Number of disk blocks read from the index.
idx_blks_hit	numeric	Number of cache hits in the index.

8.2.10.15 DBE_PERF.GLOBAL_STATIO_SYS_INDEXES

Displays I/O status information about all system catalog indexes in namespaces on each node, as described in **Table 8-135**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-135 DBE_PERF.GLOBAL_STATIO_SYS_INDEXES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of the table that the index is created for.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema that contains the index.
relname	name	Name of the table that the index is created for.
indexrelname	name	Index name.
idx_blks_read	numeric	Number of disk blocks read from the index.
idx_blks_hit	numeric	Number of cache hits in the index.

8.2.10.16 DBE_PERF.STATIO_SYS_SEQUENCES

Displays the I/O status information about all the system sequences in the current namespace, as described in **Table 8-136**.

Table 8-136 DBE_PERF.STATIO_SYS_SEQUENCES columns

Name	Type	Description
relid	oid	OID of the sequence.
schemaname	name	Name of the schema that the sequence is in.
relname	name	Sequence name.
blks_read	bigint	Number of disk blocks read from the sequence.
blks_hit	bigint	Number of cache hits in the sequence.

8.2.10.17 DBE_PERF.SUMMARY_STATIO_SYS_SEQUENCES

Displays I/O status information about all system sequences in namespaces in the cluster, as described in **Table 8-137**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-137 DBE_PERF.SUMMARY_STATIO_SYS_SEQUENCES columns

Name	Type	Description
schemaname	name	Name of the schema that the sequence is in.
relname	name	Sequence name.
blks_read	numeric	Number of disk blocks read from the sequence.
blks_hit	numeric	Number of cache hits in the sequence.

8.2.10.18 DBE_PERF.GLOBAL_STATIO_SYS_SEQUENCES

Displays I/O status information about all system sequences in namespaces on each node, as described in **Table 8-138**.

Table 8-138 DBE_PERF.GLOBAL_STATIO_SYS_SEQUENCES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of the sequence.
schemaname	name	Name of the schema that the sequence is in.
relname	name	Sequence name.
blks_read	bigint	Number of disk blocks read from the sequence.
blks_hit	bigint	Number of cache hits in the sequence.

8.2.10.19 DBE_PERF.STATIO_ALL_TABLES

Displays the I/O status information about each table (including TOAST tables) of the current node in the database, as described in **Table 8-139**.

Table 8-139 DBE_PERF.STATIO_ALL_TABLES columns

Name	Type	Description
relid	oid	Table OID.
schemaname	name	Name of the schema that contains the table.
relname	name	Table name.
heap_blks_read	bigint	Number of disk blocks read from the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
heap_blks_hit	bigint	Number of cache hits in the table.
idx_blks_read	bigint	Number of disk blocks read from indexes in the table.
idx_blks_hit	bigint	Number of cache hits in indexes in the table.
toast_blks_read	bigint	Number of disk blocks read from TOAST tables (if any) in the table.
toast_blks_hit	bigint	Number of buffer hits in TOAST tables (if any) in the table.
tidx_blks_read	bigint	Number of disk blocks read from the TOAST tables index (if any) in the table.
tidx_blks_hit	bigint	Number of buffer-hits in the TOAST table index (if any) in the table.
last_updated	timestamp with time zone	Time when the monitoring data of the object in the view is updated for the last time.

8.2.10.20 DBE_PERF.SUMMARY_STATIO_ALL_TABLES

Displays a summary of the I/O status information about each table (including TOAST tables) of each node in the cluster, as described in **Table 8-140**.

Table 8-140 DBE_PERF.SUMMARY_STATIO_ALL_TABLES columns

Name	Type	Description
schemaname	name	Name of the schema that contains the table.
relname	name	Table name.
heap_blks_read	numeric	Number of disk blocks read from the table.
heap_blks_hit	numeric	Number of cache hits in the table.
idx_blks_read	numeric	Number of disk blocks read from indexes in the table.
idx_blks_hit	numeric	Number of cache hits in indexes in the table.
toast_blks_read	numeric	Number of disk blocks read from TOAST tables (if any) in the table.
toast_blks_hit	numeric	Number of buffer hits in TOAST tables (if any) in the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
tidx_blks_read	numeric	Number of disk blocks read from the TOAST tables index (if any) in the table.
tidx_blks_hit	numeric	Number of buffer-hits in the TOAST table index (if any) in the table.
last_updated	timestamp with time zone	Time when the monitoring data of the table in the view is updated for the last time.

8.2.10.21 DBE_PERF.GLOBAL_STATIO_ALL_TABLES

Displays the I/O status information about each table (including TOAST tables) of each node in the cluster, as described in **Table 8-141**.

Table 8-141 DBE_PERF.GLOBAL_STATIO_ALL_TABLES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	Table OID.
schemaname	name	Name of the schema that contains the table.
relname	name	Table name.
heap_blks_read	bigint	Number of disk blocks read from the table.
heap_blks_hit	bigint	Number of cache hits in the table.
idx_blks_read	bigint	Number of disk blocks read from indexes in the table.
idx_blks_hit	bigint	Number of cache hits in indexes in the table.
toast_blks_read	bigint	Number of disk blocks read from TOAST tables (if any) in the table.
toast_blks_hit	bigint	Number of buffer hits in TOAST tables (if any) in the table.
tidx_blks_read	bigint	Number of disk blocks read from the TOAST tables index (if any) in the table.
tidx_blks_hit	bigint	Number of buffer-hits in the TOAST table index (if any) in the table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
last_updated	timestamp with time zone	Time when the monitoring data of the table in the view is updated for the last time.

8.2.10.22 DBE_PERF.STATIO_ALL_INDEXES

Contains one row for each index in the current database, showing I/O statistics about specific indexes, as described in **Table 8-142**.

Table 8-142 DBE_PERF.STATIO_ALL_INDEXES columns

Name	Type	Description
relid	oid	OID of the table that the index is created for.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema that contains the index.
relname	name	Name of the table that the index is created for.
indexrelname	name	Index name.
idx_blks_read	bigint	Number of disk blocks read from the index.
idx_blks_hit	bigint	Number of cache hits in the index.
last_updated	timestamp with time zone	Time when the monitoring data of the object in the view is updated for the last time.

8.2.10.23 DBE_PERF.SUMMARY_STATIO_ALL_INDEXES

Contains one row for each index in the cluster database, showing a summary of I/O statistics about specific indexes, as described in **Table 8-143**.

Table 8-143 DBE_PERF.SUMMARY_STATIO_ALL_INDEXES columns

Name	Type	Description
schemaname	name	Name of the schema that contains the index.
relname	name	Name of the table that the index is created for.
indexrelname	name	Index name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
idx_blks_read	numeric	Number of disk blocks read from the index.
idx_blks_hit	numeric	Number of cache hits in the index.
last_updated	timestamp with time zone	Time when the monitoring data of the index in the view is updated for the last time.

8.2.10.24 DBE_PERF.GLOBAL_STATIO_ALL_INDEXES

Contains one row for each index in databases on each node, showing I/O statistics about specific indexes, as described in **Table 8-144**.

Table 8-144 DBE_PERF.GLOBAL_STATIO_ALL_INDEXES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of the table that the index is created for.
indexrelid	oid	OID of the index.
schemaname	name	Name of the schema that contains the index.
relname	name	Name of the table that the index is created for.
indexrelname	name	Index name.
idx_blks_read	numeric	Number of disk blocks read from the index.
idx_blks_hit	numeric	Number of cache hits in the index.
last_updated	timestamp with time zone	Time when the monitoring data of the index in the view is updated for the last time.

8.2.10.25 DBE_PERF.STATIO_ALL_SEQUENCES

Contains one row for each sequence in the current database, showing I/O statistics about specific sequences, as described in **Table 8-145**.

Table 8-145 DBE_PERF.STATIO_ALL_SEQUENCES columns

Name	Type	Description
relid	oid	OID of the sequence.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
schemaname	name	Name of the schema that the sequence is in.
relname	name	Sequence name.
blks_read	bigint	Number of disk blocks read from the sequence.
blks_hit	bigint	Number of cache hits in the sequence.

8.2.10.26 DBE_PERF.SUMMARY_STATIO_ALL_SEQUENCES

Contains I/O statistics about one row of each sequence in databases in the cluster, as described in **Table 8-146**.

Table 8-146 DBE_PERF.SUMMARY_STATIO_ALL_SEQUENCES columns

Name	Type	Description
schemaname	name	Name of the schema that the sequence is in.
relname	name	Sequence name.
blks_read	numeric	Number of disk blocks read from the sequence.
blks_hit	numeric	Number of cache hits in the sequence.

8.2.10.27 DBE_PERF.GLOBAL_STATIO_ALL_SEQUENCES

Contains every row of each sequence in databases on each node, showing I/O statistics about specific sequences, as described in **Table 8-147**.

Table 8-147 DBE_PERF.GLOBAL_STATIO_ALL_SEQUENCES columns

Name	Type	Description
node_name	name	Node name.
relid	oid	OID of the sequence.
schemaname	name	Name of the schema that the sequence is in.
relname	name	Sequence name.
blks_read	bigint	Number of disk blocks read from the sequence.
blks_hit	bigint	Number of cache hits in the sequence.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.11 Communications Library

8.2.11.1 DBE_PERF.COMM_DELAY

Displays the TCP proxy communications library status for a single DN, as described in **Table 8-148**.

Table 8-148 DBE_PERF.COMM_DELAY columns

Name	Type	Description
node_name	text	Node name.
remote_name	text	Name of the peer node.
remote_host	text	IP address of the peer node.
stream_num	integer	Number of logical stream connections used by the current physical connection.
min_delay	integer	Minimum delay of the current physical connection within 1 minute (unit: μs). **NOTE** A negative result is invalid. Wait until the delay status is updated and query again.
average	integer	Average delay of the current physical connection within 1 minute (unit: μs).
max_delay	integer	Maximum delay of the current physical connection within 1 minute (unit: μs).

8.2.11.2 DBE_PERF.GLOBAL_COMM_DELAY

Displays the TCP proxy communications library status for all the DNs, as described in **Table 8-149**.

Table 8-149 DBE_PERF.GLOBAL_COMM_DELAY columns

Name	Type	Description
node_name	text	Node name.
remote_name	text	Name of the peer node.
remote_host	text	IP address of the peer node.
stream_num	integer	Number of logical stream connections used by the current physical connection.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
min_delay	integer	Minimum delay of the current physical connection within 1 minute (unit: μs). **NOTE** A negative result is invalid. Wait until the delay status is updated and query again.
average	integer	Average delay of the current physical connection within 1 minute (unit: μs).
max_delay	integer	Maximum delay of the current physical connection within 1 minute (unit: μs).

8.2.11.3 DBE_PERF.COMM_RECV_STREAM

Displays the receiving stream status of all TCP proxy communications libraries on a single DN, as described in **Table 8-150**.

Table 8-150 DBE_PERF.COMM_RECV_STREAM columns

Name	Type	Description
node_name	text	Node name.
local_tid	bigint	ID of the thread using this stream.
remote_name	text	Name of the peer node.
remote_tid	bigint	Peer thread ID.
idx	integer	Peer DN ID in the local DN.
sid	integer	Stream ID in the physical connection.
tcp_sock	integer	TCP socket used in the stream.
state	text	Stream status.
query_id	bigint	**debug_query_id** corresponding to the stream.
pn_id	integer	**plan_node_id** of the query executed by the stream.
send_smp	integer	**smpid** of the sender of the query executed by the stream.
recv_smp	integer	**smpid** of the receiver of the query executed by the stream.
recv_bytes	bigint	Total data volume received by the stream (unit: byte).
time	bigint	Current lifecycle service duration of the stream (unit: ms).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
speed	bigint	Average receiving rate of the stream (unit: byte/s).
quota	bigint	Current communication quota value of the stream (unit: byte).
buff_usize	bigint	Current size of the data cache of the stream (unit: byte).

8.2.11.4 DBE_PERF.GLOBAL_COMM_RECV_STREAM

Displays the receiving stream status of all TCP proxy communications libraries on all the DNs, as described in **Table 8-151**.

Table 8-151 DBE_PERF.GLOBAL_COMM_RECV_STREAM columns

Name	Type	Description
node_name	text	Node name.
local_tid	bigint	ID of the thread using this stream.
remote_name	text	Name of the peer node.
remote_tid	bigint	Peer thread ID.
idx	integer	Peer DN ID in the local DN.
sid	integer	Stream ID in the physical connection.
tcp_sock	integer	TCP socket used in the stream.
state	text	Stream status.
query_id	bigint	**debug_query_id** corresponding to the stream.
pn_id	integer	**plan_node_id** of the query executed by the stream.
send_smp	integer	**smpid** of the sender of the query executed by the stream.
recv_smp	integer	**smpid** of the receiver of the query executed by the stream.
recv_bytes	bigint	Total data volume received by the stream (unit: byte).
time	bigint	Current lifecycle service duration of the stream (unit: ms).
speed	bigint	Average receiving rate of the stream (unit: byte/s).
quota	bigint	Current communication quota value of the stream (unit: byte).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
buff_usize	bigint	Current size of the data cache of the stream (unit: byte).

8.2.11.5 DBE_PERF.COMM_SEND_STREAM

Displays the sending stream status of all TCP proxy communications libraries on a single DN, as described in **Table 8-152**.

Table 8-152 DBE_PERF.COMM_SEND_STREAM columns

Name	Type	Description
node_name	text	Node name.
local_tid	bigint	ID of the thread using this stream.
remote_name	text	Name of the peer node.
remote_tid	bigint	Peer thread ID.
idx	integer	Peer DN ID in the local DN.
sid	integer	Stream ID in the physical connection.
tcp_sock	integer	TCP socket used in the stream.
state	text	Stream status.
query_id	bigint	**debug_query_id** corresponding to the stream.
pn_id	integer	**plan_node_id** of the query executed by the stream.
send_smp	integer	**smpid** of the sender of the query executed by the stream.
recv_smp	integer	**smpid** of the receiver of the query executed by the stream.
send_bytes	bigint	Total data volume sent by the stream (unit: byte).
time	bigint	Current lifecycle service duration of the stream (unit: ms).
speed	bigint	Average sending rate of the stream (unit: byte/s).
quota	bigint	Current communication quota value of the stream (unit: byte).
wait_quota	bigint	Extra time generated when the stream waits for the quota value (unit: ms).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.11.6 DBE_PERF.GLOBAL_COMM_SEND_STREAM

Displays the sending stream status of all TCP proxy communications libraries on all the DNs, as described in **Table 8-153**.

Table 8-153 DBE_PERF.GLOBAL_COMM_SEND_STREAM columns

Name	Type	Description
node_name	text	Node name.
local_tid	bigint	ID of the thread using this stream.
remote_name	text	Name of the peer node.
remote_tid	bigint	Peer thread ID.
idx	integer	Peer DN ID in the local DN.
sid	integer	Stream ID in the physical connection.
tcp_sock	integer	TCP socket used in the stream.
state	text	Stream status.
query_id	bigint	**debug_query_id** corresponding to the stream.
pn_id	integer	**plan_node_id** of the query executed by the stream.
send_smp	integer	**smpid** of the sender of the query executed by the stream.
recv_smp	integer	**smpid** of the receiver of the query executed by the stream.
send_bytes	bigint	Total data volume sent by the stream (unit: byte).
time	bigint	Current lifecycle service duration of the stream (unit: ms).
speed	bigint	Average sending rate of the stream (unit: byte/s).
quota	bigint	Current communication quota value of the stream (unit: byte).
wait_quota	bigint	Extra time generated when the stream waits for the quota value (unit: ms).

8.2.11.7 DBE_PERF.COMM_STATUS

Displays the TCP proxy communications library status on a single DN, as described in **Table 8-154**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-154 DBE_PERF.COMM_STATUS columns

Name	Type	Description
node_name	text	Node name.
rxpck_rate	integer	Receiving rate of the communications library on the node, in byte/s.
txpck_rate	integer	Sending rate of the communications library on the node, in byte/s.
rxkbyte_rate	bigint	Receiving rate of the communications library on the bigint node, in KByte/s.
txkbyte_rate	bigint	Sending rate of the communications library on the bigint node, in KByte/s.
buffer	bigint	Size of the buffer of the Cmailbox.
memkbyte_libcomm	bigint	Communication memory size of the **libcomm** thread, in bytes.
memkbyte_libpq	bigint	Communication memory size of the **libpq** thread, in bytes.
used_pm	integer	Real-time usage of the **postmaster** thread.
used_sflow	integer	Real-time usage of the gs_sender_flow_controller thread.
used_rflow	integer	Real-time usage of the gs_receiver_flow_controller thread.
used_rloop	integer	Highest real-time usage among multiple gs_receivers_loop threads.
stream	integer	Total number of used logical connections.

8.2.11.8 DBE_PERF.GLOBAL_COMM_STATUS

Displays the TCP proxy communications library status on all the DNs, as described in **Table 8-155**.

Table 8-155 DBE_PERF.GLOBAL_COMM_STATUS columns

Name	Type	Description
node_name	text	Node name.
rxpck_rate	integer	Receiving rate of the communications library on the node, in byte/s.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
txpck_rate	integer	Sending rate of the communications library on the node, in byte/s.
rxkbyte_rate	bigint	Receiving rate of the communications library on the bigint node, in KByte/s.
txkbyte_rate	bigint	Sending rate of the communications library on the bigint node, in KByte/s.
buffer	bigint	Size of the buffer of the Cmailbox.
memkbyte_libcomm	bigint	Communication memory size of the **libcomm** thread, in bytes.
memkbyte_libpq	bigint	Communication memory size of the **libpq** thread, in bytes.
used_pm	integer	Real-time usage of the **postmaster** thread.
used_sflow	integer	Real-time usage of the gs_sender_flow_controller thread.
used_rflow	integer	Real-time usage of the gs_receiver_flow_controller thread.
used_rloop	integer	Highest real-time usage among multiple gs_receivers_loop threads.
stream	integer	Total number of used logical connections.

8.2.11.9 DBE_PERF.GLOBAL_COMM_CHECK_CONNECTION_STATUS

Displays the connection status of all CNs and all active nodes (CNs and primary DNs). The permission control is inherited from the **DBE_PERF** schema, as described in **Table 8-156**.

Table 8-156 DBE_PERF.GLOBAL_COMM_CHECK_CONNECTION_STATUS columns

Name	Type	Description
node_name	name	Instance name.
remote_name	name	Name of the peer instance.
remote_host	text	IP address of the peer instance.
remote_port	bigint	Port number of the peer instance.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
is_connected	boolean	Detects the network status between the current instance and the peer instance: • **t** (true): The network between instances is normal. • **f** (false): The network between instances is abnormal.
no_error_occur	boolean	Pooler connection result between the current instance and the peer instance. • **t** (true) indicates that the pooler connection is normal. • **f** (false) indicates that the pooler connection is abnormal.

8.2.12 Utility

8.2.12.1 DBE_PERF.REPLICATION_STAT

Describes information about log synchronization status, such as the locations where the sender sends logs and where the receiver receives logs, as described in **Table 8-157**.

Table 8-157 DBE_PERF.REPLICATION_STAT columns

Name	Type	Description
pid	bigint	PID of the thread.
usesysid	oid	User system ID.
usename	name	Username.
application_name	text	Program name.
client_addr	inet	Client address.
client_hostname	text	Client name.
client_port	integer	Port of the client.
backend_start	timestamp with time zone	Start time of the program.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
state	text	Log replication state: • Catch-up state. • Consistent streaming state.
sender_sent_location	text	Location where the sender sends logs.
receiver_write_location	text	Location where the receiver writes logs.
receiver_flush_location	text	Location where the receiver flushes logs.
receiver_replay_location	text	Location where the receiver replays logs.
sync_priority	integer	Priority of synchronous duplication. (**0** indicates asynchronization.)
sync_state	text	Synchronization state: • Asynchronous replication. • Synchronous replication. • Potential synchronization.
dbid	oid	ID of the database to which the statistics belong.

8.2.12.2 DBE_PERF.GLOBAL_REPLICATION_STAT

Displays information about log synchronization status on each node, such as the locations where the sender sends logs and where the receiver receives logs, as described in **Table 8-158**.

Table 8-158 DBE_PERF.GLOBAL_REPLICATION_STAT columns

Name	Type	Description
node_name	name	Node name.
pid	bigint	PID of the thread.
usesysid	oid	User system ID.
usename	name	Username.
application_name	text	Program name.
client_addr	inet	Client address.
client_hostname	text	Client name.
client_port	integer	Port of the client.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
backend_start	timestamp with time zone	Start time of the program.
state	text	Log replication state: • Catch-up state. • Consistent streaming state.
sender_sent_location	text	Location where the sender sends logs.
receiver_write_location	text	Location where the receiver writes logs.
receiver_flush_location	text	Location where the receiver flushes logs.
receiver_replay_location	text	Location where the receiver replays logs.
sync_priority	integer	Priority of synchronous duplication. (**0** indicates asynchronization.)
sync_state	text	Synchronization state: • Asynchronous replication. • Synchronous replication. • Potential synchronization.
dbid	oid	ID of the database to which the statistics belong.

8.2.12.3 DBE_PERF.REPLICATION_SLOTS

Displays replication slot information, as described in **Table 8-159**.

Table 8-159 DBE_PERF.REPLICATION_SLOTS columns

Name	Type	Description
slot_name	text	Replication slot name.
plugin	text	Name of the output plug-in corresponding to the logical replication slot.
slot_type	text	Replication slot type. • **physical**: physical replication slot. • **logical**: logical replication slot.
datoid	oid	OID of the database where the replication slot resides.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
database	name	Name of the database where the replication slot resides.
active	boolean	Determines whether the replication slot is activated. • **t** (true): yes. • **f** (false): no.
xmin	xid	XID of the earliest transaction that the database must reserve for the replication slot.
catalog_xmin	xid	XID of the earliest system-catalog-involved transaction that the database must reserve for the logical replication slot.
restart_lsn	text	Physical location of the earliest Xlog required by the replication slot.
dummy_standby	boolean	Reserved parameter. Specifies whether the peer end connected to the replication slot is a secondary node. • **t** (true): yes. • **f** (false): no.

8.2.12.4 DBE_PERF.GLOBAL_REPLICATION_SLOTS

Displays information about replicated slots on each node in the cluster, as described in **Table 8-160**.

Table 8-160 DBE_PERF.GLOBAL_REPLICATION_SLOTS columns

Name	Type	Description
node_name	name	Node name.
slot_name	text	Replication slot name.
plugin	text	Name of the output plug-in corresponding to the logical replication slot.
slot_type	text	Replication slot type. • **physical**: physical replication slot. • **logical**: logical replication slot.
datoid	oid	OID of the database where the replication slot resides.
database	name	Name of the database where the replication slot resides.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
active	boolean	Determines whether the replication slot is activated. • **t** (true): yes. • **f** (false): no.
x_min	xid	XID of the earliest transaction that the database must reserve for the replication slot.
catalog_xmin	xid	XID of the earliest system-catalog-involved transaction that the database must reserve for the logical replication slot.
restart_lsn	text	Physical location of the earliest Xlog required by the replication slot.
dummy_standby	boolean	Reserved parameter. Specifies whether the peer end connected to the replication slot is a secondary node. • **t** (true): yes. • **f** (false): no.

8.2.12.5 DBE_PERF.BGWRITER_STAT

Displays statistics about the BGWRITER thread's activities, as described in **Table 8-161**.

Table 8-161 DBE_PERF.BGWRITER_STAT columns

Name	Type	Description
checkpoints_timed	bigint	Number of scheduled checkpoints that have been performed.
checkpoints_req	bigint	Number of requested checkpoints that have been performed.
checkpoint_write_time	double precision	Total time spent on performing write operations during checkpointing, in milliseconds.
checkpoint_sync_time	double precision	Total time spent on performing sync operations during checkpointing, in milliseconds.
buffers_checkpoint	bigint	Number of buffers written during checkpointing.
buffers_clean	bigint	Number of buffers written by the BGWRITER thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
maxwritten_clean	bigint	Number of times that the BGWRITER thread stops cleanup scanning because the number of pages refreshed in a single scan is too large.
buffers_backend	bigint	Number of buffers written directly by the backend.
buffers_backend_fsync	bigint	Number of times that the backend thread automatically executes fsync. Generally, fsync is called by the BGWRITER thread.
buffers_alloc	bigint	Number of buffers allocated.
stats_reset	timestamp with time zone	Time when the statistics of the current row were reset last time.

8.2.12.6 DBE_PERF.GLOBAL_BGWRITER_STAT

Displays statistics about the bgwriter thread's activities on all instances, as described in **Table 8-162**.

Table 8-162 DBE_PERF.GLOBAL_BGWRITER_STAT columns

Name	Type	Description
node_name	name	Instance name.
checkpoints_timed	bigint	Number of scheduled checkpoints that have been performed.
checkpoints_req	bigint	Number of requested checkpoints that have been performed.
checkpoint_write_time	double precision	Total time spent on performing write operations during checkpointing, in milliseconds.
checkpoint_sync_time	double precision	Total time spent on performing sync operations during checkpointing, in milliseconds.
buffers_checkpoint	bigint	Number of buffers written during checkpointing.
buffers_clean	bigint	Number of buffers written by the BGWRITER thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
maxwritten_clean	bigint	Number of times that the BGWRITER thread stops cleanup scanning because the number of pages refreshed in a single scan is too large.
buffers_backend	bigint	Number of buffers written directly by the backend.
buffers_backend_fsync	bigint	Number of times that the backend thread automatically executes fsync. Generally, fsync is called by the BGWRITER thread.
buffers_alloc	bigint	Number of buffers allocated.
stats_reset	timestamp with time zone	Time when the statistics of the current row were reset last time.

8.2.12.7 DBE_PERF.POOLER_STATUS

Queries the cache connection status of the pooler module on the local CN, as described in **Table 8-163**.

Table 8-163 DBE_PERF.POOLER_STATUS columns

Name	Type	Description
database	text	Database name.
user_name	text	Username.
tid	bigint	In non-thread pool logic, this parameter indicates the ID of the thread connected to the CN. In thread pool logic, this parameter indicates the ID of the session connected to the CN.
node_oid	bigint	OID of the node connected.
node_name	name	Name of the node connected.
in_use	boolean	Specifies whether the connection is currently used. • **t** (true): The connection is in use. • **f** (false): The connection is not in use.
local_host	text	Local IP address.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
local_port	bigint	Local port number.
remote_host	text	IP address of the node connected.
node_port	bigint	Port number of the node connected.
fdsock	bigint	Port file descriptor.
remote_pid	bigint	If the peer site is not in the thread pool logic, the value is the thread ID of the peer site. If the peer site is in the thread pool logic, the value is the session ID of the peer site.
session_params	text	Session parameter.
used_count	bigint	Number of reuse times of a connection.
idx	bigint	Logical connection ID of the connected instance node.
streamid	bigint	Stream ID corresponding to each logical connection.

8.2.12.8 DBE_PERF.POOLER_AGENT

Queries information about the pooler agent on the connected CN to help locate connection pool problems. If a DN is connected for query, an empty table is returned. Only users with the SYSADMIN permission can access this view. For details about the return values, see **Table 8-164**.

Table 8-164 DBE_PERF.POOLER_AGENT columns

Name	Type	Description
index	bigint	Index of the current agent in the global agents.
database_name	text	Name of the database in the current session.
user_name	text	Username of the current session.
session_id	bigint	• ID of the thread connected to the CN for non-thread pool logic. • Session ID for connecting to the CN for thread pool logic.
dn_oids	oid[]	List of recorded DN OIDs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
cn_oids	oid[]	List of recorded CN OIDs.
agent_init	timestamp with time zone	Time when the agent is initialized. The agent is initialized when a session is created or the pgxc catalog information is reloaded.

8.2.12.9 DBE_PERF.GLOBAL_CKPT_STATUS

Displays the information about checkpoints and flushing pages of all instances in the entire cluster, as described in **Table 8-165**.

Table 8-165 DBE_PERF.GLOBAL_CKPT_STATUS columns

Name	Type	Description
node_name	text	Instance name.
ckpt_redo_point	text	Checkpoint of the current instance.
ckpt_clog_flush_num	bigint	Number of Clog flushing pages from the startup time to the current time.
ckpt_csnlog_flush_num	bigint	Number of CSN log flushing pages from the startup time to the current time.
ckpt_multixact_flush_num	bigint	Number of MultiXact flushing pages from the startup time to the current time.
ckpt_predicate_flush_num	bigint	Number of predicate flushing pages from the startup time to the current time.
ckpt_twophase_flush_num	bigint	Number of two-phase flushing pages from the startup time to the current time.

8.2.12.10 DBE_PERF.GLOBAL_DOUBLE_WRITE_STATUS

Displays the information about doublewrite files of all instances in the entire cluster, as described in **Table 8-166**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-166 DBE_PERF.GLOBAL_DOUBLE_WRITE_STATUS columns

Name	Type	Description
node_name	text	Instance name.
curr_dwn	bigint	Sequence number of the doublewrite file.
curr_start_page	bigint	Start page for restoring the doublewrite file.
file_trunc_num	bigint	Number of times that the doublewrite file is reused.
file_reset_num	bigint	Number of reset times after the doublewrite file is full.
total_writes	bigint	Total number of I/Os of the doublewrite file.
low_threshold_writes	bigint	Number of I/Os for inefficient doublewrite files.
high_threshold_writes	bigint	Number of I/Os for efficient doublewrite files.
total_pages	bigint	Total number of pages that are flushed to the doublewrite file area.
low_threshold_pages	bigint	Number of pages that are flushed with low efficiency.
high_threshold_pages	bigint	Number of pages that are flushed with high efficiency.
file_id	bigint	ID of the current doublewrite file.

8.2.12.11 DBE_PERF.GLOBAL_PAGEWRITER_STATUS

Displays the information about checkpoints and flushing pages of all instances in the entire cluster, as described in **Table 8-167**.

Table 8-167 DBE_PERF.GLOBAL_PAGEWRITER_STATUS columns

Name	Type	Description
node_name	text	Instance name.
pgwr_actual_flush_total_num	bigint	Total number of dirty pages flushed from the startup to the current time.
pgwr_last_flush_num	integer	Number of dirty pages flushed in the previous batch.
remain_dirty_page_num	bigint	Estimated number of remaining dirty pages.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
queue_head_page_rec_lsn	text	**recovery_lsn** of the first dirty page in the dirty page queue of the current instance.
queue_rec_lsn	text	**recovery_lsn** of the dirty page queue of the current instance.
current_xlog_insert_lsn	text	Position where Xlogs of the current instance are written. This parameter is invalid for a standby node.
ckpt_redo_point	text	Checkpoint of the current instance.

8.2.12.12 DBE_PERF.GLOBAL_POOLER_STATUS

Queries the cache connection status of the pooler modules on all CNs, as described in **Table 8-168**.

Table 8-168 DBE_PERF.GLOBAL_POOLER_STATUS columns

Name	Type	Description
source_node_name	name	Source node name.
database	text	Database name.
user_name	text	Username.
tid	bigint	In non-thread pool logic, this parameter indicates the ID of the thread connected to the CN. In thread pool logic, this parameter indicates the ID of the session connected to the CN.
node_oid	bigint	OID of the node connected.
node_name	name	Name of the node connected.
in_use	boolean	Specifies whether the connection is currently used. • **t** (true): The connection is in use. • **f** (false): The connection is not in use.
local_host	text	Local IP address.
local_port	bigint	Local port number.
remote_host	text	IP address of the connected node.
node_port	bigint	Port number of the node connected.
fdsock	bigint	Port file descriptor.
remote_pid	bigint	Thread ID of the remote node connected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
session_params	text	Session parameter.
used_count	bigint	Number of reuse times of a connection.
idx	bigint	Logical connection ID of the connected instance node.
streamid	bigint	Stream ID corresponding to each logical connection.

8.2.12.13 DBE_PERF.GLOBAL_RECORD_RESET_TIME

Obtains statistics about the reset (restart, active/standby switchover, and database deletion) time of each node in the cluster, as described in **Table 8-169**.

Table 8-169 DBE_PERF.GLOBAL_RECORD_RESET_TIME columns

Name	Type	Description
node_name	text	Node name.
reset_time	timestamp with time zone	Time to be reset.

8.2.12.14 DBE_PERF.GLOBAL_REDO_STATUS

Displays the replaying of logs about instances in the entire cluster, as described in **Table 8-170**.

Table 8-170 DBE_PERF.GLOBAL_REDO_STATUS columns

Name	Type	Description
node_name	text	Instance name.
redo_start_ptr	bigint	Start point for replaying the instance logs.
redo_start_time	bigint	Start time (UTC) when the instance logs are replayed.
redo_done_time	bigint	End time (UTC) when the instance logs are replayed.
curr_time	bigint	Current time (UTC) of the instance.
min_recovery_point	bigint	Minimum barrier for the current instance log to provide services after the replay is complete.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
read_ptr	bigint	Position for reading the instance logs.
last_replayed_read_ptr	bigint	Position for replaying the instance logs.
recovery_done_ptr	bigint	Replay position after the instance is started.
read_xlog_io_counter	bigint	Number of I/Os when the current instance reads and replays logs.
read_xlog_io_total_dur	bigint	Total I/O duration when the current instance reads and replays logs.
read_data_io_counter	bigint	Number of data page I/O reads during replay in the current instance log.
read_data_io_total_dur	bigint	Total I/O duration reads during replay in the current instance log.
write_data_io_counter	bigint	Number of data page I/O writes during replay in the current instance log.
write_data_io_total_dur	bigint	Total I/O duration writes during replay in the current instance log.
process_pending_counter	bigint	Number of synchronization times of log distribution threads during replay in the instance log.
process_pending_total_dur	bigint	Synchronization duration of log distribution threads during replay in the instance log.
apply_counter	bigint	Number of synchronization times of replay threads during replay in the instance log.
apply_total_dur	bigint	Synchronization duration of replay threads during replay in the instance log.
speed	bigint	Log replay rate of the current instance. The value is updated every time when a 256 MB log is replayed. The unit is byte/s.
local_max_ptr	bigint	Maximum number of replay logs received by the localhost after the instance is started.
primary_flush_ptr	bigint	Log point where the host flushes logs to a disk.
worker_info	text	Replay thread information of the instance. If parallel replay is not enabled, the value is null.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.12.15 DBE_PERF.GLOBAL_RECOVERY_STATUS

Displays log flow control information about the primary and standby nodes, as described in **Table 8-171**.

Table 8-171 DBE_PERF.GLOBAL_RECOVERY_STATUS columns

Name	Type	Description
node_name	text	Node name (including the primary and standby nodes).
standby_node_name	text	Name of the standby node.
source_ip	text	IP address of the primary node.
source_port	integer	Port number of the primary node.
dest_ip	text	IP address of the standby node.
dest_port	integer	Port number of the standby node.
current_rto	bigint	Current log flow control time of the standby node (unit: s).
target_rto	bigint	Expected flow control time of the standby node specified by the corresponding GUC parameter (unit: s).
current_sleep_time	bigint	Sleep time (in microseconds) required by the host to achieve the expected RTO.

8.2.12.16 DBE_PERF.CLASS_VITAL_INFO

Checks whether the OIDs of the same table or index are consistent for WDR snapshots, as described in **Table 8-172**.

Table 8-172 DBE_PERF.CLASS_VITAL_INFO columns

Name	Type	Description
relid	oid	Table OID.
schemaname	name	Schema name.
relname	name	Table name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
relkind	"char"	Object type. Its value can be: • **r**: ordinary table. • **t**: TOAST table. • **i**: index. • **G**: global secondary index.
last_updated	timestamp with time zone	Time when the monitoring data of the object in the view is updated for the last time.

8.2.12.17 DBE_PERF.USER_LOGIN

Records the number of user logins and logouts, as described in **Table 8-173**.

Table 8-173 DBE_PERF.USER_LOGIN columns

Name	Type	Description
node_name	text	Node name.
user_name	text	Username.
user_id	integer	User OID (Its value is the same as that of **oid** in pg_authid.)
login_counter	bigint	Number of logins.
logout_counter	bigint	Number of logouts.
dbid	oid	ID of the database to which the user logs in.

8.2.12.18 DBE_PERF.SUMMARY_USER_LOGIN

Records information about user logins and logouts on all CNs, as described in **Table 8-174**.

Table 8-174 DBE_PERF.SUMMARY_USER_LOGIN columns

Name	Type	Description
node_name	text	Node name.
user_name	text	Username.
user_id	integer	User OID (Its value is the same as that of **oid** in pg_authid.)
login_counter	bigint	Number of logins.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
logout_counter	bigint	Number of logouts.
dbid	oid	ID of the database to which the user logs in.

8.2.12.19 DBE_PERF.GLOBAL_GET_BGWRITER_STATUS

Displays the information about pages flushed by the bgwriter threads of all instances in the entire cluster, number of pages in the candidate buffer chain, and buffer eviction information, as described in **Table 8-175**.

Table 8-175 DBE_PERF.GLOBAL_GET_BGWRITER_STATUS columns

Name	Type	Description
node_name	text	Instance name.
bgwr_actual_flush_total_num	bigint	Total number of dirty pages flushed by the bgwriter thread from the startup time to the current time.
bgwr_last_flush_num	integer	Number of dirty pages flushed by the bgwriter thread in the previous batch.
candidate_slots	integer	Number of pages in the current candidate buffer chain.
get_buffer_from_list	bigint	Number of times that pages are obtained from the candidate buffer chain during buffer eviction.
get_buf_clock_sweep	bigint	Number of times that pages are obtained from the original eviction solution during buffer eviction.

8.2.12.20 DBE_PERF.GLOBAL_SINGLE_FLUSH_DW_STATUS

Displays information about doublewrite files eliminated on a single page of all instances in the entire cluster, as described in **Table 8-176**. In the displayed information, the information before the slash (/) indicates the page flushing status of the first version, and the information after the slash (/) indicates the page flushing status of the second version. The doublewrite page flushing status of the first version refers to performing single-page eviction doublewrite on a non-segment page, and the doublewrite page flushing case of the second version refers to performing single-page eviction doublewrite on a segment page.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-176 DBE_PERF.GLOBAL_SINGLE_FLUSH_DW_STATUS columns

Name	Type	Description
node_name	text	Instance name.
curr_dwn	text	Sequence number of the doublewrite file.
curr_start_page	text	Start position of the current doublewrite file.
total_writes	text	Total number of data write pages in the current doublewrite file.
file_trunc_num	text	Number of times that the doublewrite file is reused.
file_reset_num	text	Number of reset times after the doublewrite file is full.

8.2.12.21 DBE_PERF.GLOBAL_CANDIDATE_STATUS

Displays the number of candidate buffers and buffer eviction information of all instances in the database, as described in **Table 8-177**.

Table 8-177 DBE_PERF.GLOBAL_CANDIDATE_STATUS columns

Name	Type	Description
node_name	text	Node name.
candidate_slots	integer	Number of pages in the candidate buffer chain of the current normal buffer pool.
get_buf_from_list	bigint	Number of times that pages are obtained from the candidate buffer chain during buffer eviction in the current normal buffer pool.
get_buf_clock_sweep	bigint	Number of times that pages are obtained from the original eviction solution during buffer eviction in the current normal buffer pool.
seg_candidate_slots	integer	Number of pages in the candidate buffer chain of the current segment buffer pool.
seg_get_buf_from_list	bigint	Number of times that pages are obtained from the candidate buffer chain during buffer eviction in the current segment buffer pool.
seg_get_buf_clock_sweep	bigint	Number of times that pages are obtained from the original eviction solution during buffer eviction in the current segment buffer pool.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.12.22 DBE_PERF.PARALLEL_DECODE_STATUS

Displays parallel decoding information about replication slots on the current node, as described in **Table 8-178**. This view needs to be queried on DNs. If it is queried on CNs, no information is returned.

Table 8-178 DBE_PERF.PARALLEL_DECODE_STATUS columns

Name	Type	Description
slot_name	text	Specifies the replication slot name.
parallel_decode_num	integer	Specifies the number of parallel decoder threads of the replication slot.
read_change_queue_length	text	Concatenates the current length of the log queue read by each decoder thread and then outputs the result.
decode_change_queue_length	text	Concatenates the current length of the decoding result queue of each decoder thread and then outputs the result.
reader_lsn	text	Specifies the location of the log read by the current reader thread.
working_txn_cnt	bigint	Specifies the number of transactions that are being concatenated in the current sender thread.
working_txn_memory	bigint	Specifies the total memory occupied by concatenation transactions in the sender thread, in bytes.
decoded_time	timestamp with time zone	Time when the replication slot reads the logical log for the last time.

8.2.12.23 DBE_PERF.GLOBAL_PARALLEL_DECODE_STATUS

Displays parallel decoding information about replication slots on each primary DN in a cluster, as described in **Table 8-179**. The view needs to be queried on a CN. An error is reported when the view is queried on a DN.

Table 8-179 DBE_PERF.GLOBAL_PARALLEL_DECODE_STATUS columns

Name	Type	Description
node_name	name	Node name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
slot_name	text	Specifies the replication slot name.
parallel_decode_num	integer	Specifies the number of parallel decoder threads of the replication slot.
read_change_queue_length	text	Concatenates the current length of the log queue read by each decoder thread and then outputs the result.
decode_change_queue_length	text	Concatenates the current length of the decoding result queue of each decoder thread and then outputs the result.
reader_lsn	text	Specifies the location of the log read by the current reader thread.
working_txn_cnt	bigint	Specifies the number of transactions that are being concatenated in the current sender thread.
working_txn_memory	bigint	Total memory occupied by concatenation transactions in the sender thread, in bytes.
decoded_time	timestamp with time zone	Time when the replication slot reads the logical log for the last time.

8.2.12.24 DBE_PERF.PARALLEL_DECODE_THREAD_INFO

Displays information about threads that perform parallel decoding on the current node, as described in **Table 8-180**. This view needs to be queried on DNs. If it is queried on CNs, no information is returned.

Table 8-180 DBE_PERF.PARALLEL_DECODE_THREAD_INFO columns

Name	Type	Description
thread_id	bigint	Thread ID.
slot_name	text	Replication slot name.
thread_type	text	Thread type (sender, reader, or decoder).
seq_number	integer	Sequence number (starting from 1) of threads of the same type in the current replication slot.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.12.25 DBE_PERF.GLOBAL_PARALLEL_DECODE_THREAD_INFO

Displays information about threads that perform parallel decoding on each primary DN in a cluster, as described in **Table 8-181**. The view needs to be queried on a CN. An error is reported when the view is queried on a DN.

Table 8-181 DBE_PERF.GLOBAL_PARALLEL_DECODE_THREAD_INFO columns

Name	Type	Description
node_name	name	Node name.
thread_id	bigint	Thread ID.
slot_name	text	Replication slot name.
thread_type	text	Thread type (sender, reader, or decoder).
seq_number	integer	Sequence number (starting from 1) of threads of the same type in the current replication slot.

8.2.12.26 DBE_PERF.GLOBAL_ADIO_COMPLETER_STATUS

Displays statistics about the AIO Completer thread in all cluster instances, as described in **Table 8-182**. The view needs to be queried on a CN. An error is reported when the view is queried on a DN.

Table 8-182 DBE_PERF.GLOBAL_ADIO_COMPLETER_STATUS columns

Parameter	Type	Description
node_name	text	Instance name.
tid	bigint	ID of the AIO Completer thread.
thread_type	text	AIO Completer thread type (read or write).
aio_submitted_num	bigint	Number of committed asynchronous I/O requests of the AIO Completer thread.
aio_completed_num	bigint	Number of completed asynchronous I/O requests of the AIO Completer thread.
aio_incompleted_num	bigint	Number of asynchronous I/O requests that are not completed of the AIO Completer thread.
slot_count_left	bigint	Indicates the number of idle slots.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.12.27 DBE_PERF.GLOBAL_AIO_SLOT_USAGE_STATUS

Displays statistics about asynchronous I/O commit slots in all cluster instances, as described in **Table 8-183**. The view needs to be queried on a CN. An error is reported when the view is queried on a DN.

Table 8-183 DBE_PERF.GLOBAL_AIO_SLOT_USAGE_STATUS columns

Parameter	Type	Description
node_name	text	Instance name.
slot_id	integer	Slot ID.
slot_type	"char"	Slot type. The value can be **r** (read) or **w** (write).
status	boolean	Slot usage status.
buffer_id	bigint	Buffer ID corresponding to the slot.
relfilenode_blocknum	text	Position of the physical page where the buffer corresponding to the slot is located.
lsn	bigint	LSN corresponding to the page.
submitted_time	bigint	Time when a page is committed asynchronously.
elapsed_time	bigint	Waiting time of the page.

8.2.12.28 DBE_PERF.GLOBAL_XLOG_REDO_STAT

This view displays statistics about various types of logs that have been replayed on each standby DN in the cluster, as shown in **Table 8-184**.

Table 8-184 DBE_PERF.GLOBAL_XLOG_REDO_STAT columns

Parameter	Type	Description
node_name	text	Instance name.
xlog_type	text	Log types.
rmid	xid	Resource manager ID.
num	xid	Number of replayed statements.
extra	xid	The valid values for page replay logs and xact logs are different: • Number of pages read from the disk if the log is of the page type. • Number of deleted files if the log is of the xact type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.12.29 DBE_PERF.EMERGENCY_OPERATION_RECORDS

This view is used to view the usage of the UDS O&M channel on each CN or DN, as shown in **Table 8-185**.

Table 8-185 DBE_PERF.EMERGENCY_OPERATION_RECORDS columns

Parameter	Type	Description
record_time	timestamptz	Time when the O&M channel is used.
action	text	Operations performed through the O&M channel.
source	text	Source of the O&M channel. The value can be **gsql** or **cm**.
detail	text	Operation result.

8.2.12.30 DBE_PERF.KERNEL_INFO

Displays key data of the corresponding module on each CN or DN, as described in **Table 8-186**.

Table 8-186 DBE_PERF.KERNEL_INFO columns

Parameter	Type	Description
node_name	text	Node name.
module	text	Module name. • **XACT** (transaction module) • **STANDBY** (standby module) • **UNDO** (undo module) • **HOTPATH** (hot patch module) • **LOCK** (lock module) • FULL_SQL • SQL

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
name	text	Name of the key data in the memory state. • **startup_max_xid** (**XACT**: maximum XID when a thread is started) • **recent_local_xmin** (**XACT**: minimum XID of local active transactions) • **recent_global_xmin** (**XACT**: minimum XID of global active transactions) • **standby_xmin** (**STANDBY**: minimum XID of an active transaction on the standby node) • **standby_redo_cleanup_xmin** (**STANDBY**: minimum XID of cleanup logs during redo on the standby node) • **standby_redo_cleanup_xmin_lsn** (**STANDBY**: LSN of the minimum XID of cleanup logs during redo on the standby node) • **local_csn_min** (**XACT**: minimum CSN of a local active transaction) • **replication_slot_xmin** (**XACT**: minimum XID of a replication slot) • **replication_slot_catalog_xmin** (**XACT**: minimum XID of a catalog replication slot) • **global_recycle_xid** (**UNDO**: minimum XID of a global undo recycling transaction) • **global_frozen_xid** (**XACT**: minimum XID of a globally frozen transaction) • **recent_xmin XACT**: (minimum XID of active transactions in the current session) • **next_csn** (**XACT**: CSN of the next transaction) • **hotpatch_additional_info** (**HOTPATH**: reserved column for hot patches) • **stmt_session_discard_records** (**FULL SQL**: data volume discarded by full SQL statements due to full slots supported by the kernel) • **stmt_shm_flush_discard_records** (**FULL SQL**: data volume discarded by full SQL statements due to full ringbuf supported by the kernel) • **idle_in_trans_timeout_records** (**XACT**: accumulated data volume of idle transaction timeout counts supported by the kernel)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
		• **total_lock_usage** (**LOCK**: number of locks applied for through the primary lock table) • **total_lock_usage_rate** (**LOCK**: usage of the prim+ary lock tablespace) • **fastpath_strong_relation_locks_count** (**LOCK**: partition where FASTPATH strong lock conflicts exist and the number of strong locks corresponding to the partition) • **fastpath_failure_rate** (**LOCK**: percentage of locks that cannot be applied for through FASTPATH due to strong lock conflicts in the corresponding partition) • **lock_acquire_lack_fastpath_count** (**LOCK**: total number of times that locks fail to be applied for through FASTPATH because FASTPATH slots are used up) • **n_soft_parse** (**SQL**: number of soft parsing times) • **n_hard_parse** (**SQL**: number of hard parsing times)
value	text	Value of the key data in the memory state.

8.2.13 Lock

8.2.13.1 DBE_PERF.LOCKS

Displays information about locks held by each open transaction, as described in **Table 8-187**.

Table 8-187 DBE_PERF.LOCKS columns

Name	Type	Description
locktype	text	Type of the locked object. The value can be **relation**, **extend**, **page**, **tuple**, **transactionid**, **virtualxid**, **object**, **userlock**, **advisory**, or **tablespace**.
database	oid	OID of the database in which the locked object exists. • The OID is **0** if the object is a shared object. • The OID is **NULL** if the object is a transaction ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
relation	oid	OID of the relationship targeted by the lock. The value is **NULL** if the object is not a relationship or part of a relationship.
page	integer	Page number targeted by the lock within the relationship. The value is **NULL** if the object is not a relationship page or row page.
tuple	smallint	Row number targeted by the lock within the page. The value is **NULL** if the object is not a row.
bucket	integer	Hash bucket number.
virtualxid	text	Virtual ID of the transaction targeted by the lock. The value is **NULL** if the object is not a virtual transaction ID.
transactionid	xid	ID of the transaction targeted by the lock. The value is **NULL** if the object is not a transaction ID.
classid	oid	OID of the system catalog that contains the object. The value is **NULL** if the object is not a general database object.
objid	oid	OID of the locked object within its system catalog. The value is **NULL** if the object is not a general database object.
objsubid	smallint	Column number for a column in the table. The value is **0** if the object is some other object type. The value is **NULL** if the object is not a general database object.
virtualtransaction	text	Virtual ID of the transaction holding or awaiting this lock.
pid	bigint	Logical ID of the server thread holding or awaiting this lock. The value is **NULL** if the lock is held by a prepared transaction.
sessionid	bigint	ID of the session holding or awaiting this lock The value is **NULL** if the lock is held by a prepared transaction.
mode	text	Lock mode held or desired by this thread.
granted	boolean	• The value is **TRUE** if the lock is a held lock. • The value is **FALSE** if the lock is an awaited lock.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
fastpath	boolean	The value is **TRUE** if the lock is obtained through **fast-path**, and is **FALSE** if the lock is obtained through the main lock table.
locktag	text	Information about the lock that the session waits for. It can be parsed using the **locktag_decode()** function.
global_sessionid	text	Global session ID.

8.2.13.2 DBE_PERF.GLOBAL_LOCKS

Displays information about locks held by open transactions on each node, as described in **Table 8-188**.

Table 8-188 DBE_PERF.GLOBAL_LOCKS columns

Name	Type	Description
node_name	name	Node name.
locktype	text	Type of the locked object. The value can be **relation**, **extend**, **page**, **tuple**, **transactionid**, **virtualxid**, **object**, **userlock**, **advisory**, or **tablespace**.
database	oid	OID of the database in which the locked object exists. • The OID is **0** if the object is a shared object. • The OID is **NULL** if the object is a transaction ID.
relation	oid	OID of the relationship targeted by the lock. The value is **NULL** if the object is not a relationship or part of a relationship.
page	integer	Page number targeted by the lock within the relationship. The value is **NULL** if the object is not a relationship page or row page.
tuple	smallint	Row number targeted by the lock within the page. The value is **NULL** if the object is not a row.
bucket	integer	Hash bucket ID.
virtualxid	text	Virtual ID of the transaction targeted by the lock. The value is **NULL** if the object is not a virtual transaction ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
transactionid	xid	ID of the transaction targeted by the lock. The value is **NULL** if the object is not a transaction ID.
classid	oid	OID of the system catalog that contains the object. The value is **NULL** if the object is not a general database object.
objid	oid	OID of the locked object within its system catalog. The value is **NULL** if the object is not a general database object.
objsubid	smallint	Column number for a column in the table. The value is **0** if the object is some other object type. The value is **NULL** if the object is not a general database object.
virtualtransaction	text	Virtual ID of the transaction holding or awaiting this lock.
pid	bigint	Logical ID of the server thread holding or awaiting this lock. The value is **NULL** if the lock is held by a prepared transaction.
sessionid	bigint	ID of the session holding or awaiting this lock. The value is **NULL** if the lock is held by a prepared transaction.
global_sessionid	text	Global session ID.
mode	text	Lock mode held or desired by this thread.
granted	boolean	• The value is **TRUE** if the lock is a held lock. • The value is **FALSE** if the lock is an awaited lock.
fastpath	boolean	The value is **TRUE** if the lock is obtained through **fast-path**, and is **FALSE** if the lock is obtained through the main lock table.
locktag	text	Information about the lock that the session waits for. It can be parsed using the locktag_decode() function.

8.2.14 Wait Event

8.2.14.1 DBE_PERF.WAIT_EVENTS

Displays statistics about wait events on the current node, as described in **Table 8-189**. For details about key events in the kernel, see **Table 8-192**. Alternatively,

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

view the list of all events in the system in the wait_event_info view. For details about the impact of each transaction lock on services, see **LOCK**.

The constraints on the query of this system catalog are as follows:

This system catalog is controlled by **track_activities** and **enable_instr_track_wait**. If the parameters are disabled, the system does not record wait events, and no data can be queried in this catalog.

Table 8-189 DBE_PERF.WAIT_EVENTS columns

Name	Type	Description
nodename	text	Database node name.
type	text	Event type.
event	text	Event name.
wait	bigint	Number of waiting times.
failed_wait	bigint	Number of times that the lock wait fails.
total_wait_time	bigint	Total waiting time (unit: μs).
avg_wait_time	bigint	Average waiting time (unit: μs).
max_wait_time	bigint	Maximum waiting time (unit: μs).
min_wait_time	bigint	Minimum waiting time (unit: μs).
last_updated	timestamp with time zone	Last time when the event was updated.

8.2.14.2 DBE_PERF.GLOBAL_WAIT_EVENTS

Displays statistics about wait events on each node, as described in **Table 8-190**.

Table 8-190 DBE_PERF.GLOBAL_WAIT_EVENTS columns

Name	Type	Description
nodename	text	Database node name.
type	text	Event type.
event	text	Event name.
wait	bigint	Number of waiting times.
failed_wait	bigint	Number of times that the lock wait fails.
total_wait_time	bigint	Total waiting time (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
avg_wait_time	bigint	Average waiting time (unit: μs).
max_wait_time	bigint	Maximum waiting time (unit: μs).
min_wait_time	bigint	Minimum waiting time (unit: μs).
last_updated	timestamp with time zone	Last time when the event was updated.

8.2.14.3 DBE_PERF.WAIT_EVENT_INFO

Displays the details about wait events, as described in **Table 8-191** and **Table 8-192**.

Table 8-191 DBE_PERF.WAIT_EVENT_INFO columns

Name	Type	Description
module	text	Name of the module an event belongs to.
type	text	Event type.
event	text	Event name.

Table 8-192 Wait event information list

Module Category	Event Category	Event	Description
Lock	Wait event	acquire lock	Waiting for locking until the locking succeeds or times out.
SharedMemory	LWLock event	ShmemIndexLock	Protects the primary index table, a hash table, in shared memory.
Shared buffer	LWLock event	BufMappingLock	Protects operations on a shared-buffer mapping table.
Lmgr	LWLock event	LockMgrLock	Protects the information about a common lock structure.
LWLock	Wait event	acquire lwlock	Waits for a lightweight lock.
I/O	Wait event	wait io	Waiting until the I/O is complete.
COMM	Wait event	wait cmd	Waits for reading network communication packets to complete.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
COMM	Wait event	wait pooler get conn	Waits for pooler to obtain connections.
COMM	Wait event	wait pooler abort conn	Waits for pooler to terminate connections.
COMM	Wait event	wait pooler clean conn	Waits for pooler to clear connections.
COMM	Wait event	get conn	Obtaining the connection to other nodes.
COMM	Wait event	set cmd	Waits for running the **SET**, **RESET**, or **TRANSACTION BLOCK LEVEL** statement on the connection.
COMM	Wait event	cancel query	Canceling the SQL statement that is being executed through the connection.
COMM	Wait event	stop query	Stopping the query that is being executed through the connection.
COMM	Wait event	wait node	Waits for receiving data through the connection to an instance.
COMM	Wait event	flush data	Waits for sending data to other nodes in the network.
COMM	Wait event	stream get conn	Waits for establishing connections to consumer nodes when the stream flow is initialized.
COMM	Wait event	wait producer ready	Waits for every producer to get ready when the stream flow is initialized.
Stream	Wait event	synchronize quit	Waits for the threads in the stream thread group to quit when the stream plan ends. If information similar to "synchronize quit, total *m*, remain *n*" is displayed in the **wait_status** column, there are **m** concurrent tasks, and **n** of them are not in the unified exit state.
Stream	Wait event	wait stream group destroy	Waits for destroying the stream node group when the stream plan ends.
Stream	Wait event	wait stream thread ready	Waits for the stream producer to be ready. It is blocked until its thread ID is successfully set.
Stream	Wait event	wait create stream thread	Waits for the stream thread to start.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
Transaction	Wait event	wait transaction sync	Waits for transaction synchronization.
Transaction	Wait event	wait data sync	Waits for the completion of data page synchronization to the standby instance.
Transaction	Wait event	wait data sync queue	Waits for putting the row-store data pages into the synchronization queue.
Transaction	LWLock event	OidGenLock	Prevents different threads from generating the same OID.
Transaction	LWLock event	XidGenLock	Prevents two transactions from obtaining the same transaction ID.
Transaction	LWLock event	ProcArrayLock	Prevents concurrent access to or concurrent modification on ProcArray shared arrays.
Transaction	LWLock event	SubtransControlLock	Used to prevent concurrent access to or concurrent modification on the subtransaction control data structure.
Transaction	LWLock event	MultiXactGenLock	Used to allocate a unique MultiXact ID in serial mode.
Transaction	LWLock event	TwoPhaseStateLock	Used to prevent concurrent access to or modification on two-phase information sharing arrays.
Transaction	LWLock event	SerializableXactHashLock	Used to prevent concurrent read/write or concurrent write/write on a sharing structure for serializable transactions.
Transaction	LWLock event	SerializableFinishedListLock	Used to prevent concurrent read/write or concurrent write/write on a shared linked list for completed serial transactions.
Transaction	LWLock event	SerializablePredicateLockListLock	Used to protect a linked list of serializable transactions that have locks.
Transaction	LWLock event	PredicateLockMgrLock	Used to protect the information about a lock structure that has serializable transactions.
Transaction	LWLock event	OldSerXid SLRU lwlock	Used to protect SLRU buffers of old transaction IDs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
Transaction	LWLock event	OldSerXidLock	Used to protect a structure that records serializable transactions that have conflicts.
Transaction	Lock event	transactionid	Adds a lock to a transaction ID.
Transaction	Lock event	virtualxid	Adds a lock to a virtual transaction ID.
Checkpoint	LWLock event	CheckpointLock	Used to prevent multi-checkpoint concurrent execution.
Checkpoint	LWLock event	CheckpointerCommLock	Used to send file flush requests to a checkpointer. The request structure needs to be inserted to a request queue in serial mode.
Analyze	LWLock event	AutoanalyzeLock	Used to obtain and release resources related to a task that allows for autoanalyze execution.
Vacuum	LWLock event	BtreeVacuumLock	Used to prevent **VACUUM** from clearing pages that are being used by B-tree indexes.
Vacuum	LWLock event	AutovacuumLock	Used to access the autovacuum worker array in serial mode.
Vacuum	LWLock event	AutovacuumScheduleLock	Used to distribute tables requiring **VACUUM** in serial mode.
Auto vacuum	LWLock event	AutovacuumLock	Protects the autovacuum shared memory structure.
Auto vacuum	LWLock event	AutovacuumScheduleLock	Used to protect the information about autovacuum workers.
Auto analyze	LWLock event	AutoanalyzeLock	Used to protect the *autoAnalyzeFreeProcess* variable and ensure that no more than 10 autoanalyze threads are running at the same time.
WAL	Wait event	wait wal sync	Waits for the completion of WAL of synchronization from the specified LSN to the standby instance.
WAL	I/O event	WALBootstrapSync	Flushes an initialized WAL file to a disk during database initialization.
WAL	I/O event	WALBootstrapWrite	Writes an initialized WAL file during database initialization.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
WAL	I/O event	WALCopyRead	Reads operation generated when an existing WAL file is read for replication after archiving and restoration.
WAL	I/O event	WALCopySync	Flushes a replicated WAL file to a disk after archiving and restoration.
WAL	I/O event	WALCopyWrite	Writes operation generated when an existing WAL file is read for replication after archiving and restoration.
WAL	I/O event	WALInitSync	Flushes a newly initialized WAL file to a disk during log reclaiming or writing.
WAL	I/O event	WALInitWrite	Initializes a newly created WAL file to **0** during log reclaiming or writing.
WAL	I/O event	WALRead	Reads data from Xlogs during redo operations on two-phase files.
WAL	I/O event	WALSyncMethodAssign	Flushes all open WAL files to a disk.
WAL	I/O event	WALWrite	Writes a WAL file.
WAL	I/O event	LOGCTRL_SLEEP	Collects statistics on the number of stream control times and the sleep time of log stream control.
WAL	LWLock event	RcvWriteLock	Used to prevent concurrent call of **WalDataRcvWrite**.
WAL	LWLock event	WALBufMappingLock	An exclusive (X) lock needs to be added when the next page of an Xlog buffer is initialized.
WAL	LWLock event	WALInsertLock	Prevents multiple programs from writing data to the same Xlog buffer at the same time.
WAL	LWLock event	WALWriteLock	Used to prevent concurrent WAL write.
Relation	LWLock event	SInvalReadLock	Used to prevent concurrent execution with invalid message deletion.
Relation	LWLock event	SinvalWriteLock	Used to prevent concurrent execution with invalid message write and deletion.
Relation	LWLock event	RelCacheInitLock	Used to add a lock before any operations are performed on the **init** file when messages are invalid.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
Relation	LWLock event	TablespaceCreateLock	Used to check whether a tablespace already exists.
Relation	LWLock event	RelfilenodeReuseLock	Used to prevent the link to a reused column attribute file from being canceled by mistake.
Relation	Lock event	relation	Adds a lock to a table.
Relation	Lock event	extend	Adds a lock to a table being scaled out.
Relation	Lock event	partition	Adds a lock to a partitioned table.
Relation	Lock event	partition_seq	Adds a lock to a partition of a partitioned table.
WLM	Wait event	wait active statement	Waits for active statements.
WLM	Wait event	wait memory	Waits for free memory.
DDL/DCL	Wait event	create index	Waits for the completion of index creation.
DDL/DCL	Wait event	analyze	Waits for analysis completion.
DDL/DCL	Wait event	vacuum	Waits for the completion of the **VACUUM** operation.
DDL/DCL	LWLock event	DelayDDLLock	Used to prevent concurrent DDL operations.
DDL/DCL	Wait event	vacuum full	Waits for the completion of the **VACUUM FULL** operation.
Executor	Wait event	Sort	Waits for the completion of tuple sorting.
Executor	Wait event	Sort - write file	Writes sorted data to a file temporarily since the memory is limited during merge sort.
Executor	Wait event	Material	Waits for tuple materialization.
Executor	Wait event	Material - write file	Waits for writing a materialized tuple to a file.
Executor	Wait event	HashJoin - build hash	Waits until a hash table is created when a hash join is executed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
Executor	Wait event	HashJoin - write file	Waits for writing the hash result of a tuple to a disk when a hash join is executed.
Executor	Wait event	HashAgg - build hash	Waits until a hash table is created when a hash aggregate is executed.
Executor	Wait event	HashAgg - write file	Waits for writing the hash result of a tuple to a disk when a hash aggregate is executed.
Executor	Wait event	HashSetop - build hash	Waits until a hash table is created when an OP operation is performed using the hash algorithm.
Executor	Wait event	HashSetop - write file	Waits for writing the hash result of a tuple to a disk when an OP operation is performed using the hash algorithm.
Executor	Wait event	wait sync consumer next step	Waits for the stream consumer to perform the next step.
Executor	Wait event	wait sync producer next step	Waits for the stream producer to perform the next step.
GTM	Wait event	gtm connect	Waits for connecting to GTM.
GTM	Wait event	gtm reset xmin	Waits for GTM to reset the minimum transaction ID.
GTM	Wait event	gtm get xmin	Waits for obtaining the minimum transaction ID from GTM.
GTM	Wait event	gtm get gxid	Waits for obtaining the global transaction ID from GTM during transaction startup.
GTM	Wait event	gtm get csn	Waits for obtaining the CSN from GTM during transaction startup.
GTM	Wait event	gtm get snapshot	Waits for obtaining snapshots from GTM during transaction startup.
GTM	Wait event	gtm begin trans	Waits for GTM to start a transaction.
GTM	Wait event	gtm commit trans	Waits for GTM to commit a transaction.
GTM	Wait event	gtm rollback trans	Waits for GTM to roll back transactions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
GTM	Wait event	gtm start preprare trans	Waits for GTM to complete the first phase during two-phase commit.
GTM	Wait event	gtm prepare trans	Waits for GTM to complete the second phase during two-phase commit.
GTM	Wait event	gtm open sequence	Waits for GTM to create a sequence.
GTM	Wait event	gtm close sequence	Waits for GTM to complete the **ALTER SEQUENCE** operation.
GTM	Wait event	gtm set sequence val	Waits for GTM to set a sequence value.
GTM	Wait event	gtm drop sequence	Waits for GTM to delete a sequence.
GTM	Wait event	gtm rename sequence	Waits for GTM to rename a sequence.
GTM	LWLock event	GTMHostInfo Lock	Used to protect GTM information.
Temp File	I/O event	BufFileRead	Reads data from a temporary file to a specified buffer.
Temp File	I/O event	BufFileWrite	Writes the content of a specified buffer to a temporary file.
Pg_control	I/O event	ControlFileRead	Reads the **pg_control** file, mainly during database startup, checkpoint execution, and primary/standby verification.
Pg_control	I/O event	ControlFileSync	Flushes the **pg_control** file to a disk, mainly during database initialization.
Pg_control	I/O event	ControlFileSyncUpdate	Flushes the **pg_control** file to a disk, mainly during database startup, checkpoint execution, and primary/standby verification.
Pg_control	I/O event	ControlFileWrite	Writes the **pg_control** file, mainly during database initialization.
Pg_control	I/O event	ControlFileWriteUpdate	Updates the **pg_control** file, mainly during database startup, checkpoint execution, and primary/standby verification.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
Pg_control	LWLock event	ControlFileLock	Used to prevent concurrent read/write or concurrent write/write on the **pg_control** file.
File operation	I/O event	CopyFileRead	Reads a file during file copying.
File operation	I/O event	CopyFileWrite	Writes a file during file copying.
File operation	I/O event	DataFileExtend	Writes a file during file name extension.
Table data file	I/O event	DataFileImmediateSync	Flushes a table data file to a disk immediately.
Table data file	I/O event	DataFilePrefetch	Reads a table data file asynchronously.
Table data file	I/O event	DataFileRead	Reads a table data file synchronously.
Table data file	I/O event	DataFileSync	Synchronizes a table data file to a disk.
Table data file	I/O event	DataFileTruncate	Truncates a table data file.
Table data file	I/O event	DataFileWrite	Writes a table data file.
Table data file	LWLock event	SyncScanLock	Used to determine the start position of a relfilenode during heap scanning
Table data file	LWLock event	RelationMappingLock	Used to wait for the mapping file between system catalogs and storage locations to be updated.
metadata	LWLock event	MetaCacheSweepLock	Used to add a lock when metadata is cyclically washed out.
postmaster.pid	I/O event	LockFileAddToDataDirRead	Reads the **postmaster.pid** file.
postmaster.pid	I/O event	LockFileAddToDataDirSync	Flushes the **postmaster.pid** file to a disk.
postmaster.pid	I/O event	LockFileAddToDataDirWrite	Writes PID information into the **postmaster.pid** file.
Pid File	I/O event	LockFileCreateRead	Reads the LockFile file **%s.lock**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
Pid File	I/O event	LockFileCreateSync	Flushes the LockFile file **%s.lock** to a disk.
Pid File	I/O event	LockFileCreateWRITE	Writes PID information into the LockFile file **%s.lock**.
System catalog mapping file	I/O event	RelationMapRead	Reads the mapping file between system catalogs and storage locations.
System catalog mapping file	I/O event	RelationMapSync	Flushes the mapping file between system catalogs and storage locations to a disk.
System catalog mapping file	I/O event	RelationMapWrite	Writes the mapping file between system catalogs and storage locations.
Streaming replication	I/O event	ReplicationSlotRead	Reads a streaming replication slot file This operation is performed during a restart.
Streaming replication	I/O event	ReplicationSlotRestoreSync	Flushes a streaming replication slot file to a disk.
Streaming replication	I/O event	ReplicationSlotSync	Flushes a temporary streaming replication slot file to a disk during checkpoint execution.
Streaming replication	I/O event	ReplicationSlotWrite	Writes a temporary streaming replication slot file during checkpoint execution.
Streaming replication	LWLock event	ReplicationSlotAllocationLock	Used to allocate a replication slot.
Streaming replication	LWLock event	ReplicationSlotControlLock	Used to detect replication slot name conflicts and identify replication slots that can be allocated.
Clog	I/O event	SLRUFlushSync	Flushes the **pg_clog** file to a disk, mainly during checkpoint execution and database shutdown.
Clog	I/O event	SLRURead	Reads the **pg_clog** file.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
Clog	I/O event	SLRUSync	Writes dirty pages into the **pg_clog** file, and flushes the file to a disk, mainly during checkpoint execution and database shutdown.
Clog	I/O event	SLRUWrite	Writes the **pg_clog** file.
Clog	LWLock event	CLogControlLock	Used to prevent concurrent access to or concurrent modification on the Clog control data structure.
Clog	LWLock event	MultiXactOffsetControlLock	Used to prevent concurrent read/write or concurrent write/write on **pg_multixact/offset**.
Clog	LWLock event	MultiXactMemberControlLock	Used to prevent concurrent read/write or concurrent write/write on **pg_multixact/members**.
timelinehistory	I/O event	TimelineHistoryRead	Reads the **timelinehistory** file, during database startup.
timelinehistory	I/O event	TimelineHistorySync	Flushes the **timelinehistory** file to a disk, during database startup.
timelinehistory	I/O event	TimelineHistoryWrite	Writes the **timelinehistory** file.
pg_twophase	I/O event	TwophaseFileRead	Reads the **pg_twophase** file, mainly during two-phase transaction commit and restoration.
pg_twophase	I/O event	TwophaseFileSync	Flushes the **pg_twophase** file to a disk, mainly during two-phase transaction commit and restoration.
pg_twophase	I/O event	TwophaseFileWrite	Writes the **pg_twophase** file, mainly during two-phase transaction commit and restoration.
Cluster	LWLock event	NodeTableLock	Used to protect a shared structure that stores CNs and DNs information.
Concurrency	LWLock event	PoolerLock	Used to prevent two threads from simultaneously obtaining the same connection from a connection pool.
Concurrency	LWLock event	AsyncCtlLock	Used to prevent concurrent access to or concurrent modification on the sharing notification status.
Concurrency	LWLock event	AsyncQueueLock	Used to prevent concurrent access to or concurrent modification on the sharing notification queue.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
Double write	I/O event	DoubleWriteFileWrite	Writes pages to a doublewrite file during the doublewrite process.
Double write	I/O event	DoubleWriteFileRead	Reads a doublewrite file during restoration for a halfwrite.
Statistics file	LWLock event	FileStatLock	Used to protect a data structure that stores statistics file information.
Master-slave replication	LWLock event	SyncRepLock	Protects Xlog synchronization information during primary/standby replication.
Master-slave replication	LWLock event	ReplicationSlotAllocationLock	Used to add a lock when a primary node allocates streaming replication slots during primary/standby replication.
Master-slave replication	LWLock event	ReplicationSlotControlLock	Used to prevent concurrent update of streaming replication slot status during primary/standby replication.
Master-slave replication	LWLock event	LsnXlogChkFileLock	Used to serially update the Xlog flush points for primary and standby servers recorded in a specific structure.
Master-slave replication	LWLock event	DataSyncRepLock	Used to protect data page synchronization information during primary/standby replication.
Speed up the cluster	LWLock event	SearchServerLibLock	Used to add a lock on the file read operation when a specific dynamic library is initially loaded in GPU-accelerated scenarios.
Resource manage	LWLock event	ResourcePoolHashLock	Used to prevent concurrent access to or concurrent modification on a resource pool table, a hash table.
OBS	LWLock event	OBSGetPathLock	Used to prevent concurrent read/write or concurrent write/write on an OBS path.
OBS	LWLock event	OBSRuntimeLock	Used to obtain environment variables, for example, *GAUSSHOME*.
LLVM	LWLock event	LLVMDumpIRLock	Used to export the assembly language for dynamically generating functions.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
LLVM	LWLock event	LLVMParseIRLock	Used to compile and parse a finished IR function from the IR file at the start position of a query.
MPP is compatible with ORACLE scheduled task function	LWLock event	JobShmemLock	Used to protect global variables in the shared memory that is periodically read during a scheduled job where MPP is compatible with Oracle Database.
Operator history information statistics	LWLock event	OperatorRealTLock	Used to prevent concurrent access to or concurrent modification on a global structure that contains real-time data at the operator level.
Operator history information statistics	LWLock event	OperatorHistLock	Used to prevent concurrent access to or concurrent modification on a global structure that contains historical data at the operator level.
query history information statistics	LWLock event	SessionRealTLock	Used to prevent concurrent access to or concurrent modification on a global structure that contains real-time data at the query level.
query history information statistics	LWLock event	SessionHistLock	Used to prevent concurrent access to or concurrent modification on a global structure that contains historical data at the query level.
query history information statistics	LWLock event	WaitCountHashLock	Used to protect a shared structure in user statement counting scenarios.
barrier	LWLock event	BarrierLock	Used to ensure that only one thread is creating a barrier at a time.
CSN	LWLock event	CSNBufMappingLock	Used to protect CSN pages.
instrumentation	LWLock event	UniqueSQLMappingLock	Used to protect a unique SQL hash table.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
instrumentation	LWLock event	InstrUserLock	Used to protect a user hash table.
instrumentation	LWLock event	PercentileLock	Used to protect global percentile buffers.
instrumentation	LWLock event	InstrWorkloadLock	Used to protect a workload transaction hash table.
Pgproc	LWLock event	Pgproc lwlock	Used to protect the PGPROC structure.
Async buffer	LWLock event	AsyncCtlLock	Used to protect asynchronization buffers.
MultiXact	LWLock event	MultiXactOffset lwlock	Used to protect SLRU buffers of a MultiXact offset.
MultiXact	LWLock event	MultiXactMemberlwlock	Used to protect SLRU buffer of a MultiXact member.
CBM	LWLock event	CBMParseXlogLock	Protects the lock used when CBM parses Xlogs.
BadBlock	LWLock event	BadBlockStatHashLock	Used to protect the hash table **global_bad_block_stat**.
Page	Lock event	page	Adds a lock to a table page.
Tuple	Lock event	tuple	Adds a lock to a tuple on a page.
object	Lock event	object	Adds a lock to an object.
user	Lock event	userlock	Adds a lock to a user.
advisor	Lock event	advisory	Adds an advisory lock.
ODBC	LWLock event	ExtensionConnectorLibLock	Adds a lock when a specific dynamic library is loaded or uninstalled in ODBC connection initialization scenarios.
COMM	Wait event	Accept client conn - Total Time	(After GaussDB receives a connection request from a client) Total duration from the time when a connection is established to the time when the connection is successful.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Module Category	Event Category	Event	Description
COMM	Wait event	Accept client conn - ThrdPool - add epoll	(After GaussDB receives a connection request from a client) Duration from the time when a connection is established to the time when the thread pool adds the session handle to the epoll in thread pool mode.
COMM	Wait event	Accept client conn - ThrdPool - wait worker	(After GaussDB receives a connection request from a client) Duration from the time when the thread pool adds the session handle to the epoll to the time when the worker thread starts to work in thread pool mode.
COMM	Wait event	Accept client conn - ThrdPool - init session	(After GaussDB receives a connection request from a client) Duration from the time when the worker thread starts to work to the time when the session initialization ends in thread pool mode.
COMM	Wait event	Accept client conn - Worker - init proc	(After GaussDB receives a connection request from a client) Duration from the time when a connection is established to the time when the thread initialization ends in non-thread pool mode.
COMM	Wait event	Accept client conn - Worker - init session	(After GaussDB receives a connection request from a client) Duration from the time when the thread initialization ends to the time when the session initialization ends in non-thread pool mode.

8.2.14.4 DBE_PERF.GLOBAL_DATANODES_WAIT_EVENTS

Displays statistics about wait events on the DN, as described in **Table 8-193**. When the **enable_direct_standby_datanodes** parameter is enabled, statistics about wait events on the standby DN are displayed.

Table 8-193 DBE_PERF.GLOBAL_DATANODES_WAIT_EVENTS columns

Name	Type	Description
nodename	text	Database node name.
type	text	Event type.
event	text	Event name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
wait	bigint	Number of times that the lock wait fails.
failed_wait	bigint	Number of waiting failures.
total_wait_time	bigint	Total waiting time (unit: μs).
avg_wait_time	bigint	Average waiting time (unit: μs).
max_wait_time	bigint	Maximum waiting time (unit: μs).
min_wait_time	bigint	Minimum waiting time (unit: μs).
last_updated	timestamp with time zone	Last time when the event was updated.

8.2.15 Configuration

8.2.15.1 DBE_PERF.CONFIG_SETTINGS

Displays information about GUC parameters of the database, as described in Table 8-194.

Table 8-194 DBE_PERF.CONFIG_SETTINGS columns

Name	Type	Description
name	text	Parameter name.
setting	text	Current parameter value.
unit	text	Implicit unit of the parameter.
category	text	Logical group of the parameter.
short_desc	text	Brief description of the parameter.
extra_desc	text	Detailed description of the parameter.
context	text	Context required to set the parameter value, including **internal**, **postmaster**, **sighup**, **backend**, **superuser**, and **user**.
vartype	text	Parameter type, including **bool**, **enum**, **integer**, **real**, or **string**.
source	text	Method of assigning the parameter value.
min_val	text	Minimum value of the parameter. If the parameter type is not numeric, the value of this column is **null**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
max_val	text	Maximum value of the parameter. If the parameter type is not numeric, the value of this column is **null**.
enumvals	text[]	Valid values of an enum parameter. If the parameter type is not enum, the value of this column is **null**.
boot_val	text	Default parameter value used upon the database startup.
reset_val	text	Default parameter value used upon the database reset.
sourcefile	text	Configuration file used to set parameter values. If parameter values are not configured using the configuration file, the value of this column is **null**.
sourceline	integer	Row number in the configuration file for setting parameter values. If parameter values are not configured using the configuration file, the value of this column is **null**.

8.2.15.2 DBE_PERF.GLOBAL_CONFIG_SETTINGS

Displays information about GUC parameters of databases on each node, as described in **Table 8-195**.

Table 8-195 DBE_PERF.GLOBAL_CONFIG_SETTINGS columns

Name	Type	Description
node_name	text	Node name.
name	text	Parameter name.
setting	text	Current parameter value.
unit	text	Implicit unit of the parameter.
category	text	Logical group of the parameter.
short_desc	text	Brief description of the parameter.
extra_desc	text	Detailed description of the parameter.
context	text	Context required to set the parameter value, including **internal**, **postmaster**, **sighup**, **backend**, **superuser**, and **user**.
vartype	text	Parameter type, including **bool**, **enum**, **integer**, **real**, or **string**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
source	text	Method of assigning the parameter value.
min_val	text	Maximum value of the parameter. If the parameter type is not numeric, the value of this column is **null**.
max_val	text	Minimum value of the parameter. If the parameter type is not numeric, the value of this column is **null**.
enumvals	text[]	Valid values of an enum parameter. If the parameter type is not enum, the value of this column is **null**.
boot_val	text	Default parameter value used upon the database startup.
reset_val	text	Default parameter value used upon the database reset.
sourcefile	text	Configuration file used to set parameter values. If parameter values are not configured using the configuration file, the value of this column is **null**.
sourceline	integer	Row number in the configuration file for setting parameter values. If parameter values are not configured using the configuration file, the value of this column is **null**.

8.2.16 Operator

8.2.16.1 DBE_PERF.OPERATOR_HISTORY_TABLE

Displays records about operators of completed jobs. Data is dumped from the kernel to this system catalog, as described in **Table 8-196**.

Table 8-196 DBE_PERF.OPERATOR_HISTORY_TABLE columns

Name	Type	Description
queryid	bigint	Internal query ID used for statement execution.
pid	bigint	Backend thread ID.
plan_node_id	integer	Plan node ID of the execution plan of a query.
plan_node_name	text	Name of the operator corresponding to the plan node ID.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
start_time	timestamp with time zone	Time when the operator starts to process the first data record.
duration	bigint	Total execution time of the operator (unit: ms).
query_dop	integer	DOP of the operator.
estimated_rows	bigint	Number of rows estimated by the optimizer.
tuple_processed	bigint	Number of elements returned by the operator.
min_peak_memory	integer	Minimum peak memory used by the operator on all DNs (unit: MB).
max_peak_memory	integer	Maximum peak memory used by the operator on all DNs (unit: MB).
average_peak_memory	integer	Average peak memory used by the operator on all DNs (unit: MB).
memory_skew_percent	integer	Memory usage skew of the operator among each DN.
min_spill_size	integer	Minimum spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
max_spill_size	integer	Maximum spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
average_spill_size	integer	Average spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
spill_skew_percent	integer	DN spill skew when a spill occurs.
min_cpu_time	bigint	Minimum execution time of the operator on all DNs (unit: ms).
max_cpu_time	bigint	Maximum execution time of the operator on all DNs (unit: ms).
total_cpu_time	bigint	Total execution time of the operator on all DNs (unit: ms).
cpu_skew_percent	integer	Skew of the execution time among DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
warning	text	Warning. The following warnings are displayed: • Sort/SetOp/HashAgg/HashJoin spill • Spill file size large than 256MB • Broadcast size large than 100MB • Early spill • Spill times is greater than 3 • Spill on memory adaptive • Hash table conflict

8.2.16.2 DBE_PERF.OPERATOR_RUNTIME

Displays information about operators of the jobs that are being executed by the current user, as described in **Table 8-197**.

Table 8-197 DBE_PERF.OPERATOR_RUNTIME columns

Name	Type	Description
queryid	bigint	Internal query ID used for statement execution.
pid	bigint	Backend thread ID.
plan_node_id	integer	Plan node ID of the execution plan of a query.
plan_node_name	text	Name of the operator corresponding to the plan node ID.
start_time	timestamp with time zone	Time when the operator starts to process the first data record.
duration	bigint	Total execution time of the operator (unit: ms).
status	text	Execution status of the current operator. Its value can be **finished** or **running**.
query_dop	integer	DOP of the operator.
estimated_rows	bigint	Number of rows estimated by the optimizer.
tuple_processed	bigint	Number of elements returned by the operator.
min_peak_memory	integer	Minimum peak memory used by the operator on all DNs (unit: MB).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
max_peak_memory	integer	Maximum peak memory used by the operator on all DNs (unit: MB).
average_peak_memory	integer	Average peak memory used by the operator on all DNs (unit: MB).
memory_skew_percent	integer	Memory usage skew of the operator among each DN.
min_spill_size	integer	Minimum spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
max_spill_size	integer	Maximum spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
average_spill_size	integer	Average spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
spill_skew_percent	integer	DN spill skew when a spill occurs.
min_cpu_time	bigint	Minimum execution time of the operator on all DNs (unit: ms).
max_cpu_time	bigint	Maximum execution time of the operator on all DNs (unit: ms).
total_cpu_time	bigint	Total execution time of the operator on all DNs (unit: ms).
cpu_skew_percent	integer	Skew of the execution time among DNs.
warning	text	Warning. The following warnings are displayed: • Sort/SetOp/HashAgg/HashJoin spill • Spill file size large than 256MB • Broadcast size large than 100MB • Early spill • Spill times is greater than 3 • Spill on memory adaptive • Hash table conflict

8.2.16.3 DBE_PERF.GLOBAL_OPERATOR_HISTORY

Displays the records about operators after jobs are executed by the current user on all CNs, as described in **Table 8-198**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-198 DBE_PERF.GLOBAL_OPERATOR_HISTORY columns

Name	Type	Description
queryid	bigint	Internal query ID used for statement execution.
pid	bigint	Backend thread ID.
plan_node_id	integer	Plan node ID of the execution plan of a query.
plan_node_name	text	Name of the operator corresponding to the plan node ID.
start_time	timestamp with time zone	Time when the operator starts to process the first data record.
duration	bigint	Total execution time of the operator (unit: ms).
query_dop	integer	DOP of the operator.
estimated_rows	bigint	Number of rows estimated by the optimizer.
tuple_processed	bigint	Number of elements returned by the operator.
min_peak_memory	integer	Minimum peak memory used by the operator on all DNs (unit: MB).
max_peak_memory	integer	Maximum peak memory used by the operator on all DNs (unit: MB).
average_peak_memory	integer	Average peak memory used by the operator on all DNs (unit: MB).
memory_skew_percent	integer	Memory usage skew of the operator among each DN.
min_spill_size	integer	Minimum spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
max_spill_size	integer	Maximum spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
average_spill_size	integer	Average spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
spill_skew_percent	integer	DN spill skew when a spill occurs.
min_cpu_time	bigint	Minimum execution time of the operator on all DNs (unit: ms).
max_cpu_time	bigint	Maximum execution time of the operator on all DNs (unit: ms).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
total_cpu_time	bigint	Total execution time of the operator on all DNs (unit: ms).
cpu_skew_percent	integer	Skew of the execution time among DNs.
warning	text	Warning. The following warnings are displayed: • Sort/SetOp/HashAgg/HashJoin spill • Spill file size large than 256MB • Broadcast size large than 100MB • Early spill • Spill times is greater than 3 • Spill on memory adaptive • Hash table conflict

8.2.16.4 DBE_PERF.GLOBAL_OPERATOR_HISTORY_TABLE

Displays the records about operators of completed jobs on all CNs. Data is dumped from the kernel to the system catalog GS_WLM_OPERATOR_INFO. This view is an aggregated view for querying the system catalog GS_WLM_OPERATOR_INFO from all CNs.

Columns in this view are the same as those in **Table 8-198**.

8.2.16.5 DBE_PERF.GLOBAL_OPERATOR_RUNTIME

Displays information about operators of the jobs that are being executed by the current user on all CNs, as described in **Table 8-199**.

Table 8-199 DBE_PERF.GLOBAL_OPERATOR_RUNTIME columns

Name	Type	Description
queryid	bigint	Internal query ID used for statement execution.
pid	bigint	Backend thread ID.
plan_node_id	integer	Plan node ID of the execution plan of a query.
plan_node_name	text	Name of the operator corresponding to the plan node ID.
start_time	timestamp with time zone	Time when the operator starts to process the first data record.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
duration	bigint	Total execution time of the operator (unit: ms).
status	text	Execution status of the current operator. Its value can be **finished** or **running**.
query_dop	integer	DOP of the operator.
estimated_rows	bigint	Number of rows estimated by the optimizer.
tuple_processed	bigint	Number of elements returned by the operator.
min_peak_memory	integer	Minimum peak memory used by the operator on all DNs (unit: MB).
max_peak_memory	integer	Maximum peak memory used by the operator on all DNs (unit: MB).
average_peak_memory	integer	Average peak memory used by the operator on all DNs (unit: MB).
memory_skew_percent	integer	Memory usage skew of the operator among DNs.
min_spill_size	integer	Minimum spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
max_spill_size	integer	Maximum spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
average_spill_size	integer	Average spilled data among all DNs when a spill occurs (unit: MB; default value: **0**).
spill_skew_percent	integer	DN spill skew when a spill occurs.
min_cpu_time	bigint	Minimum execution time of the operator on all DNs (unit: ms).
max_cpu_time	bigint	Maximum execution time of the operator on all DNs (unit: ms).
total_cpu_time	bigint	Total execution time of the operator on all DNs (unit: ms).
cpu_skew_percent	integer	Skew of the execution time among DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
warning	text	Warning. The following warnings are displayed: • Sort/SetOp/HashAgg/HashJoin spill • Spill file size large than 256MB • Broadcast size large than 100MB • Early spill • Spill times is greater than 3 • Spill on memory adaptive • Hash table conflict

8.2.17 Workload Manager

8.2.17.1 DBE_PERF.WLM_CGROUP_CONFIG

Displays information about a Cgroup for a job that is being executed, as described in **Table 8-200**.

Table 8-200 DBE_PERF.WLM_CGROUP_CONFIG columns

Name	Type	Description
cgroup_name	text	Cgroup name.
priority	integer	Job priority.
usage_percent	integer	Percentage of resources used by the Cgroup.
shares	bigint	CPU quota allocated to a Cgroup.
cpuacct	bigint	Allocated CPU quota.
cpuset	text	Allocated CPU cores.
relpath	text	Relative path of the Cgroup.
valid	text	Specifies whether the Cgroup is valid.
node_group	text	Node group name.

8.2.17.2 DBE_PERF.WLM_CLUSTER_RESOURCE_RUNTIME

Displays a DN resource summary, as described in **Table 8-201**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-201 DBE_PERF.WLM_CLUSTER_RESOURCE_RUNTIME columns

Name	Type	Description
min_mem_util	integer	Minimum memory usage of a DN.
max_mem_util	integer	Maximum memory usage of a DN.
min_cpu_util	integer	Minimum CPU usage of a DN.
max_cpu_util	integer	Maximum CPU usage of a DN.
min_io_util	integer	Minimum I/O usage of a DN.
max_io_util	integer	Maximum I/O usage of a DN.
used_mem_rate	integer	Maximum memory usage of a physical node.

8.2.17.3 DBE_PERF.WLM_CONTROLGROUP_CONFIG

Displays information about all Cgroups in the current database, as described in **Table 8-202**.

Table 8-202 DBE_PERF.WLM_CONTROLGROUP_CONFIG columns

Name	Type	Description
name	text	Cgroup name.
type	text	Cgroup type.
gid	bigint	Cgroup ID.
classgid	bigint	ID of the **Class** Cgroup to which a **Workload** Cgroup belongs.
class	text	Class Cgroup.
workload	text	Workload Cgroup.
shares	bigint	CPU quota allocated to a Cgroup.
limits	bigint	Limit of CPUs allocated to a Cgroup.
wdlevel	bigint	Workload Cgroup level.
cpucores	text	Usage of CPU cores in a Cgroup.

8.2.17.4 DBE_PERF.WLM_CONTROLGROUP_NG_CONFIG

Displays information about all node groups in the database, as described in **Table 8-203**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-203 DBE_PERF.WLM_CONTROLGROUP_NG_CONFIG columns

Name	Type	Description
group_name	text	Cgroup name.
group_type	text	Cgroup type.
gid	bigint	Cgroup ID.
classgid	bigint	ID of the **Class** Cgroup to which a **Workload** Cgroup belongs.
class	text	Class Cgroup.
group_workload	text	Workload Cgroup.
shares	bigint	CPU quota allocated to a Cgroup.
limits	bigint	Limit of CPUs allocated to a Cgroup.
wdlevel	bigint	Workload Cgroup level.
cpucores	text	Usage of CPU cores in a Cgroup.
group_kind	text	Node group type. Its value can be **i**, **n**, **v**, or **e**. • **i**: installation node group. • **n**: node group in a common cluster. • **e**: elastic cluster.

8.2.17.5 DBE_PERF.WLM_RESOURCEPOOL_RUNTIME

Displays statistics about a resource pool, as described in **Table 8-204**.

Table 8-204 DBE_PERF.WLM_RESOURCEPOOL_RUNTIME columns

Name	Type	Description
rpoid	oid	OID of the resource pool.
respool	name	Name of the resource pool.
control_group	name	Cgroup associated with the resource pool.
parentid	oid	OID of the parent resource pool.
ref_count	integer	Number of jobs associated with the resource pool.
active_points	integer	Number of used points in the resource pool.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
running_count	integer	Number of jobs running in the resource pool.
waiting_count	integer	Number of jobs queuing in the resource pool.
io_limits	integer	IOPS upper limit of the resource pool.
io_priority	integer	I/O priority of the resource pool.

8.2.17.6 DBE_PERF.WLM_USER_RESOURCE_CONFIG

Displays the resource configuration information of a user, as described in **Table 8-205**.

Table 8-205 DBE_PERF.WLM_USER_RESOURCE_CONFIG columns

Name	Type	Description
userid	oid	OID of the user.
username	name	Username.
sysadmin	boolean	Specifies whether the user has the SYSADMIN permission.
rpoid	oid	OID of the resource pool.
respool	name	Name of the resource pool.
parentid	oid	OID of the parent user.
totalspace	bigint	Size of the occupied space.
spacelimit	bigint	Upper limit of the space size.
childcount	integer	Number of child users.
childlist	texto	Child user list.

8.2.17.7 DBE_PERF.WLM_USER_RESOURCE_RUNTIME

Displays resource usage of all users. Only administrators can query this view, as described in **Table 8-206**. This view is valid only when the GUC parameter **use_workload_manager** is set to **on**.

Table 8-206 DBE_PERF.WLM_USER_RESOURCE_RUNTIME columns

Name	Type	Description
username	name	Username.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
used_memory	integer	Size of the memory being used (unit: MB).
total_memory	integer	Available memory (unit: MB). The value **0** indicates that the available memory is not limited and depends on the maximum memory available in the database.
used_cpu	integer	Number of CPU cores in use.
total_cpu	integer	Total number of CPU cores of the Cgroup associated with the user on the node.
used_space	bigint	Used storage space (unit: KB).
total_space	bigint	Available storage space (unit: KB). The value **-1** indicates that the space is not limited.
used_temp_space	bigint	Used temporary space (reserved column; unit: KB).
total_temp_space	bigint	Available temporary space (reserved column; unit: KB). The value **-1** indicates that the maximum temporary storage space is not limited.
used_spill_space	bigint	Used space for storing spilled data (reserved column; unit: KB).
total_spill_space	bigint	Available space for storing spilled data (reserved column; unit: KB). The value **-1** indicates that the maximum space for storing spilled data is not limited.

8.2.17.8 DBE_PERF.WLM_WORKLOAD_HISTORY_INFO

Displays information about workload management after a job is complete or the exception has been handled, as described in **Table 8-207**.

Table 8-207 DBE_PERF.WLM_WORKLOAD_HISTORY_INFO columns

Name	Type	Description
statement	text	Statement executed for exception handling.
block_time	bigint	Block time before the statement is executed.
elapsed_time	bigint	Elapsed time when the statement is executed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
total_cpu_time	bigint	Total time used by the CPU on the DN when the statement is executed for exception handling.
qualification_time	bigint	Period when the statement checks the skew.
cpu_skew_percent	integer	CPU usage skew on the DN when the statement is executed for exception handling.
control_group	text	Cgroup used when the statement is executed for exception handling.
status	text	Statement status after statement are executed for exception handling, including: • **pending**: waiting to be executed. • **running**: being executed. • **finished**: finished normally. • **abort**: terminated unexpectedly.
action	text	Actions when statements are executed for exception handling, including: • **abort**: terminating the operation. • **adjust**: executing the Cgroup adjustment operations. Currently, you can only perform the demotion operation. • **finish**: finished normally.

8.2.17.9 DBE_PERF.WLM_WORKLOAD_RUNTIME

Displays the status of jobs being executed by the current user on the current CN, as described in **Table 8-208**.

Table 8-208 DBE_PERF.WLM_WORKLOAD_RUNTIME columns

Name	Type	Description
node_name	text	Name of the CN where the job is executed.
thread_id	bigint	Backend thread ID.
processid	integer	PID of the backend thread.
time_stamp	bigint	Time when the statement starts to be executed.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
username	name	Username for logging in to the backend.
memory	integer	Memory required by the statement.
active_points	integer	Number of resources consumed by the statement in the resource pool.
max_points	integer	Number of resources consumed by the statement in the resource pool.
priority	integer	Job priority.
resource_pool	text	Resource pool to which the job belongs.
status	text	Job execution status. Its value can be: • **pending**: blocked. • **running**: running. • **finished**: ended. • **aborted**: aborted. • **unknown**: unknown.
control_group	text	Cgroups used by the job.
enqueue	text	Queue that the job is in. Its value can be: • **GLOBAL**: global queue. • **RESPOOL**: resource pool queue. • **ACTIVE**: not in a queue.
query	text	Statement being executed.
node_group	text	Node group name.

8.2.17.10 DBE_PERF.GLOBAL_WLM_WORKLOAD_RUNTIME

Displays the status of jobs executed by the current user on CNs, as described in Table 8-209. This view is accessible only to users with the MONADMIN permission.

Table 8-209 DBE_PERF.GLOBAL_WLM_WORKLOAD_RUNTIME columns

Name	Type	Description
node_name	text	Name of the CN where the job is executed.
thread_id	bigint	Backend thread ID.
processid	integer	LWP ID of the thread.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
time_stamp	bigint	Time when the statement starts to be executed.
username	name	Username for logging in to the backend.
memory	integer	Memory required by the statement.
active_points	integer	Number of resources consumed by the statement in the resource pool.
max_points	integer	Maximum number of resources in the resource pool.
priority	integer	Job priority.
resource_pool	text	Resource pool to which the job belongs.
status	text	Job execution status. Its value can be: • **pending**: blocked status. • **running**: running status. • **finished**: final status. • **aborted**: termination status. • **unknown**: unknown status.
control_group	name	Cgroups used by the job.
enqueue	text	Queue that the job is in. Its value can be: • **GLOBAL**: global queue. • **RESPOOL**: resource pool queue. • **ACTIVE**: not in a queue.
query	text	Statement being executed.
node_group	text	Node group name.

8.2.17.11 DBE_PERF.LOCAL_IO_WAIT_INFO

Returns the real-time statistics of I/O control on the current node, as described in Table 8-210.

Table 8-210 DBE_PERF.LOCAL_IO_WAIT_INFO columns

Name	Type	Description
node_name	text	Node name.
device_name	text	Name of the data disk mounted to the node.
read_per_second	double precision	Number of read completions per second.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
write_per_second	double precision	Number of write completions per second.
write_ratio	double precision	Ratio of the disk write I/Os to the total I/Os.
io_util	double precision	Percentage of the I/O time to the total CPU time per second.
total_io_util	integer	Level of the CPU time occupied by the last three I/Os. The value ranges from 0 to 6.
tick_count	integer	Interval for updating disk I/O information. The value is fixed to 1 second. The value is cleared each time before data is read.
io_wait_list_len	integer	Size of the I/O request thread wait queue. If the value is **0**, no I/O is under control.

8.2.17.12 DBE_PERF.GLOBAL_IO_WAIT_INFO

Returns the real-time statistics of I/O control on all nodes, as described in **Table 8-211**.

Table 8-211 DBE_PERF.GLOBAL_IO_WAIT_INFO columns

Name	Type	Description
node_name	text	Node name.
device_name	text	Name of the data disk mounted to the node.
read_per_second	double precision	Number of read completions per second.
write_per_second	double precision	Number of write completions per second.
write_ratio	double precision	Ratio of the disk write I/Os to the total I/Os.
io_util	double precision	Percentage of the I/O time to the total CPU time per second.
total_io_util	integer	Level of the CPU time occupied by the last three I/Os. The value ranges from 0 to 6.
tick_count	integer	Interval for updating disk I/O information. The value is fixed to 1 second. The value is cleared each time before data is read.
io_wait_list_len	integer	Size of the I/O request thread wait queue. If the value is **0**, no I/O is under control.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.18 Global Plan Cache

Global plan cache (GPC) views are valid only when the GUC parameter **enable_global_plancache** is enabled.

8.2.18.1 DBE_PERF.LOCAL_PLANCACHE_STATUS

Displays the status of the GPC plan cache on the current node, as described in Table 8-212. To query this view, you must have the MONADMIN permission.

Table 8-212 DBE_PERF.LOCAL_PLANCACHE_STATUS columns

Name	Type	Description
nodename	text	Name of the node that the plan cache belongs to.
query	text	Text of query statements.
refcount	integer	Number of times that the plan cache is referenced.
valid	bool	Specifies whether the plan cache is valid.
databaseid	oid	ID of the database that the plan cache belongs to.
schema_name	text	Schema that the plan cache belongs to.
params_num	integer	Number of parameters.
func_id	oid	OID of the stored procedure where the plan cache is located. If the plancache does not belong to the stored procedure, the value is **0**.
pkg_id	oid	Package to which the stored procedure where the plancache is located belongs. If the stored procedure does not belong to a package, the value is **0**.
stmt_id	integer	Sequence number of the statement plan in the stored procedure.
size	bigint	Size of the memory occupied by the plan.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
parameterized_info	text	This column specifies the path for saving the plan to the GPC. • Automatic parameterization paths: Forcibly bind parameters (the value is **force**) or reuse a plan for the same statement (the value is **exact**). • Non-automatic parameterization path: The value is **normal**.
sql_hash	bigint	Unique ID of the SQL statement in the SPM. In distributed mode, the value is **0**.
plan_hash	bigint	Unique ID of the plan under the current SQL statement in the SPM. In distributed mode, the value is **0**.
unique_sql_id	bigint	Unique ID of an SQL statement in the database.
outline	text	The combination character string of all hints of the outline corresponding to the current plan. In distributed mode, the value is empty.
last_used_time	timestamp with time zone	Last time when the plan cache was used. The last use time is not recorded for invalid plans.
depend_object_ids	text	List of OID objects on which the plan cache depends.
hash_value	bigint	Hash value of a parameter, which may not be unique.
plan_info	text	Displays detailed information about valid plans. By default, this column is empty in the view.
opcache_reuse_cnt	integer	Number of times that the execution cache is reused.
opcache_mem_size	bigint	Memory size occupied by the execution cache.

8.2.18.2 DBE_PERF.GLOBAL_PLANCACHE_STATUS

Displays the status of GPC plan caches on all nodes. To query this view, you must have the MONADMIN permission.

For details about the columns, see **DBE_PERF.LOCAL_PLANCACHE_STATUS**.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.18.3 DBE_PERF.LOCAL_PREPARE_STATEMENT_STATUS

Displays the information about prepared statements corresponding to the GPC on the current node, as described in **Table 8-213**.

Table 8-213 LOCAL_PREPARE_STATEMENT_STATUS columns

Name	Type	Description
nodename	text	Name of the node that the statement belongs to.
cn_sess_id	bigint	Session ID of the CN that the statement is sent from.
cn_node_id	integer	Node ID of the CN that the statement is sent from.
cn_time_line	integer	Number of restart times of the CN that the statement is sent from.
statement_name	text	Statement name.
refcount	integer	Number of times that the corresponding plan cache is referenced.
is_shared	bool	Specifies whether the corresponding plan cache is shared.
query	text	Corresponding query statement.

8.2.18.4 DBE_PERF.GLOBAL_PREPARE_STATEMENT_STATUS

Displays information about prepared statements corresponding to GPC plan caches on all nodes.

For details about the columns, see **Table 8-213**.

8.2.18.5 DBE_PERF.LOCAL_PLANCACHE_CLEAN

Clears the unused global plan cache on the current node. The return type is Boolean.

After a cluster is created, by default, you must have the MONADMIN permission to query this view.

8.2.18.6 DBE_PERF.GLOBAL_PLANCACHE_CLEAN

Clears the unused global plan cache on all nodes. The return value is of the Boolean type.

After a cluster is created, by default, you must have the MONADMIN permission to query this view.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.19 RTO & RPO

8.2.19.1 DBE_PERF.GLOBAL_RTO_STATUS

Displays log flow control information about the primary and standby nodes (except the current node and DNs), as described in **Table 8-214**.

Table 8-214 DBE_PERF.GLOBAL_RTO_STATUS columns

Parameter	Type	Description
node_name	text	Node name (including the primary and standby nodes).
rto_info	text	Flow control information, including the current log flow control time (unit: second) of the standby node, the expected flow control time (unit: second) specified by the GUC parameter, and the primary node sleep time (unit: μs) required to reach the expectation.

8.2.19.2 DBE_PERF.GLOBAL_STREAMING_HADR_RTO_AND_RPO_STAT

Displays the log flow control information about the primary and standby clusters for streaming DR, as described in **Table 8-215**. (This view can be used only on the CN in the primary cluster and cannot obtain statistics from the DN or standby cluster.)

Table 8-215 DBE_PERF.GLOBAL_STREAMING_HADR_RTO_AND_RPO_STAT columns

Parameter	Type	Description
hadr_sender_node_name	text	Name of the primary cluster and the main standby node in the standby cluster.
hadr_receiver_node_name	text	Name of the main standby node in the standby cluster.
current_rto	int	Flow control information, that is, log RTO time of the current primary and standby clusters (unit: second). **-1** indicates that the majority of the standby cluster for DR is abnormal.
target_rto	int	Flow control information, that is, RTO time between the target primary and standby clusters (unit: second).
current_rpo	int	Flow control information, that is, log RPO time of the current primary and standby clusters (unit: second). **-1** indicates that the majority of the standby cluster for DR is abnormal.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
target_rpo	int	Flow control information, that is, RPO time between the target primary and standby clusters (unit: second).
rto_sleep_time	int	RTO flow control information, that is, expected sleep time (unit: μs) required by WAL sender on the host to reach the specified RTO.
rpo_sleep_time	int	RPO flow control information, that is, the expected sleep time (unit: μs) required by xlogInsert on the host to reach the specified RPO.

8.2.20 AI Watchdog

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.20.1 DBE_PERF.AI_WATCHDOG_MONITOR_STATUS

Table 8-216 DBE_PERF.AI_WATCHDOG_MONITOR_STATUS columns

Parameter	Type	Description
metric_name	text	Metric names: • **tps**: TPS. • **tps_hourly**: average TPS per hour. • **shared_used_mem**: the used shared memory (MB). • **dynamic_used_shrctx**: the used shared memory context (MB). • **other_used_mem**: other used memory (MB). • **process_used_mem**: the used resident memory (MB). • **dynamic_used_mem**: the used dynamic memory (MB). • **malloc_failures**: the number of memory allocation failures in each collection interval. • **D_state_rate**: percentage of threads in the D state. • **R_state_rate**: percentage of threads in the R state. • **S_state_rate**: ratio of threads in the S state. • **db_state**: database state (**68** indicates D, **82** indicates R, and **83** indicates S). • **cpu_usage**: CPU usage. The upper limit is 100. • **disk_io**: disk I/O delay between two collection intervals. • **network_io**: network I/O delay between two collection intervals. • **threadpool_usage**: thread pool usage. • **threadpool_hang_rate**: percentage of the thread pool group in the hang state.
max_length	int	Collection queue length.
current_length	int	The number of currently collected samples.
collection_interval	int	Collection interval, in seconds.
latest_value	int	Value collected last time. If no value is collected, the value is **null**.
last_report	timestamp	The last collection time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.20.2 DBE_PERF.AI_WATCHDOG_DETECTION_WARNINGS

Table 8-217 DBE_PERF.AI_WATCHDOG_DETECTION_WARNINGS columns

Parameter	Type	Description
event	text	Event name.
cause	text	Event cause.
details	text	Event details.
time	timestamp	Reporting time.
need_to_handle	bool	Determines whether automatic processing is required.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.2.20.3 DBE_PERF.AI_WATCHDOG_PARAMETERS

Table 8-218 DBE_PERF.AI_WATCHDOG_PARAMETERS columns

Parameter	Type	Description
name	text	Parameter name. The options are as follows: • **enable_ai_watchdog**: determines whether to enable this function. • **ai_watchdog_max_consuming_time_ms**: specifies the maximum duration. • **ai_watchdog_used_memory_kb**: specifies the memory used by this function. • **ai_watchdog_detection_times**: specifies the number of detection times. • **enable_self_healing**: determines whether self-healing can be performed after a problem is detected. • **oom_detected_times**: specifies the number of detected OOMs. • **hang_detected_times**: specifies the number of detected hang times. • **enable_oom_detection**: specifies whether the OOM detection function is automatically enabled. • **in_wait_time**: specifies whether the system is waiting. • **other_used_memory_has_risk**: specifies whether other memory usage is risky. • **shared_used_mem_has_risk**: specifies whether risks exist when the shared memory context is used. • **dynamic_used_shrctx_has_risk**: specifies whether the dynamic memory usage is risky.
value	text	Parameter value.

8.2.20.4 DBE_PERF.AI_WATCHDOG_FTASK_STATUS

Table 8-219 DBE_PERF.AI_WATCHDOG_FTASK_STATUS columns

Parameter	Type	Description
name	text	Name of the counter clearing task.
timeout_threshold	int	Timeout threshold of the counter clearing task.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
register_time	timestamp	Registration time of the counter clearing task.
thread_id	bigint	ID of the thread on which the counter clearing task is used.
thread_name	text	Name of the thread on which the counter clearing task is used.
last_feed	timestamp	Last time when the counter clearing task is executed. The default initial value is the same as the value of **register_time**.
restart_count	int	Number of times that the same name of counter clearing task is repeatedly registered due to the same background thread exception. This parameter is cleared after the task calls the API for canceling counter clearing task registration.
timeout_action	text	Timeout processing action specified during task registration. Currently, the following actions are provided: • **WATCHDOG_REPORT_WARNING**: Prompt the timeout locating and demarcation. • **WATCHDOG_SEND_SIGTERM**: Send SIGTERM signals to threads. • **WATCHDOG_SEND_SIGUSR2**: Send SIGUSR2 signals to threads. • **WATCHDOG_SEND_SIGQUIT**: Send SIGQUIT signals to threads. • **WATCHDOG_EXIT_PROCESS**: Exit the process.

8.2.21 Trace Function

8.2.21.1 DBE_PERF.pl_trace_func_start

Function for enabling the trace feature of a node. For details about the columns, see **Table 8-220**.

Table 8-220 DBE_PERF.pl_trace_func_start input parameters

Parameter	Type	Description
session_id	bigint	ID of the session for which the trace function needs to be enabled.
node	text	Name of the node on which the trace function is to be enabled.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
max_num	integer	Maximum number of messages that can be traced. The value ranges from 100 to 10000.
level	text	Trace level. • **session**: session level. The trace function is always enabled before the pl_trace_func_end function is called or the session is closed. • **portal**: portal level. The trace function is automatically disabled after a stored procedure is executed. • **time**: session level. The statement execution time is displayed at the session level. • **threshold**: session level. At the time level, only statements whose values are greater than or equal to the threshold are recorded.
threshold	float8	Threshold of the current trace, in milliseconds. This parameter is valid only when **level** is **threshold**.

8.2.21.2 DBE_PERF.pl_trace_func_end

Function for disabling the trace function of a node. For details about the columns, see **Table 8-221**.

Table 8-221 DBE_PERF.pl_trace_func_end input parameters

Parameter	Type	Description
session_id	bigint	ID of the session for which the trace function needs to be disabled.
node	text	Name of the node on which the trace function is to be disabled.

8.2.21.3 DBE_PERF.pl_trace_info_clean

Function for clearing trace records of a node. For details about the columns, see **Table 8-222**.

Table 8-222 DBE_PERF.pl_trace_info_clean input parameters

Parameter	Type	Description
node	text	Name of the node whose trace records need to be cleared.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
session_id	bigint	ID of the session whose trace information needs to be deleted. If this parameter is left empty, all information is deleted.

8.2.21.4 DBE_PERF.pl_get_trace_info

Function for checking trace records of the current node. For details about columns, see **Table 8-223**.

Table 8-223 DBE_PERF.pl_get_trace_info return values

Parameter	Type	Description
session_id	bigint	ID of the session to which the SQL statement belongs.
level	text	Level of the SQL statement.
database	text	Name of the database to which the SQL statement belongs.
func	oid	OID of the stored procedure to which the SQL statement belongs.
source	text	This SQL statement.
plan	text	Execution plan of this SQL statement.
time	text	Execution time of this SQL statement.
threshold	float8	Threshold of the SQL statement, in milliseconds.

8.2.21.5 DBE_PERF.pl_get_global_trace_info

Function for checking trace records of all nodes. For details about the columns, see **Table 8-224**.

Table 8-224 DBE_PERF.pl_get_global_trace_info return values

Parameter	Type	Description
session_id	bigint	ID of the session to which the SQL statement belongs.
node	text	Name of the node to which the SQL statement belongs.
level	text	Level of the SQL statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Parameter	Type	Description
database	text	Name of the database to which the SQL statement belongs.
func	oid	OID of the stored procedure to which the SQL statement belongs.
source	text	This SQL statement.
plan	text	Execution plan of this SQL statement.
time	text	Execution time of this SQL statement.
threshold	float8	Threshold of the SQL statement, in milliseconds.

8.2.22 Deprecated Views

8.2.22.1 Query

The following views in GaussDB have been deprecated in the current version:

- DBE_PERF.GS_SLOW_QUERY_HISTORY
- DBE_PERF.GLOBAL_SLOW_QUERY_HISTORY
- DBE_PERF.GLOBAL_SLOW_QUERY_INFO
- DBE_PERF.GLOBAL_STATEMENT_COMPLEX_RUNTIME

8.2.22.2 Operator

The following view in GaussDB has been deprecated in the current version:

DBE_PERF.OPERATOR_HISTORY

8.3 WDR Snapshot Schema

After the WDR snapshot function is enabled (the GUC parameter **enable_wdr_snapshot** is set to **on**), objects are created in the snapshot schema in the postgres database of the pg_default tablespace to persist WDR snapshot data. By default, the initial user or users with the MONADMIN permission can access and operate objects in the snapshot schema.

You can set the GUC parameter **wdr_snapshot_retention_days** to automatically manage the snapshot lifecycle.

8.3.1 Original Information of WDR Snapshots

8.3.1.1 SNAPSHOT.SNAPSHOT

SNAPSHOT records the index information, start time, and end time of WDR snapshots stored in the current system. The result can be queried only in the

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

system database but cannot be queried in the user database, as described in **Table 8-225**. After the WDR snapshot function is enabled (the GUC parameter **enable_wdr_snapshot** is set to **on**), the table is created. Only the initial user or MONADMIN has the permission to view the information.

Table 8-225 SNAPSHOT columns

Name	Type	Description	Example
snapshot_id	bigint	WDR snapshot ID.	1
start_ts	timestamp	Start time of a WDR snapshot.	2019-12-28 17:11:27.423742+08
end_ts	timestamp	End time of a WDR snapshot.	2019-12-28 17:11:43.67726+08
version	int	Version of a WDR snapshot.	1
snap_flag	bigint	Attribute of the WDR snapshot. • **NULL**: The snapshot is of an earlier version. • **0**: The snapshot is a full snapshot. • **1**: The snapshot is an incremental snapshot.	0
base_snapshot_id	bigint	ID of the full snapshot corresponding to the WDR snapshot.	1

8.3.1.2 SNAPSHOT.TABLES_SNAP_TIMESTAMP

TABLES_SNAP_TIMESTAMP records the databases, table objects, and start and end time of all WDR snapshots, as described in **Table 8-226**. After the WDR snapshot function is enabled (the GUC parameter **enable_wdr_snapshot** is set to **on**), the table is created. Only the initial user or users with MONADMIN permissions has the permission to view the information.

Table 8-226 TABLES_SNAP_TIMESTAMP columns

Name	Type	Description	Example
snapshot_id	bigint	WDR snapshot ID.	1
db_name	text	Database corresponding to a WDR snapshot.	tpcc1000

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description	Example
tablename	text	Table corresponding to a WDR snapshot.	snap_summary_statement
start_ts	timestamp	Start time of a WDR snapshot.	2019-12-28 17:11:27.425849+08
end_ts	timestamp	End time of a WDR snapshot.	2019-12-28 17:11:27.707398+08

8.3.1.3 SNAPSHOT.SNAP_SEQ

SNAP_SEQ is an ascending sequence, which provides IDs for WDR snapshots.

After the WDR snapshot is started (the GUC parameter **enable_wdr_snapshot** is enabled), the sequence is created.

8.3.1.4 SNAPSHOT.SNAP_QUERY

The SNAP_QUERY table records information about SQL statements in all stored WDR snapshots. After a WDR snapshot is started (that is, the GUC parameter **enable_wdr_snapshot** is set to **on**), the SNAP_QUERY table is created, as shown in **Table 8-227**.

Table 8-227 SNAP_QUERY columns

Name	Type	Description
snap_node_name	name	Node name.
snap_node_id	integer	Node ID.
snap_user_name	name	Username. If the user who runs the query statement does not exist, ***REMOVED_USER*** is displayed.
snap_user_id	oid	User OID.
snap_unique_sql_id	bigint	ID of the normalized SQL statement.
snap_query	text	Normalized SQL statement. Note: The length is controlled by the GUC parameter **instr_unique_sql_size**.
snap_last_updated	timestamp with time zone	Last time when the statement was updated.
snap_parent_unique_sql_id	bigint	Unique ID of the parent SQL statement. The value is **0** for a non-stored procedure substatement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
snap_db_name	name	Name of the database connected to the backend. If the database where the query statement is executed does not exist, ***REMOVED_DB*** is displayed.
snap_db_id	oid	OID of the database that the backend is connected to.

8.3.1.5 SNAPSHOT.SNAP_PLAN

The SNAP_PLAN table records information about SQL statements in all stored WDR snapshots. After a WDR snapshot is started (that is, the GUC parameter **enable_wdr_snapshot** is set to **on**), the SNAP_PLAN table is created, as shown in **Table 8-228**.

Table 8-228 SNAP_PLAN columns

Name	Type	Description
snap_node_id	integer	Node ID.
snap_user_id	oid	User OID.
snap_unique_sql_id	bigint	ID of a normalized SQL statement.
snap_last_updated	timestamp with time zone	Last time when a statement was updated.
snap_db_id	oid	OID of a database that the backend is connected to.
snap_sql_hash	bigint	Hash value of the SQL statement, which is **0**.
snap_plan_hash	bigint	Hash value of the execution plan of the SQL statement, which is **0**.
snap_query_plan	text	Execution plan of the SQL statement.

8.3.2 WDR Snapshot Data Table

The naming rule of a WDR snapshot data table is **snap**_*{source data table}*.

The source of WDR snapshot data tables is the view under **DBE_PERF Schema**. After the WDR snapshot function is enabled (the GUC parameter **enable_wdr_snapshot** is set to **on**), the WDR snapshot data table is created.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

NOTE

The initial user or users with the MONADMIN permission can view the WDR snapshot data table.

8.4 DBE_PLDEBUGGER Schema

System functions in the DBE_PLDEBUGGER schema are used to debug stored procedures. Only administrators have the permission to execute these debugging APIs, but do not have the permission to modify or create functions. The following table describes the supported APIs.

NOTICE

- When a user is created in the function body, the plaintext password is returned when attach, next, continue, info_code, step, info_breakpoint, backtrace, or finish is called. You are advised not to create a user in the function body.
- Stored procedures can be debugged only on CNs, and the debugging end and the debugged end must be connected to the same CN.
- The debugging end and the debugged end must be connected to the same database. Otherwise, the API that uses the function OID to obtain function information is unavailable.
- After you use the dbe_pldebugger.turn_on function to enable stored procedure debugging in the current session, all PL/SQL stored procedures of the current session cannot be pushed down to DNs. Therefore, after the debugging is complete, use the dbe_pldebugger.turn_off function to disable stored procedure debugging in the current session in a timely manner.
- Triggers cannot be directly debugged. EXECUTE DIRECT ON cannot be used to deliver statements to DNs to trigger debugging.
- Prepared statements generated before the dbe_pldebugger.turn_on function is used to enable stored procedure debugging may fail to be debugged.
- The dbe_pldebugger.error_* functions can be used only when an error is reported.
- During the debugging of a stored procedure, if the stored procedure to be debugged calls lock operations, do not perform operations that may cause deadlocks on the debugging end.
- When separation of duties is enabled, only the initial user and owner can debug a function with the definer permission. When separation of duties is disabled, the system administrator, owner, and initial user can debug a function with the definer permission.

The administrator can run the following command to grant the gs_role_pldebugger role and debugger permissions to a user:

```
GRANT gs_role_pldebugger to user;
```

Two clients are required to connect to the database. One client is responsible for executing the debugging API as the debug end, and the other client is responsible for executing the debugging function to control the execution of stored procedures on the server. Example:

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

- Prepare for debugging.

  Use PG_PROC to find the OID of the stored procedure to be debugged and execute **DBE_PLDEBUGGER.turn_on(oid)**. In this case, the client functions as the server.

```
gaussdb=# CREATE OR REPLACE PROCEDURE test_debug ( IN  x INT)
AS
BEGIN
    INSERT INTO t1 (a) VALUES (x);
    DELETE FROM t1 WHERE a = x;
END;
/
CREATE PROCEDURE
gaussdb=# SELECT OID FROM PG_PROC WHERE PRONAME='test_debug';
  oid
-------
 16389
(1 row)
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.turn_on(16389);
 nodename | port
----------+------
 datanode |    0
(1 row)
```

- Start debugging.

  When the server executes the stored procedure, the server hangs before the first SQL statement in the stored procedure and waits for the debugging message sent by the debug end. Debugging is supported only by directly executing a stored procedure and cannot be achieved by calling an executed stored procedure through a trigger.

```
gaussdb=# call test_debug(1);
```

  Start another client as the debug end and call DBE_PLDEBUGGER.attach to attach with the stored procedure for debugging based on the data returned by **turn_on**.

```
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.attach('datanode',0);
 funcoid |  funcname  | lineno |              query
---------+------------+--------+----------------------------------
   16389 | test_debug |      3 |   INSERT INTO t1 (a) VALUES (x);
(1 row)
```

  Execute the next statement on the client where the attach operation is performed.

```
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.next();
 funcoid |  funcname  | lineno |        query
---------+------------+--------+----------------------
   16389 | test_debug |      0 | [EXECUTION FINISHED]
(1 row)
```

  Execute the following variable operations on the client where the attach command is performed.

```
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.info_locals(); -- Print all variables.
 varname | vartype | value | package_name | isconst
---------+---------+-------+--------------+---------
 x       | int4    | 1     |              | f
(1 row)
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.set_var('x', 2); -- Assign a value to a variable.
 set_var
---------
 t
(1 row)
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.print_var('x'); -- Print a single variable.
 varname | vartype | value | package_name | isconst
---------+---------+-------+--------------+---------
 x       | int4    | 2     |              | f
(1 row)
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Directly execute the stored procedure that is being debugged.

```
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.continue();
 funcoid |  funcname  | lineno |        query
---------+------------+--------+----------------------
   16389 | test_debug |      0 | [EXECUTION FINISHED]
(1 row)
```

When the stored procedure reports an error, the following information is displayed. In this case, the suspended process logic is triggered.

```
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.continue();
 funcoid |  funcname  | lineno |            query
---------+------------+--------+------------------------------
   16389 | test_debug |      0 | [EXECUTION HAS ERROR OCCURRED!]
(1 row)
```

If the suspended process logic is triggered, you can call the error_info_locals,error_backtrace, error_end, and print_var APIs to view information. Other APIs cannot be used anymore. You need to use error_end to end the suspended process logic.

```
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.error_end();
 funcoid |  funcname  | lineno |      query
---------+------------+--------+------------------
   16389 | test_debug |      0 | [END HANG ERROR!]
(1 row)
```

Exit the stored procedure that is being debugged (the suspended process logic is unavailable), and do not execute statements that have not been executed before.

```
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.abort();
 abort
-------
 t
(1 row)
```

View the code information on the client and identify the line number of the breakpoint that can be set.

```
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.info_code(16389);
 lineno |                          query                          | canbreak
--------+---------------------------------------------------------+----------
        | CREATE OR REPLACE PROCEDURE public.test_debug( IN  x INT) | f
      1 | AS  DECLARE                                               | f
      2 | BEGIN                                                     | f
      3 |     INSERT INTO t1 (a) VALUES (x);                        | t
      4 |     DELETE FROM t1 WHERE a = x;                           | t
      5 | END;                                                      | f
      6 | /                                                         | f
(7 rows)
```

Sets a breakpoint.

```
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.add_breakpoint(16389,4);
 breakpointno
--------------
            0
(1 row)
```

View the breakpoint information.

```
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.info_breakpoints();
 breakpointno | funcoid | lineno |             query             | enable
--------------+---------+--------+-------------------------------+--------
            0 |   16389 |      4 |     DELETE FROM t1 WHERE a = x; | t
(1 row)
```

Execute to the breakpoint.

```
gaussdb=# SELECT * FROM DBE_PLDEBUGGER.continue();
 funcoid |  funcname  | lineno |             query
---------+------------+--------+---------------------------------
```

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

```
  16389 | test_debug |      4 |     DELETE FROM t1 WHERE a = x;
(1 row)
```

After the stored procedure is executed, the debugging automatically exits. To debug the stored procedure again, you need to attach again. If the server does not need to be debugged, run the **turn_off** command to disable the debugging or exit the session. **Table 8-229** describes the debugging APIs.

Table 8-229 DBE_PLDEBUGGER

API	Description
DBE_PLDEBUGGER.turn_on	Called by the server, indicating that the stored procedure can be debugged. After the API is called, the stored procedure is hung to wait for debugging information.
DBE_PLDEBUGGER.turn_off	Called by the server, indicating that debugging the stored procedure is disabled.
DBE_PLDEBUGGER.local_debug_server_info	Called by the server to print all stored procedures that have been turned on in the current session.
DBE_PLDEBUGGER.attach	Called by the debug end to attach with the stored procedure that is being debugged.
DBE_PLDEBUGGER.info_locals	Called by the debug end to print the current values of variables in the stored procedure that is being debugged.
DBE_PLDEBUGGER.next	Called by the debug end to execute the next step.
DBE_PLDEBUGGER.continue	Called by the debug end to continue the execution until the breakpoint or stored procedure ends.
DBE_PLDEBUGGER.abort	Called by the debug end to stop debugging. The server reports a long jump error.
DBE_PLDEBUGGER.print_var	Called by the debug end to print the current values of specified variables in the stored procedure that is being debugged.
DBE_PLDEBUGGER.info_code	Called by the debug end or server to print the source statement of a specified stored procedure and the line number corresponding to each line.
DBE_PLDEBUGGER.step	Called by the debug end to execute step by step.
DBE_PLDEBUGGER.add_breakpoint	Called by the debug end to add a breakpoint.
DBE_PLDEBUGGER.delete_breakpoint	Called by the debug end to delete a breakpoint.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_PLDEBUGGER.info_breakpoints	Called by the debug end to view all breakpoints.
DBE_PLDEBUGGER.backtrace	Called by the debug end to check the current call stack.
DBE_PLDEBUGGER.enable_breakpoint	Called by the debug end to enable breakpoints.
DBE_PLDEBUGGER.disable_breakpoint	Called by the debug end to disable breakpoints.
DBE_PLDEBUGGER.finish	Called by the debug end to continue the debugging until the breakpoint is reached or the upper-layer call stack is returned.
DBE_PLDEBUGGER.set_var	Called by the debug end to assign a value to a variable.

8.4.1 DBE_PLDEBUGGER.turn_on

This function is used to mark a stored procedure as debuggable, as described in Table 8-230. After **turn_on** is executed, the server can execute the stored procedure for debugging. You need to manually obtain the OID of the stored procedure based on the PG_PROC system catalog and transfer it to the function. After **turn_on** is executed, the execution of the stored procedure in the current session is hung before the first SQL statement to wait for the debugging instruction from the debug end. This setting is cleared by default after the session is disconnected. Currently, stored procedures and functions with autonomous transactions enabled cannot be debugged. The number of **turn_on** stored procedures cannot exceed 100.

The function prototype is as follows:

```
DBE_PLDEBUGGER.turn_on(Oid)
RETURN Record;
```

Table 8-230 DBE_PLDEBUGGER.turn_on input parameters and return values

Name	Type	Description
func_oid	IN oid	Function OID.
nodename	OUT text	Node name.
port	OUT integer	Port number of the connection.

8.4.2 DBE_PLDEBUGGER.turn_off

This function is used only to remove the debugging flag added by **turn_on** to the current session. The return value indicates success or failure, as described in Table

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8-231. You can run the **DBE_PLDEBUGGER.local_debug_server_info** command to query the OID of the stored procedure that has been turned on.

The function prototype is as follows:

```
DBE_PLDEBUGGER.turn_off(Oid)
RETURN boolean;
```

Table 8-231 DBE_PLDEBUGGER.turn_off input parameters and return values

Name	Type	Description
func_oid	IN oid	Function OID.
turn_off	OUT boolean	Specifies whether turn-off is successful.

8.4.3 DBE_PLDEBUGGER.local_debug_server_info

This function is used to query the OID of the stored procedure that has been turned on in the current connection. You can use funcoid and pg_proc together to determine which stored procedures are to be debugged, as described in **Table 8-232**.

Table 8-232 DBE_PLDEBUGGER.local_debug_server_info return values

Name	Type	Description
nodename	OUT text	Node name.
port	OUT bigint	Port number.
funcoid	OUT oid	Stored procedure OID.

8.4.4 DBE_PLDEBUGGER.attach

When the server executes a stored procedure, the server hangs the execution before the first statement and waits for attaching with the debug end. The debug end calls the attach function and transfers node name and port number to attach with the specified stored procedure, as described in **Table 8-233**.

If an error is reported during debugging, the attach operation automatically becomes invalid. If the debug end is attached to another stored procedure during debugging, the debugging of the attached stored procedure becomes invalid. If the attach operation is performed repeatedly, the current stored procedure is disconnected.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-233 DBE_PLDEBUGGER.attach input parameters and return values

Name	Type	Description
nodename	IN text	Node name.
port	IN integer	Connection port number.
funcoid	OUT oid	Function ID.
funcname	OUT text	Function name.
lineno	OUT integer	Number of the next line in the current debugging process.
query	OUT text	Source code of the next line of the function that is being debugged.

8.4.5 DBE_PLDEBUGGER.info_locals

During debugging on the debug end, **info_locals** is called to print the variables in the current stored procedure. The input parameter **frameno** of this function indicates the stack layer to be traversed. This function can be called without **frameno**. By default, the top-layer stack variable is queried, as described in **Table 8-234**.

Table 8-234 DBE_PLDEBUGGER.info_locals input parameters and return values

Name	Type	Description
frameno	IN integer (optional)	Specified stack layer. The default value is the top layer.
varname	OUT text	Variable name.
vartype	OUT text	Variable type.
value	OUT text	Variable value.
package_name	OUT text	Name of the package corresponding to the variable. If the variable is not a package, the value is null.
isconst	OUT boolean	Determines whether the value is a constant.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.4.6 DBE_PLDEBUGGER.next

This function is used to execute the current SQL statement in a stored procedure and return the number of the next SQL statement and the corresponding query, as described in **Table 8-235**.

Table 8-235 DBE_PLDEBUGGER.next return values

Name	Type	Description
funcoid	OUT oid	Function ID.
funcname	OUT text	Function name.
lineno	OUT integer	Number of the next line in the current debugging process.
query	OUT text	Source code of the next line of the function that is being debugged.

8.4.7 DBE_PLDEBUGGER.continue

Executes the current stored procedure until reaching the next breakpoint or end, and returns the line number of the next execution and the corresponding query, as described in **Table 8-236**.

The function prototype is as follows:

```
DBE_PLDEBUGGER.continue()
RETURN Record;
```

Table 8-236 DBE_PLDEBUGGER.continue return values

Name	Type	Description
funcoid	OUT oid	Function ID.
funcname	OUT text	Function name.
lineno	OUT integer	Number of the next line in the current debugging process.
query	OUT text	Source code of the next line of the function that is being debugged.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.4.8 DBE_PLDEBUGGER.abort

This function is used to abort the stored procedure executed on the server and report an error. The return value indicates whether the abort message is successfully sent, as described in **Table 8-237**.

The function prototype is as follows:

```
DBE_PLDEBUGGER.abort()
RETURN boolean;
```

Table 8-237 DBE_PLDEBUGGER.abort input parameters and return values

Name	Type	Description
abort	OUT boolean	Specifies whether the operation is successful.

8.4.9 DBE_PLDEBUGGER.print_var

During debugging on the debug end, print_var is called to print the name and value of the specified variable in the current stored procedure. You can also use this function to print variables when an error is reported. The input parameter **frameno** of this function indicates the stack layer to be traversed. This function can be called without **frameno**. By default, the top-layer stack variable is queried, as described in **Table 8-238**.

Table 8-238 DBE_PLDEBUGGER.print_var input parameters and return values

Name	Type	Description
var_name	IN text	Variable.
frameno	IN integer (optional)	Specified stack layer. The default value is the top layer.
varname	OUT text	Variable name.
vartype	OUT text	Variable type.
value	OUT text	Variable value.
package_name	OUT text	Package name corresponding to the variable. This parameter is reserved and is left empty currently.
isconst	OUT boolean	Determines whether the value is a constant.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.4.10 DBE_PLDEBUGGER.info_code

During debugging on the debug end, **info_code** is called to view the source statement of the specified stored procedure and the line number corresponding to each line. The line number starts from the function body, and the line number in the function header is empty, as described in **Table 8-239**.

Table 8-239 DBE_PLDEBUGGER.info_code input parameters and return values

Name	Type	Description
funcoid	IN oid	Function ID.
lineno	OUT integer	Line number.
query	OUT text	Source statement.
canbreak	OUT bool	Specifies whether the current line supports breakpoints.

8.4.11 DBE_PLDEBUGGER.step

During debugging on the debug end, if a stored procedure is being executed, the stored procedure continues to be executed and information such as the line number in the first line of the stored procedure is returned. If the executed object is not a stored procedure, the return is the same as that for **next**. After the SQL statement is executed, information such as the line number in the next line is returned, as described in **Table 8-240**.

Table 8-240 DBE_PLDEBUGGER.step input parameters and return values

Name	Type	Description
funcoid	OUT oid	Function ID.
funcname	OUT text	Function name.
lineno	OUT integer	Number of the next line in the current debugging process.
query	OUT text	Source code of the next line of the function that is being debugged.

8.4.12 DBE_PLDEBUGGER.add_breakpoint

During debugging on the debug end, call **add_breakpoint** to add a breakpoint. If **–1** is returned, the specified breakpoint is invalid. Determine a proper breakpoint position by referring to the **canbreak** column. If the outer function does not have

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

a breakpoint before entering the inner function, a breakpoint cannot be added to the outer function when the inner function is executed, as described in **Table 8-241**.

Table 8-241 DBE_PLDEBUGGER.add_breakpoint input parameters and return values

Name	Type	Description
funcoid	IN text	Function ID.
lineno	IN integer	Line number.
breakpointno	OUT integer	Breakpoint number.

8.4.13 DBE_PLDEBUGGER.delete_breakpoint

During debugging on the debug end, call delete_breakpoint to delete the existing breakpoint, as described in **Table 8-242**.

Table 8-242 DBE_PLDEBUGGER.delete_breakpoint input parameters and return values

Name	Type	Description
breakpointno	IN integer	Breakpoint number.
result	OUT bool	Specifies whether the result is successful.

8.4.14 DBE_PLDEBUGGER.info_breakpoints

During debugging on the debug end, call **info_breakpoints** to view the current function breakpoint, as described in **Table 8-243**.

Table 8-243 DBE_PLDEBUGGER.info_breakpoints return values

Name	Type	Description
breakpointno	OUT integer	Breakpoint number.
funcoid	OUT oid	Function ID.
lineno	OUT integer	Line number.
query	OUT text	Breakpoint content.
enable	OUT boolean	Specifies whether the value is valid.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.4.15 DBE_PLDEBUGGER.backtrace

During debugging on the debug end, call backtrace to view the current call stack, as described in **Table 8-244**.

Table 8-244 DBE_PLDEBUGGER.backtrace return values

Name	Type	Description
frameno	OUT integer	Call stack ID.
funcname	OUT text	Function name.
lineno	OUT integer	Line number.
query	OUT text	Breakpoint content.
funcoid	OUT oid	Function OID.

8.4.16 DBE_PLDEBUGGER.enable_breakpoint

During debugging on the debug end, calls enable_breakpoint to enable breakpoints, as described in **Table 8-245**.

Table 8-245 DBE_PLDEBUGGER.enable_breakpoint input parameters and return values

Name	Type	Description
breakpointno	IN integer	Breakpoint number.
result	OUT bool	Specifies whether the result is successful.

8.4.17 DBE_PLDEBUGGER.disable_breakpoint

During debugging on the debug end, call disable_breakpoint to disable breakpoints, as described in **Table 8-246**.

Table 8-246 DBE_PLDEBUGGER.disable_breakpoint input parameters and return values

Name	Type	Description
breakpointno	IN integer	Breakpoint number.
result	OUT bool	Specifies whether the result is successful.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

8.4.18 DBE_PLDEBUGGER.finish

Executes the current SQL statement in the stored procedure until the next breakpoint is triggered or the next line of the upper-layer stack is executed, as described in **Table 8-247**.

Table 8-247 DBE_PLDEBUGGER.finish input parameters and return values

Name	Type	Description
funcoid	OUT oid	Function ID.
funcname	OUT text	Function name.
lineno	OUT integer	Number of the next line in the current debugging process.
query	OUT text	Source code of the next line of the function that is being debugged.

8.4.19 DBE_PLDEBUGGER.set_var

Changes the variable on the top-layer stack in the specified debugging stored procedure to the value of the input parameter. If a stored procedure contains variables with the same name, set_var supports only the setting of the first variable value, as described in **Table 8-248**.

Table 8-248 DBE_PLDEBUGGER.set_var input parameters and return values

Name	Type	Description
var_name	IN text	Variable name.
value	IN text	New value.
result	OUT boolean	Specifies whether the result is successful.

8.4.20 DBE_PLDEBUGGER.error_backtrace

If the server is suspended due to an error reported by the stored procedure, you can call error_backtrace on the debug end to view the current function call stack, as described in **Table 8-249**. (This function is used only for suspension after an error is reported.)

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 8-249 DBE_PLDEBUGGER.error_backtrace return values

Name	Type	Description
frameno	OUT integer	Call stack ID.
funcname	OUT text	Function name.
lineno	OUT integer	Line number.
query	OUT text	Breakpoint content.
funcoid	OUT oid	Function OID.

8.4.21 DBE_PLDEBUGGER.error_end

If the server is suspended due to an error reported by the stored procedure, you can call error_end on the debug end to end the suspended process, end the debugging process, return the ended suspended process, and stop debugging, as described in **Table 8-250**. (This function is used only for suspension after an error is reported.)

Table 8-250 DBE_PLDEBUGGER.error_end return values

Name	Type	Description
funcoid	OUT oid	Function ID.
funcname	OUT text	Function name.
lineno	OUT integer	The next debugging line number (fixed value: **0**).
query	OUT text	Ends the suspended process and stops debugging.

8.4.22 DBE_PLDEBUGGER.error_info_locals

If the server is suspended due to an error reported by the stored procedure, you can call error_info_locals on the debug end to print the variables in the current stored procedure. The input parameter **frameno** of this function indicates the stack layer to be traversed. This function can be called without input parameters. By default, the top-layer stack variable is queried, as described in **Table 8-251**. (This function is used only for suspension after an error is reported.)

Table 8-251 DBE_PLDEBUGGER.error_info_locals return values

Name	Type	Description
frameno	IN integer (optional)	Specified stack layer. The default value is the top layer.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
varname	OUT text	Variable name.
vartype	OUT text	Variable type.
value	OUT text	Variable value.
package_name	OUT text	Name of the package corresponding to the variable. If the variable is not a package, the value is null.
isconst	OUT boolean	Determines whether the value is a constant.

8.5 DBE_PLDEVELOPER

The DBE_PLDEVELOPER system catalog records information required for compiling PL/SQL packages, functions, and stored procedures.

8.5.1 DBE_PLDEVELOPER.GS_SOURCE

Records information about PL/SQL objects (stored procedures, functions, packages, and package bodies), as described in **Table 8-252**.

NOTE

- The gs_source table records only user-defined original object statements. Even if a user uses ALTER to change the created schema or name, the information in the gs_source table does not change. If the user changes the schema or name of an object, after the user deletes the object, the object is deleted from the gs_source table.
- The owner in the gs_source table is the user who creates the table, not the user specified when the user creates the stored procedure or package.
- After the GUC parameter **behavior_compat_options** is set to **"skip_insert_gs_source"**, information about PL/SQL objects (stored procedures, functions, packages, and package bodies) is not recorded in the gs_source table.
- The permissions of gs_source are the same as those of system catalogs in pg_catalog. System administrators and common users cannot perform DROP, ALTER, DELETE, UPDATE, and INSERT operations, but can perform the SELECT operation. The GUC parameter **allow_system_table_mods** is used to control the permissions of gs_source, and the default value is **off**. After modifying this parameter, you need to restart the database for the modification to take effect.

Table 8-252 DBE_PLDEVELOPER.GS_SOURCE columns

Name	Type	Description
id	oid	Object ID.
owner	bigint	ID of the user who creates the object.
nspid	oid	Schema ID of an object.
name	name	Object name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Name	Type	Description
type	text	Object type (**procedure**/**function**/**package**/**package body**).
status	Boolean	Specifies whether the creation is successful.
src	text	Original statement for creating an object.

8.6 DBE_COMPRESSION Schema

DBE_COMPRESSION is a schema created by the ILM feature. It defines system functions for evaluating the sampling compression ratio of a specified data object or obtaining the compression type of data in a specified row based on input parameters. **Table 8-253** lists the APIs supported by DBE_COMPRESSION. Users can only use the APIs. For details, see **DBE_COMPRESSION**.

Table 8-253 DBE_COMPRESSION

API	Description
DBE_COMPRESSION.GET_COMPRESSION_RATIO	Evaluates the sampling compression ratio of a specified data object based on input parameters.
DBE_COMPRESSION.GET_COMPRESSION_TYPE	Obtains the compression type of data in a specified row based on input parameters.
DBE_COMPRESSION.GET_ACTUAL_COMPRESSION_RATIO	Evaluates the actual compression ratio of a specified data object based on input parameters.
DBE_COMPRESSION.GET_HOLE_RATIO	Evaluates the sampling hole rate of a specified cross-page transparent compression data object based on input parameters.
DBE_COMPRESSION.TURBO_COMPRESS_BUFFER_STATS	Outputs the usage of turbo_compress_shared_buffers.
DBE_COMPRESSION.GET_INDEX_COMPRESSION_TYPE	Obtains the compression type of data in a specified index page based on input parameters.

8.7 DBE_HEAT_MAP Schema

DBE_HEAT_MAP is a schema created by the ILM feature. It returns information such as the last modification time of a row in the target data block based on the input parameters. The information is used to browse the basis for determining cold and hot rows. This API is an O&M API and has no visibility check. That is, if

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

the input CTID is a deleted row, this API still returns the latest status of the current row on the page. **Table 8-254** describes the APIs supported by DBE_HEAT_MAP. Users can only use the APIs. Cross-page transparent compression does not support this schema. For details, see **DBE_HEAT_MAP**.

Table 8-254 DBE_HEAT_MAP

API	Description
DBE_HEAT_MAP.ROW_HEAT_MAP	Obtains information such as the last modification time of a row based on the schema to which the object belongs, data object name, data object partition name, and CTID.

8.8 DBE_ILM Schema

DBE_ILM is a schema created by the ILM feature. It is used to implement ILM policies, evaluate ILM policies, and stop compression jobs. **Table 8-255** lists the APIs supported by DBE_ILM. Users can only use the APIs. For details, see **DBE_ILM**.

Table 8-255 DBE_ILM

API	Description
DBE_ILM.EXECUTE_ILM	Evaluates the specified data and ILM policy based on parameters. If the evaluation is passed, the corresponding compression job is generated.
DBE_ILM.STOP_ILM	Stops a compression job that is being executed based on parameters.

8.9 DBE_ILM_ADMIN Schema

DBE_ILM_ADMIN is a schema created by the ILM feature. It is used to implement ILM policies, implement background scheduling of ADO, and control flow control parameters. **Table 8-256** lists the APIs supported by DBE_ILM_ADMIN. Users can only use the APIs. For details, see **DBE_ILM_ADMIN**.

Table 8-256 DBE_ILM_ADMIN

API	Description
DBE_ILM_ADMIN.CUSTOMIZE_ILM	Customizes ILM policy attributes based on input parameters.
DBE_ILM_ADMIN.DISABLE_ILM	Disables background scheduling.
DBE_ILM_ADMIN.ENABLE_ILM	Enables background scheduling.
DBE_ILM_ADMIN.CREATE_ILM_DB_POLICY	Creates a database-level compression policy.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
DBE_ILM_ADMIN.DELETE_ILM_DB_POLICY	Deletes a database-level compression policy.

8.10 PRVT_ILM Schema

The PRVT_ILM API is an internal API of the ILM feature and cannot be directly called by users. This section lists only the API names and does not provide detailed API prototype description.

Table 8-257 PRVT_ILM

API	Description
be_active_ado_window	Updates the ILM window triggering time and action.
be_create_ado_window_for_each_db	Creates an ILM window in each database of the instance.
be_execute_ilm	Executes an ILM window for automatic evaluation.
be_execute_ilm_dn	Automatic evaluation is performed on distributed DNs. This operation affects the overall ILM background scheduling of the cluster.
flush_task_executestate	The automatic scheduled job updates the task status.
generate_taskoid	CN generates a task ID for this scheduling.
evaluate_obj_policy	Data objects are evaluated after scheduling is triggered.
change_be_ilm	DBE_ILM_ADMIN.ENABLE_ILM() and DISABLE_ILM () help functions. This operation affects the overall ILM background scheduling of the cluster.
get_compression_ratio	DBE_COMPRESSION.GET_COMPRESSION_RATIO() help function.
get_compression_type	DBE_COMPRESSION.GET_COMPRESSION_TYPE() help function.
get_lastmodified_time	DBE_HEAT_MAP.ROW_HEAT_MAP() help function.
ilm_job_action	Compression task executor, which is used for distributed DNs.
delete_expired_ilm_log	Expired logs are cleared in distributed mode.
delete_gs_ilm_task	System catalog related to the operation feature, which is used for distributed DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
insert_gs_ilm_jobdetail	System catalog related to the operation feature, which is used for distributed DNs.
insert_gs_ilm_task	System catalog related to the operation feature, which is used for distributed DNs.
insert_gs_ilm_taskdetail	System catalog related to the operation feature, which is used for distributed DNs.
update_gs_ilm_jobdetail	System catalog related to the operation feature, which is used for distributed DNs.
update_gs_ilm_object	System catalog related to the operation feature, which is used for distributed DNs.
update_gs_ilm_param	System catalog related to the operation feature, which is used for distributed DNs. This operation affects the overall ILM background scheduling of the cluster.
update_gs_ilm_task	System catalog related to the operation feature, which is used for distributed DNs.
gs_ilm_ticker	The maintenance window performs an action, performs a dotting operation, and records the mapping between the LSN and time.
compress_blocks	PRVT_ILM.ILM_JOB_ACTION() help function.
compress_block_single	It is not supported in the current version.
get_job_status	DBE_ILM.STOP_ILM help function.
insert_gs_ilm_param	System catalogs related to the operation feature.
query_unfinishedjob_num	Queries whether there are jobs that are not complete, including jobs in the initial and running states.
update_gs_ilm_object_flag2	System catalog related to the operation feature, which is used for distributed DNs.
ilm_seq_nextval	Obtains the next sequence value of the ILM.
ilm_seq_setval	Sets the current sequence value of the ILM.
check_compatibility	Specifies a function used to determine a database compatibility mode in a unified manner.
set_pg_settings	Specifies a function used to set the GUC parameters for setting a database compatibility mode.
check_seq	Ensures that the current value of **GSILMTASK_SEQ** is greater than the maximum value of **taskoid** in the GS_ILM_TASK catalog. This parameter is used on distributed CNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

API	Description
get_actual_compression_ratio	DBE_COMPRESSION.GET_ACTUAL_COMPRESSION_RATIO() help function.
get_turbo_compress_hole	DBE_COMPRESSION.GET_HOLE_RATIO() help function.
turbo_compress_buffer_stats	DBE_COMPRESSION.TURBO_COMPRESS_BUFFER_STATS() help function.
get_index_compression_type	DBE_COMPRESSION.GET_INDEX_COMPRESSION_TYPE() help function.
create_ilm_db_policy	DBE_ILM_ADMIN.CREATE_ILM_DB_POLICY() help function.
delete_ilm_db_policy	DBE_ILM_ADMIN.DELETE_ILM_DB_POLICY() help function.
compress_indexes	PRVT_ILM.ILM_JOB_ACTION() help function.
eval_comp_relgrowth	It is not supported in the current version.
delete_gs_ilm_jobdetail	System catalog related to the operation feature, which is used for distributed DNs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

9 WDR

Table 9-1 lists the content of a WDR.

Table 9-1 Main WDR content

Item	Description
Database Stat	• Database performance statistics: transactions, read and write operations, row activities, write conflicts, and deadlocks. • Cluster-wide report, which can be viewed only in cluster mode.
Load Profile	• Cluster performance statistics: CPU time, DB time, logical or physical read operation, I/O performance, login and logout, load strength, and load performance. • Cluster-wide report, which can be viewed only in cluster mode. • This report can be viewed only in non-database-level WDRs.
Instance Efficiency Percentages	• Cluster-level or node-level cache hit ratio. • Cluster- or node-wide report, which can be viewed in cluster or node mode. • This report can be viewed only in non-database-level WDRs.
Top 10 Events by Total Wait Time	• Most time-consuming event. • Node-wide report, which can be viewed in node mode. • This report can be viewed only in non-database-level WDRs.
Wait Classes by Total Wait Time	• Category of the wait time that is most time-consuming. • Node-wide report, which can be viewed in node mode. • This report can be viewed only in non-database-level WDRs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Item	Description
Host CPU	• CPU usage of the host. • Node-wide report, which can be viewed in node mode. • This report can be viewed only in non-database-level WDRs.
IO Profile	• I/O usage in the cluster or node dimension. • Cluster- or node-wide report, which can be viewed in cluster or node mode. • This report can be viewed only in non-database-level WDRs.
Memory Statistics	• Kernel memory usage distribution. • Node-wide report, which can be viewed in node mode. • This report can be viewed only in non-database-level WDRs.
Time Model	• Time distribution of database operations. • Node-wide report, which can be viewed in node mode. • This report can be viewed only in non-database-level WDRs.
SQL Statistics	• SQL statement performance statistics are displayed in the following dimensions: total time, average time, CPU duration, lock time, total memory usage, maximum memory usage, average memory usage, number of returned rows, number of scanned rows, number of execution times, number of logical reads, and number of physical reads. • Cluster- or node-wide report, which can be viewed in cluster or node mode.
Wait Events	• Statistics on wait events at the node level. • Node-wide report, which can be viewed in node mode. • This report can be viewed only in non-database-level WDRs. For details, see "Waiting state list", "List of wait events corresponding to lightweight locks", "List of wait events corresponding to I/Os", and "List of wait events corresponding to transaction locks" in "System Catalogs and System Views > System Views > Other System Views > PG_THREAD_WAIT_STATUS" in *Reference*.
Cache IO Stats	• I/O statistics on user tables and indexes. • Cluster- or node-wide report, which can be viewed in cluster or node mode. • This report can be viewed only in non-database-level WDRs.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Item	Description
Utility status	• Status information about the replication slot and background checkpoint. • Node-wide report, which can be viewed in node mode. • This report can be viewed only in non-database-level WDRs.
Object stats	• Performance statistics in the index and table dimensions. • Cluster- or node-wide report, which can be viewed in cluster or node mode.
Configuration settings	• GUC parameter configuration information of a node: parameter name, current value, and default value. • Node-wide report, which can be viewed in node mode. • This report can be viewed only in non-database-level WDRs.
SQL Detail	• SQL statement text details. • Execution plan text details of the SQL statement. • Cluster- or node-wide report, which can be viewed in cluster or node mode.

9.1 Database Stat

Table 9-2 describes columns in the Database Stat report.

Table 9-2 Columns in the Database Stat report

Column	Description
DB Name	Database name.
Backends	Number of backends connected to this database.
Xact Commit	Number of transactions in this database that have been committed.
Xact Rollback	Number of transactions in this database that have been rolled back.
Blks Read	Number of disk blocks read in this database.
Blks Hit	Number of times that disk blocks have been found in the cache.
Tuple Returned	Number of rows sequentially scanned.
Tuple Fetched	Number of rows randomly scanned.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Tuple Inserted	Number of rows inserted by queries in this database.
Tuple Updated	Number of rows updated by queries in this database.
Tup Deleted	Number of rows deleted by queries in this database.
Conflicts	Number of queries canceled due to conflicts.
Temp Files	Number of temporary files created by queries in this database.
Temp Bytes	Total amount of data written to temporary files by queries in this database.
Deadlocks	Number of deadlocks detected in this database.
Blk Read Time	Time spent in reading data file blocks by backends in this database (unit: ms).
Blk Write Time	Time spent in writing data file blocks by backends in this database (unit: ms).
Stats Reset	Time at which the current statistics were reset.

9.2 Load Profile

Table 9-3 lists metrics in the Load Profile report.

Table 9-3 Metrics in the Load Profile report

Statistical Item	Description
DB Time(us)	Total elapsed time of a job.
CPU Time(us)	Total CPU time used for job running.
Redo size(blocks)	Size of the generated WAL (blocks).
Logical read (blocks)	Number of logical reads for a table or an index (number of blocks).
Physical read (blocks)	Number of physical reads (blocks) on a table or an index.
Physical write (blocks)	Number of physical writes (blocks) on a table or an index.
Read IO requests	Number of reads for a table or an index.
Write IO requests	Number of writes for a table or an index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Statistical Item	Description
Read IO (MB)	Size of reads for a table or an index (in MB).
Write IO (MB)	Size of writes for a table or an index (in MB).
Logins	Number of logins.
Executes (SQL)	Number of times SQL statements are executed.
Rollbacks	Number of rolled-back transactions.
Transactions	Number of transactions.
n_soft_parse	Number of soft parsing times.
n_hard_parse	Number of hard parsing times.
SQL response time P95(us)	Response time of 95% SQL queries, in microseconds.
SQL response time P80(us)	Response time of 80% SQL queries, in microseconds.

9.3 Instance Efficiency Percentages

Table 9-4 lists metrics in the Instance Efficiency Percentages report.

Table 9-4 Metrics in the Instance Efficiency Percentages report

Statistical Item	Description
Buffer Hit %	Hit ratio of the buffer pool.
Effective CPU %	Ratio of the CPU time to the database time.
WalWrite NoWait %	Ratio of the number of events that access the WAL buffer to the total number of wait events.
Soft Parse %	Ratio of the number of soft parsing times to the total number of parsing times.
Non-Parse CPU %	Ratio of the non-parse time to the total execution time.

9.4 Top 10 Events by Total Wait Time

Table 9-5 lists columns in the Top 10 Events by Total Wait Time report.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 9-5 Contents of the Top 10 Events by Total Wait Time report

Column	Description
Event	Name of a wait event.
Waits	Number of wait times.
Total Wait Time(us)	Total wait time, in microseconds.
Avg Wait Time(us)	Average wait time, in microseconds.
Type	Wait event type.

9.5 Wait Classes by Total Wait Time

Table 9-6 lists columns in the Wait Classes by Total Wait Time report.

Table 9-6 Contents of the Wait Classes by Total Wait Time report

Column	Description
Type	Wait events are classified as follows: • **STATUS** • **LWLOCK_EVENT** • **LOCK_EVENT** • **IO_EVENT**
Waits	Number of wait times.
Total Wait Time(us)	Total wait time, in microseconds.
Avg Wait Time(us)	Average wait time, in microseconds.

9.6 Host CPU

Table 9-7 describes columns in the Host CPU report.

Table 9-7 Columns in the Host CPU report

Column	Description
Cpus	Number of CPUs.
Cores	Number of CPU cores.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Sockets	Number of CPU sockets.
Load Average Begin	Average load of the start snapshot.
Load Average End	Average load of the end snapshot.
%User	Percentage of CPU time occupied when the system is running in user mode.
%System	Percentage of CPU time occupied when the system is running in kernel mode.
%WIO	Percentage of CPU time occupied when the system is running in wait I/O mode.
%Idle	Percentage of CPU time occupied when the system is running in idle mode.

9.7 IO Profile

Table 9-8 lists metrics in the IO Profile report.

Table 9-8 Metrics in the IO Profile report

Statistical Item	Description
Database requests	Number of database I/O times.
Database (MB)	Database I/O data volume.
Database (blocks)	Number of database I/O data blocks.
Redo requests	Number of redo I/O times.
Redo (MB)	Redo I/O data volume.

9.8 Memory Statistics

Table 9-9 lists metrics in the Memory Statistics report.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 9-9 Metrics in the Memory Statistics report

Statistical Item	Description
shared_used_memory	Size of the used shared memory, in MB.
max_shared_memory	Size of the maximum shared memory, in MB.
process_used_memory	Size of the used process memory, in MB.
max_process_memory	Size of the maximum process memory, in MB.

9.9 Time Model

Table 9-10 describes metrics in the Time Model report.

Table 9-10 Contents of the Time Model report

Name	Description
DB_TIME	Total end-to-end time consumed by all threads (unit: μs).
EXECUTION_TIME	Total time consumed on the executor (unit: μs).
PL_EXECUTION_TIME	Total time consumed for executing PL/SQL statements (unit: μs).
CPU_TIME	Total CPU time consumed by all threads (unit: μs).
PLAN_TIME	Total time consumed for generating an execution plan (unit: μs).
REWRITE_TIME	Total time consumed for rewriting queries (unit: μs).
PL_COMPILATION_TIME	Total time consumed for compiling SQL statements (unit: μs).
PARSE_TIME	Total time consumed for parsing SQL statements (unit: μs).
NET_SEND_TIME	Total time consumed for sending data over the network (unit: μs).
DATA_IO_TIME	Total time consumed for reading and writing data (unit: μs).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

9.10 SQL Statistics

Table 9-11 describes columns in the SQL Statistics report.

Table 9-11 Columns in the SQL Statistics report

Column	Description
Unique SQL Id	ID of a normalized SQL statement.
DB Name	Database name.
Parent Unique SQL Id	Normalized SQL ID of the parent statement. The value is **0** for non-stored procedure substatements.
Node Name	Node name.
User Name	Username.
Avg Tuples Fetched	Average number of rows randomly scanned.
Avg Tuples Returned	Average number of rows sequentially scanned.
Calls	Number of calls.
Total Elapsed Time(us)	Total execution time (unit: us).
Avg Elapsed Time(us)	Average execution time (unit: us).
Total Used Memory(MB)	Total used memory (unit: MB).
Max Used Memory(MB)	Maximum used memory of a single statement (unit: MB).
Min Used Memory(MB)	Minimum used memory of a single statement (unit: MB).
Returned Rows	Number of rows returned by SELECT.
Avg Returned Rows	Average number of rows returned by SELECT.
Tuples Affected	Number of rows affected by INSERT, UPDATE, and DELETE.
Avg Tuples Affected	Average number of rows affected by INSERT, UPDATE, and DELETE.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Logical Read	Number of logical reads on the buffer.
Avg Logical Read	Average number of logical reads on the buffer.
Physical Read	Number of physical reads on the buffer.
Avg Physical Read	Average number of physical reads on the buffer.
CPU Time(us)	CPU time (unit: us).
Avg CPU Time(us)	Average CPU time (unit: us).
Data IO Time(us)	Time spent on I/O (unit: us).
Avg Data IO Time(us)	Average time spent on I/O (unit: us).
Lock Wait Time(us)	Lock wait time of an SQL statements (unit: us).
Avg Lock Wait Time(us)	Average lock wait time of each SQL statement (unit: us).
Sort Count	Number of sorting execution times.
Sort Time(us)	Sorting execution time (unit: us).
Sort Mem Used(KB)	Size of work memory used during sorting (unit: KB).
Sort Spill Count	Number of file writes when data is flushed to disks during sorting.
Sort Spill Size(KB)	File size used when data is flushed to disks during sorting (unit: KB).
Hash Count	Number of hashing execution times.
Hash Time(us)	Hashing execution time (unit: us).
Hash Mem Used(KB)	Size of work memory used during hashing (unit: KB).
Hash Spill Count	Number of file writes when data is flushed to disks during hashing.
Hash Spill Size(KB)	File size used when data is flushed to disks during hashing (unit: KB).
SQL Text	Normalized SQL character string.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
SQL Hash	Hash value of the SQL statement.
Plan Hash	Hash value of the SQL statement execution plan.

9.11 Wait Events

Table 9-12 describes columns in the Wait Events report.

Table 9-12 Columns in the Wait Events report

Column	Description
Type	Wait events are classified as follows: • **STATUS** • **LWLOCK_EVENT** • **LOCK_EVENT** • **IO_EVENT**
Event	Name of a wait event.
Total Wait Time (us)	Total wait time (unit: us).
Waits	Total number of wait times.
Failed Waits	Number of wait failures.
Avg Wait Time (us)	Average wait time (unit: us).
Max Wait Time (us)	Maximum wait time (unit: us).

9.12 Cache IO Stats

The Cache IO Stats report consists of two tables, namely, **User table IO activity** and **User index IO activity**. Columns in the tables are described in Table 9-13 and Table 9-14.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

User table IO activity

Table 9-13 Columns in the User table IO activity table

Column	Description
DB Name	Database name.
Schema Name	Schema name.
Table Name	Table name.
%Heap Blks Hit Ratio	Hit ratio of the buffer pool in the table.
Heap Blks Read	Number of disk blocks read from the table.
Heap Blks Hit	Number of cache hits in the table.
Idx Blks Read	Number of disk blocks read from all indexes in the table.
Idx Blks Hit	Number of cache hits in indexes in the table.
Toast Blks Read	Number of disk blocks read from the TOAST table (if any) in the table.
Toast Blks Hit	Number of buffer hits in the TOAST table (if any) in the table.
Tidx Blks Read	Number of disk blocks read from the TOAST table index (if any) in the table.
Tidx Blks Hit	Number of buffer-hits in the TOAST table index (if any) in the table.

User index IO activity

Table 9-14 Columns in the User index IO activity table

Column	Description
DB Name	Database name.
Schema Name	Schema name.
Table Name	Table name.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Index Name	Index name.
%Idx Blks Hit Ratio	Index hit ratio.
Idx Blks Read	Number of disk blocks read from all indexes.
Idx Blks Hit	Number of cache hits of all indexes.

9.13 Utility status

The Utility status report consists of two tables, namely, **Replication slot** and **Replication stat**. Columns in the tables are described in **Table 9-15** and **Table 9-16**.

Replication slot

Table 9-15 Columns in the Replication slot table

Column	Description
Slot Name	Replication node name.
Slot Type	Type of the replication node.
DB Name	Name of the database on the replication node.
Active	Replication node status.
Xmin	Transaction ID of the replication node.
Restart Lsn	Xlog file information on the replication node.
Dummy Standby	Replication node as a dummy standby.

Replication stat

Table 9-16 Columns in the Replication stat table

Column	Description
Thread Id	PID of the thread.
Usesys Id	User system ID.
Usename	Username.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Application Name	Application name.
Client Addr	Client address.
Client Hostname	Client host name.
Client Port	Port of the client.
Backend Start	Start time of an application.
State	Log replication status.
Sender Sent Location	Location where the sender sends logs.
Receiver Write Location	Location where the receiver writes logs.
Receiver Flush Location	Location where the receiver flushes logs.
Receiver Replay Location	Location where the receiver replays logs.
Sync Priority	Synchronization priority.
Sync State	Synchronization status.

9.14 Object Stats

Object Stats contains three tables: User Tables stats, User index stats, and Bad lock stats. Columns in the tables are described as follows.

User Tables stats

Table 9-17 Contents in the User Tables stats table

Column	Description
DB Name	Database name.
Schema	Schema name.
Relname	Relation name.
Seq Scan	Number of sequential scans initiated in the table.
Seq Tup Read	Number of live rows fetched by sequential scans.
Index Scan	Number of index scans initiated in the table.
Index Tup Fetch	Number of live rows fetched by index scans.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Tuple Insert	Number of rows inserted.
Tuple Update	Number of rows updated.
Tuple Delete	Number of rows deleted.
Tuple Hot Update	Number of rows HOT updated (with no separate index update required).
Live Tuple	Estimated number of live rows.
Dead Tuple	Estimated number of inactive rows.
Last Vacuum	Last time at which this table was manually vacuumed (excluding VACUUM FULL).
Last Autovacuum	Last time when the table was vacuumed by the autovacuum daemon thread.
Last Analyze	Last time at which the table was manually analyzed
Last Autoanalyze	Last time when the table was analyzed by the autovacuum daemon thread.
Vacuum Count	Number of times this table is manually vacuumed (excluding VACUUM FULL).
Autovacuum Count	Number of times this table is vacuumed by the autovacuum daemon.
Analyze Count	Number of times this table is manually analyzed.
Autoanalyze Count	Number of times this table is analyzed by the autovacuum daemon thread.

User index stats

Table 9-18 Contents in the User index stats table

Column	Description
DB Name	Database name.
Schema	Schema name.
Relname	Relation name.
Index Relname	Index name.
Index Scan	Number of index scans initiated on the index.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Index Tuple Read	Number of index entries returned by scans on the index.
Index Tuple Fetch	Number of table rows fetched by simple index scans by using the index.

Bad lock stats

Table 9-19 Contents in the Bad lock stats table

Column	Description
DB Id	OID of the database.
Tablespace Id	Tablespace OID.
Relfilenode	File object ID.
Fork Number	File type.
Error Count	Number of failures.
First Time	First occurrence time.
Last Time	Last occurrence time.

9.15 Configuration settings

Table 9-20 describes columns in the Configuration settings report.

Table 9-20 Columns in the Configuration settings report

Column	Description
Name	GUC parameter name.
Abstract	GUC parameter description.
Type	Data type.
Current Value	Current value.
Min Value	Valid minimum value.
Max Value	Valid maximum value.
Category	GUC parameter type.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Enum Values	All enumerated values.
Default Value	Default parameter value used upon the database startup.
Reset Value	Default parameter value used upon the database reset.

9.16 SQL Detail

Table 9-21 describes columns in the SQL Detail report.

Table 9-21 Columns in the SQL Detail report

Column	Description
Unique SQL Id	ID of a normalized SQL statement.
Parent Unique SQL Id	Normalized SQL ID of the parent statement. The value is 0 for a statement that is executed independently.
DB Name	Database name. If the database that executes the query statement when the snapshot is generated does not exist, ***REMOVED_DB*** is displayed in the report.
User Name	Username. If the user who executes the query statement when the snapshot is generated does not exist, ***REMOVED_USER*** is displayed in the report.
Node Name	Node name. This column is not displayed in node mode.
SQL Hash	Hash value of the SQL statement.
Plan Hash	Hash value of the SQL statement execution plan.
SQL Text	Normalized SQL text.
Plan Text	Execution plan text of the SQL statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

10 ASP

Table 10-1 Report header

Column	Description
Start Time	Start time.
End Time	End time.
Slot Count	Number of time segments for displaying metrics.
Elapsed Time	Time segment for generating a report.
Sample Count	Total number of samples for generating ASP.
Slice Count	Total number of samplings.
Average Active Sessions	Average number of active sessions.
Boundary Time	Time point when samples in the memory and disk samples are split.
Data Source	Number of samples in the memory and on the disk for generating ASP.

Table 10-2 System load

Column	Description
Host Node Name	Node name.
CPUs	Total number of CPU cores.
Cores	Number of cores per CPU.
Sockets	Number of CPU slots.
Physical Memory	Memory size.
GaussDB Version	GaussDB version.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 10-3 Top client

Column	Description
Client Addr	Client address that accounts for a high percentage of sampled active sessions. The **client_hostname** and **client_port** columns in the **GS_ASP** table are empty, indicating that the client is connected through the UDS on the server or the thread is an internal thread. **NOTE** A client is an application or tool that establishes a connection and session with a database and interacts with a server process.
User Id	User ID.
Application Name	Name of the client connection program. **NOTE** In addition to the shielded background threads, background processes or clients (in the kernel) may exist, such as autovacuum, StandByRead, ASP, gsql, Archiver, WAL Reader and WAL Sender. Distributed processes may exist, such as GTM, CN, and DN. External client processes (outside the kernel) may exist, such as JDBC, ODBC, PSYCOPG2, DRS, and DBMind.
%Activity	Percentage of activity (Total number of active sessions/Number of samples).
Avg Active Sessions	Average number of active sessions (Total number of active sessions/ Number of aspects).

Table 10-4 Active sessions

Column	Description
sample_time (x_axis)	Sampling time.
count (y_axis)	Number of sampled events.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 10-5 Top events summary

Column	Description
Wait Status	Wait status name.
Wait Status Count	Number of wait statuses.
%Wait Status	Percentage of wait status (Number of wait status/Number of samples).
Avg Active Session	Average number of active wait status sessions (Number of wait status sessions/Number of aspects).
Event	Name of a wait event.
Event Count	Number of wait events.

Table 10-6 Top events graph

Column	Description
sample_time (x_axis)	Sampling time.
count (y_axis)	Number of sampled events.

Table 10-7 Top sessions with top event

Column	Description
Session Id	Session ID.
Session Start	Session start event.
Session Count	Number of active sessions.
%Activity	Percentage of activity (Total number of active sessions/Number of samples).
User Id	User ID.
Application Name	Name of the connection program.
Client Addr	Client address.
Client Port	Client port information.
Event	Active session event.
Event Count	Number of wait events of the session.
%Event	Percentage of session events (Number of events/Number of samples).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 10-8 Top sessions with top event split by slots

Column	Description
Slot time	Slot time.
Session Id	Session ID.
Session Start	Session start event.
Session Count	Number of session events.
%Activity	Percentage of activity (Total number of active sessions/Number of samples).
User Id	User ID.
Application Name	Name of the connection program.
Client Addr	Client address.
Client Port	Client port information.
Event	Active session event.
Event Count	Number of wait events of the session.
%Event	Percentage of session events (Number of events/Number of samples).

Table 10-9 Top sessions with top SQL

Column	Description
Session Id	Session ID.
Session Start	Session start event.
Session Count	Number of session events.
%Activity	Percentage of activity.
User Id	User ID.
Application Name	Name of the connection program.
Client Addr	Client address.
Client Port	Client port information.
SQL Id	SQL ID.
Parent SQL Id	SQL ID of the parent statement that calls the statement.
Node Id	Node ID.
SQL Text	SQL statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Table 10-10 Top sessions with top SQL split by slots

Column	Description
Slot Time	Slot time.
Session Id	Session ID.
Session Start	Session start event.
Session Count	Number of session events.
%Activity	Percentage of activity (Total number of active sessions/Number of samples).
User Id	User ID.
Application Name	Name of the connection program.
Client Addr	Client address.
Client Port	Client port information.
SQL Id	SQL ID.
Parent SQL Id	SQL ID of the parent statement that calls the statement.
Node Id	Node ID.
SQLText	SQL statement.

Table 10-11 Top final blocking sessions with top event split by slots

Column	Description
Slot Time	Slot time.
Final Blocking Session Id	ID of a blocked session.
Session Start	Start time of a session that holds a lock.
Blocked count	Number of sessions blocked by the session.
Blocked%Activity	Percentage of blocked sessions to active sessions (Total number of blocked sessions/Number of samples).
Event Caused	A wait event of final blocking session.
Event Caused Count	Number of wait events of final blocking session.
%Event	Percentage of wait events (Number of wait events/Number of samples).

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
PID	Thread ID.
User Id	User ID.
Application Name	Name of the connection program.
Client Addr	Client address.
Client Port	Client port information.
Current Xid	ID of the current transaction.
Top Xid	Top-level transaction ID.
Xact Start Time	Start time of a transaction that holds a lock.
Global Session Id	Global session ID

Table 10-12 Top blocked sessions with top SQL split by slots

Column	Description
Slot Time	Slot time.
Lock Tag	Lock tag.
Lock Mode	Lock mode.
Waiting Session Count	Number of waiting sessions.
Waiting Session Id	ID of a session waiting for a lock.
Session Start	Start time of a session that waits for a lock.
Session Count	Number of sessions that waits for a lock.
Block Session Id	ID of a blocked session.
Final Block Session Id	Session ID in the header of the blocked queue.
PID	Thread ID.
User Id	User ID.
Application Name	Name of the connection program.
Client Addr	Client address.
Client Port	Client port information.
Current Xid	ID of the current transaction.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
Top Xid	Top-level transaction ID.
Xact Start Time	Start time of a transaction that holds a lock.
Global Session Id	Global session ID
SQL Id	SQL ID.
Parent SQL Id	SQL ID of the parent statement that calls the statement.
Query Start	Start time of the SQL statement.
SQL Count	Number of SQL statements.
Node Id	Node ID.
SQL Text	SQL statement.

Table 10-13 Top SQL with top event

Column	Description
SQL Id	SQL ID.
Parent SQL Id	SQL ID of the parent statement that calls the statement.
Node Id	Node ID.
User Id	User ID.
Count	Number of session events.
%Activity	Percentage of activity (Total number of SQL sessions/Number of samples).
Event	Active session event.
Event Count	Number of wait events.
%Event	Percentage of session events (Number of SQL events/Number of samples).
SQL Text	SQL statement.

Table 10-14 Top SQL with top event split by slots

Column	Description
Slot Time	Slot time.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.

Column	Description
SQL Id	SQL ID.
Parent SQL Id	SQL ID of the parent statement that calls the statement.
Node Id	Node ID.
User Id	User ID.
Count	Number of session events.
%Activity	Percentage of activity (Total number of SQL sessions/Number of samples).
Event	Active session event.
Event Count	Number of wait events.
%Event	Percentage of session events (Number of SQL events/Number of samples).
SQL Text	SQL statement.

Table 10-15 Top SQL text

Column	Description
SQL Id	SQL ID.
Parent SQL Id	SQL ID of the parent statement that calls the statement.
Node Id	Node ID.
User Id	User ID.
SQL Text	SQL statement.

 Copyright © Huawei Cloud Computing Technologies Co., Ltd.